M1 - MIASHS IC Année 2021/2022

Manuel d'utilisation

Application de compagnon virtuel sur smartphone

Étudiants : Mathis Ruffieux et Néréïs Dugaleix

Encadrante : Mme. Lydie du Bousquet

Tuteur : Mr. Damien Pellier

Informations d’identification du document

Référence du document	D5
Version du document	1.01
Date du document	13/06/2022
Auteurs	Mathis Ruffieux et Néréïs Dugaleix

Éléments de vérification du document

Validé par	Mme. Lydie du Bousquet
Validé le	
Soumis le	13/06/2022
Type de diffusion	Document électronique (pdf)
Confidentialité	Non confidentiel

Sommaire

I) Introduction

L'application de Compagnon virtuel a pour objectif de communiquer quotidiennement avec un utilisateur afin de l'accompagner dans ses problèmes du quotidien en étant une oreille attentive.
Le robot lui posera des questions chaque jour et l'utilisateur pourra lui raconter sa journée et ses problèmes lorsqu'il le souhaite.

Après avoir utilisé l'application de manière régulière et si il donne son accord, un analyste (appelé aussi développeur) pourra brancher le téléphone de l'utilisateur sur son ordinateur afin de récupérer l'historique de conversation du dialogue et du journal (uniquement les messages et pages non secrètes). Aucun message posté par l'utilisateur n'est stocké ni récupéré sur un serveur web afin de garantir la confidentialité et l'immuabilité des données.

Dans ce manuel sont indiquées l'ensemble des fonctionnalités de l'utilisateur et du développeur afin d'en exploiter les fonctionnalités.

II) Manuel de l'utilisateur

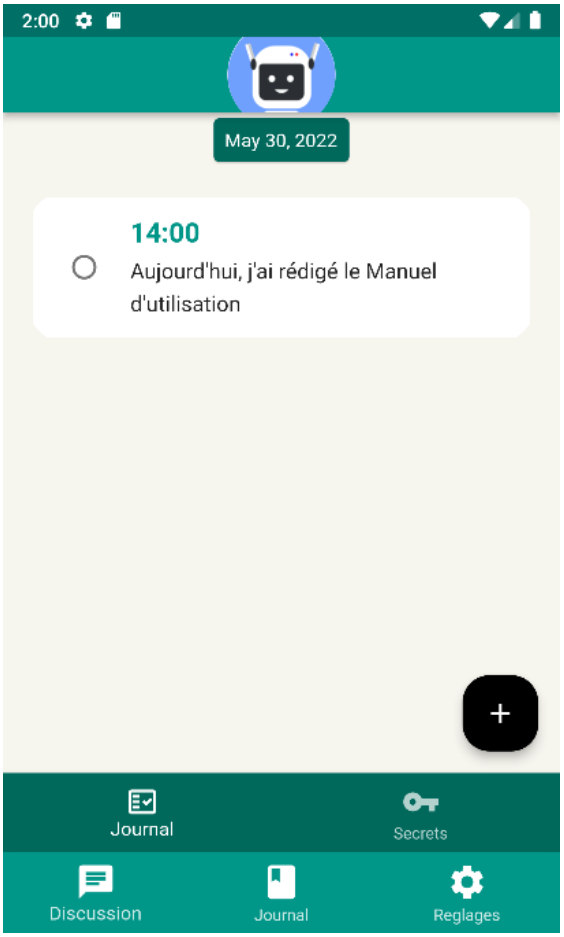
1) Page de discussion

Sur la page de discussion, un utilisateur peut répondre à une question du jour afin de démarrer un dialogue avec le robot.
Dans ce dialogue, le robot amène l'utilisateur à répondre soit par des questions fermées ou des questions ouvertes.

En dessous du dialogue se trouvent des boutons pour Aimer, Supprimer ou passer un message en Secret. Les messages catégorisés comme secrets ne seront pas lus par un analyste en cas d'exportation des données.

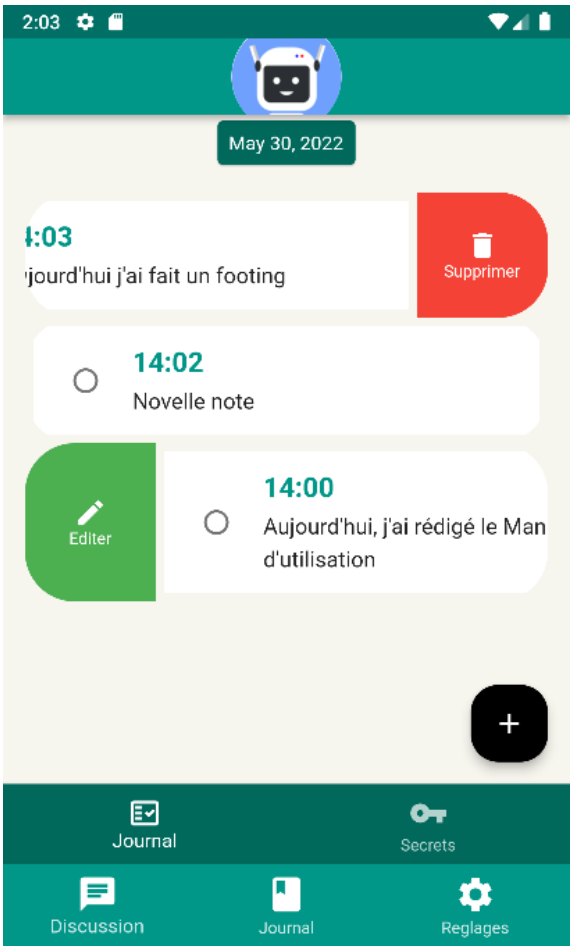
2) Page du journal

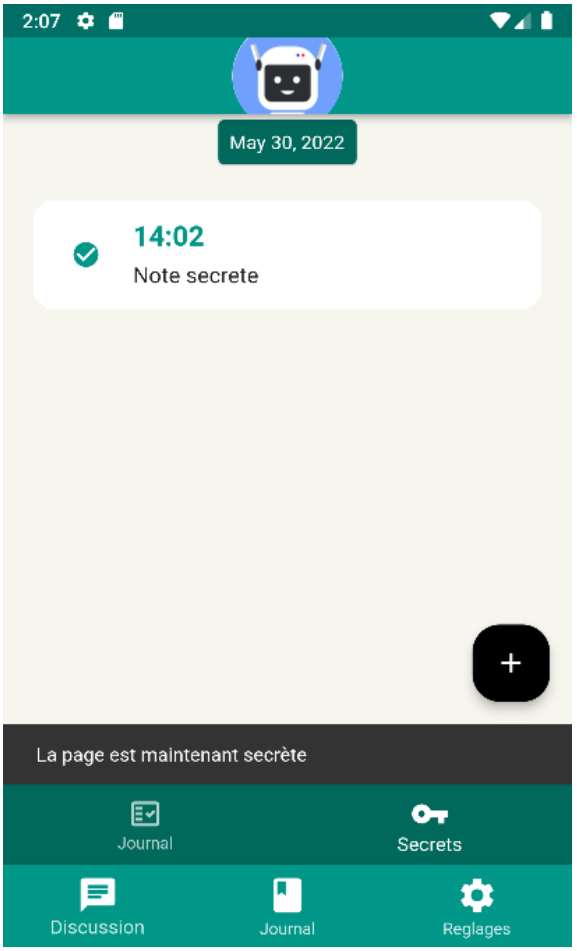
Sur la page du journal, un utilisateur peut ajouter des notes quotidiennes en cliquant l'icône “+” en bas à droite de l'écran. Ces notes peuvent être modifiées ou supprimées.



Afin de modifier ou de supprimer une page du journal, l'utilisateur peut déplacer une note vers la droite ou la gauche avec son doigt.
Il peut aussi cliquer directement sur la note pour la modifier.

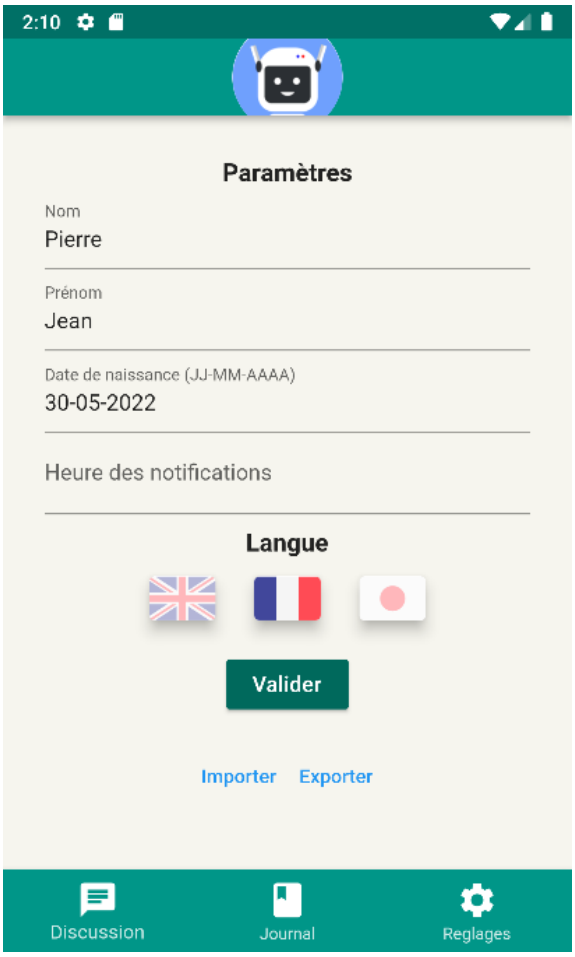
L'utilisateur peut cocher une note pour la faire passer en note secrète. Les notes secrètes sont accessibles depuis l'onglet Secrets.

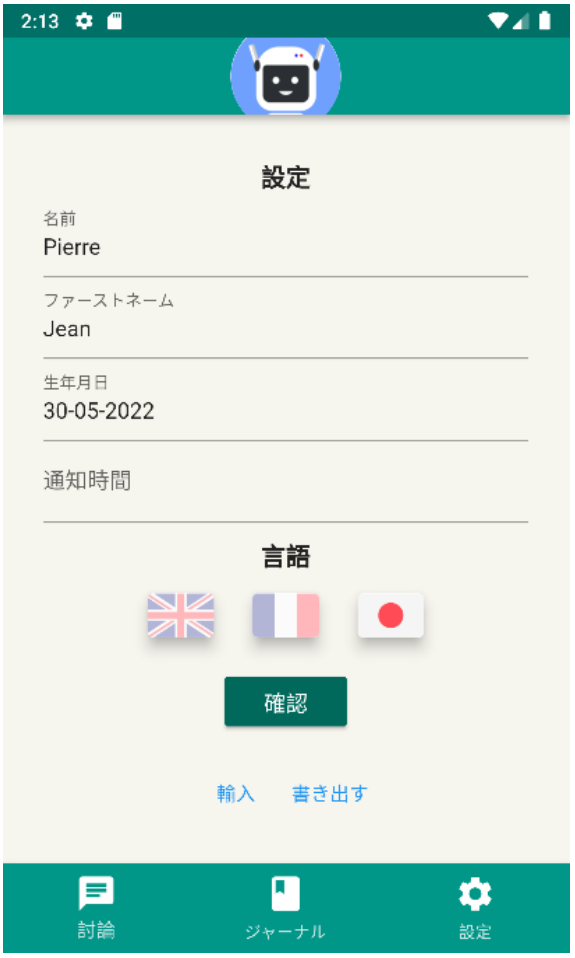




3) Page des réglages

Sur la page de réglages, l'utilisateur peut changer ses informations personnelles tel que son nom, prénom, âge et la langue de l'application.
Ces changements seront effectifs lorsqu'il clique sur le bouton valider.





Si vous n'êtes pas un utilisateur de type développeur, le manuel d'utilisation se termine ici.

III) Manuel du développeur

1) Export des données

L'analyste (ou le développeur) ne communique pas avec le robot. Il a pour rôle de récolter les informations à des fins d'analyses.
Pour celà, depuis le menu des options (annexe 8), il peut cliquer sur le bouton "Exporter" qui permet d'exporter l'historique de conversation de l'utilisateur avec le robot, en omettant les messages de type non-secrets, dans un fichier Json se trouvant dans le répertoire Download du téléphone de l'utilisateur.
Ensuite, il pourra récupérer ce fichier en connectant le téléphone sur son ordinateur avec un câble.

De la même façon sont exportées les notes du journal non secrètes dans un fichier distinct.
Le nom des deux fichiers générés ainsi dans le répertoire Download sont : *exportFileChat.json* et *exportFileJournal.json* .

2) Mise à jour des scénarios

L'analyste peut aussi modifier les scénarios du robot avec le bouton "Importer" du menu des options (annexe 8). Pour celà, il doit au préalable créer un fichier nommé *items_scenarios.json* et le placer dans le répertoire Download du téléphone de l'utilisateur, en suivant la documentation pour écrire ce fichier correctement.

IV) Documentation pour écrire les scénarios en Json

Pour mettre à jour les scénarios de l'application, il faut créer un fichier *items_scenarios.json* qui sera à placer dans le répertoire Download du téléphone de l'utilisateur avant de cliquer sur le bouton importer depuis l'application.
Lorsqu'on importe un fichier Json, les anciennes tables SQL sont supprimées et recréées avec ce nouveau fichier. Il faut donc conserver les scénarios antérieurs que l'on souhaite garder dans le fichier Json et y ajouter les nouveaux scénarios à la suite en respectant bien l'ordre des identifiants.
Le scénario numéro 1 est réservé au scénario de bienvenue (lancé une seule fois après le téléchargement de l'application). Les scénarios des questions du jour démarrent à l'idScénario 2.

Il est possible de reprendre le fichier *items_scenarios.json* sur le github et de le modifier directement ou de s'en servir comme exemple.

Ce fichier doit contenir 3 champs de type listes :

- Un champ "questions" correspondants aux objets des questions,
- Un "replies" correspondant aux objets des réponses fermées,
- et un champ "relationsQR" correspondant aux objets des relations entre les questions et les réponses fermées.

1) Les questions

Une question correspond à une réplique du robot. Elle peut être suivie d'une autre question, d'une réponse fermée, d'une réponse ouverte, ou d'une fin.

a) Les ids des questions

Les questions ont toutes un “id”, il doit démarrer à 1 et doivent être écrites dans l’ordre de leur identifiant.
Il faut impérativement écrire “idScénario” à chaque question. Les questions sans cet idScénario ne seront pas lus par l’application. (Il est donc possible de créer des questions vide afin d’incrémenter les id artificiellement)

b) Les champs de texte des questions

Les questions disposent de 3 champs de texte :

- “textEN” pour la question en anglais
- “textFR” pour la question en français
- “textJP” pour la question en japonais

Il est possible de laisser une chaîne de caractère vide sur les langues afin de les écrire plus tard.

c) Ecrire une variable dans une question

Pour écrire une variable dans une question on écrira %%[nom de la variable]
Par exemple pour écrire son nom on écrit : “Bonjour %%name !”
Il faut que le double ‘%’ soit collé au nom de la variable et qu’il y ait un caractère espace après. Lorsque le robot parlera, il remplacera cette chaîne de caractères par la variable correspondante au nom qu’il aura trouvé en base.

d) Les paramètres des questions

Lorsque la question est la première question d’un scénario, on lui précise “isFirst” à 1 (pour true).
Si on ne lui précise pas “isFirst” il sera par défaut à 0 (pour false).
Attention il ne faut pas qu’il y ait plusieurs questions “isFirst” à 1 pour un même scénario.
Si il n’y a aucune question “isFirst” pour un scénario, il ne sera jamais initialisé.
Si une question est la dernière question d’un scénario, on lui précise “isEnd” à 1.
“isEnd” est par défaut à 0 si on ne le précise pas.
Un scénario peut avoir plusieurs questions “isEnd” à 1 ou aucune question “isEnd” à 1 si on souhaite terminer par une réplique de l’utilisateur (et non une réplique du robot).

- Si une question est suivie d’une autre question (sans réponse de l’utilisateur intermédiaire), on lui précise “idNextQuestion” correspondant à l’id de la prochaine question.

- Si une question est suivie d'une réponse ouverte, on lui précise "isOpenQuestion" à 1 (par défaut 0) **ET** on lui précise aussi un "idNextQuestion" pour savoir quelle question poser une fois que l'utilisateur aura répondu.

Dans ce cas, si on souhaite enregistrer en base la réponse ouverte de l'utilisateur on lui précise "nameVariable" avec une chaîne de caractères correspondant au nom de la variable à enregistrer. Par exemple, si on souhaite enregistrer son fruit favori, on écrira "nameVariable" : "favoriteFruit".
On pourra ainsi retrouver ce qu'il a écrit lorsqu'on ira chercher en base la variable nommée "favoriteFruit" avec un %%.

- Si une question est suivie d'une réponse fermée, on ne lui précise rien car c'est le rôle de la table "relationsQR" de faire la jonction entre les questions et une liste de réponses fermées.

Dans ce cas on ne lui précise pas non plus un "idNextQuestion" (en le laissant par défaut à 0) car en fonction de la réponse de l'utilisateur on pourra arriver sur une question différente.

2) Les réponses

Une réponse correspond à une réponse fermée de l'utilisateur, sur laquelle il pourra cliquer pour répondre à une question.

a) Les ids des réponses

Comme pour les questions, les objets réponses disposent d'un "id" (écrit dans l'ordre) et d'un "idScenario" correspondant à l'id du scénario. Une réponse fait donc forcément partie d'un seul scénario à la fois.
Si une réponse n'a pas de "idScenario" elle ne sera pas interprétée par l'application.

b) Les champs de texte des réponses

Les réponses disposent aussi de 3 champs de texte :
- "textEN" pour la réponse en anglais
- "textFR" pour la réponse en français
- "textJP" pour la réponse en japonais

c) Les variables des réponses

Contrairement aux questions, les réponses ne peuvent pas appeler de variable avec '%%'. Par exemple, le '%%name' ne sera pas interprété et écrit tel quel dans la réponse.

d) Les paramètres des réponses

Une réponse est toujours suivie d'une question "idQuestion" correspondant à l'id de la prochaine réplique du robot, sauf dans le cas où une réponse termine le scénario sans question suivante.
Dans le cas où une réponse est la dernière réplique d'un scénario, il suffit de ne pas lui préciser "idQuestion" qui sera automatiquement initialisé à 0 ce qui signifie une question nulle.

Si on souhaite enregistrer une réponse fermée dans une variable en base on lui précise "nameVariable" avec le nom de la variable. Le contenu enregistré sera la chaîne de caractère de la réponse dans la langue de l'utilisateur.
Par exemple si "nameVariable" = "gender" et qu'il clique sur "Homme", alors la chaîne de caractères "Homme" sera enregistrée dans la variable nommée "gender".

3) Les relations question / réponse

Une relation Q/R est une jonction entre une question et une réponse fermée.
Une même question peut avoir zéro, une ou plusieurs réponses. Mais une réponse ne peut avoir qu'une seule question.

a) Les ids des relations Q/R

Les objets relationsQR n'ont pas besoin d'id dans le fichier Json, car on ne s'en sert pas. Cependant il est impératif d'écrire leur "idScénario" sinon ils ne seront pas lu par l'application.
A noter qu'on ne peut pas joindre une question et une réponse de deux scénarios différents.

b) Les paramètres des relations Q/R

Un objet relation Q/R dispose donc d'un id de question "idQuestion" et d'un id de réponse "idReply".

V) Messages d'erreur (pour les développeurs)

Il n'y a pas de message d'erreur directement affiché sur l'application.

Cependant si on exécute l'application via un IDE tel que VScode, il est possible de voir des messages dans la Debug console. Il ne devrait normalement pas y en avoir sauf le premier (Path invalide) lors de l'import d'un fichier inexistant.

Ces messages sont les suivants :

Error : Path invalide : Lorsqu'on clique sur le bouton d'import, celà signifie que le fichier *items_scenarios.json* n'a pas été trouvé dans le répertoire download du téléphone.

Error : createVariable failed : Signifie que la fonction createVariable a échouée à créer une nouvelle variable en BD.

Error : table variables doesn't exist : Signifie que la table des variables n'a pas été trouvée ou n'existe pas.

Error : theQuestion.length != 1 : Signifie que dans les scénarios, une réponse dispose de 0 ou plusieurs questions suivantes alors qu'elle ne doit en avoir qu'une et une seule. Si cette erreur apparaît, le fichier Json a été mal conçu.

Error : No first question : Signifie que le scénario appelé ne dispose pas de question pour démarrer. Si cette erreur apparaît, le fichier Json a été mal conçu.

Error : variable $name not found : Signifie qu'une variable lu dans la BD n'a pas été trouvée.

Error : ID question not found : Une question correspondant à un ID n'a pas été trouvée. Si cette erreur apparaît, le fichier Json a été mal conçu.

Error : ID reply not found : Une réponse correspondante à un ID n'a pas été trouvée. Si cette erreur apparaît, le fichier Json a été mal conçu.

Error : Scénario invalide : Le scénario appelé est invalide. Le plus souvent il ne dispose pas de firstQuestion, il est donc invalide. Si cette erreur apparaît, le fichier Json a été mal conçu.

Error : lang $lang : Une autre langue que en, fr ou jp a été écrite en base. Si cette erreur survient l'application s'affiche par défaut en anglais.

Error : random scénarios vide : Il n'y a aucun scénario dans les bases, on ne peut donc pas démarrer un scénario aléatoire. Si cette erreur apparaît, le fichier Json est vide ou mal conçu.

VI) Remerciements

Nous tenons tout d'abord à remercier nos professeurs du master MIASHS parcours Informatique et cognition, en particulier notre tuteur de stage, Mr Damien Pellier de nous avoir offert l'opportunité de réaliser ce projet. Le suivi hebdomadaire et l'aide qu'il nous a fourni tout au long de l'année scolaire nous ont permis d'avancer efficacement tout en gardant un rythme de travail soutenu.

Nous souhaitons également remercier notre encadrante de stage, Mme Lydie Du Bousquet, qui nous a fait confiance pour la réalisation de ce projet et nous a indiqué le bon chemin à suivre pour son développement, en nous donnant toutes les informations nécessaires au bon développement de celui-ci lors de suivis réguliers.

Nous tenons surtout à remercier la région Auvergne-Rhône-Alpes, qui au travers de l'UGA à financer notre déplacement au Japon dans la région du Kansai, lieu de naissance du projet SCUSI Kouno Tori de compagnon virtuel sur smartphone.
De plus, avoir une expérience de stage à l'international est pour nous un atout primordial dans le secteur de l'informatique et sans leur financement et leur aide nous n'aurions pas pu effectuer ce voyage pour aller au bout de ce projet.

A ce titre, nous souhaitons tout particulièrement remercier Mr. Masahide Nakamura, professeur à l'Université de Kobe, de nous accueillir au sein de son établissement afin de poursuivre et de nous aider au développement d'un agent intelligent de compagnon virtuel, qui fait l'objet de certaines de ses recherches.

Nous n'oublions pas aussi toutes les personnes de l'administration des différentes parties, de l'Université Grenoble Alpes et de l'Université de Kobe qui nous ont aidé et conseillé sur de nombreux points.

Enfin, nous remercions toutes les personnes qui se sont portées volontaires pour tester et valider notre application mobile.