

---

## RAPPORT DE STAGE

---

### SEMAINE 1 :

#### Jour 1 : 30.05.2023

Nous avons **découvert l'application AKEAD Tech**, une plateforme de gestion de tâches et de rendez-vous développée avec le framework Flutter et le langage Dart pour le front end. Le back end est du php. Cette application est développée partiellement par d'autres stagiaires. On nous a demandé de finir le développement de l'application durant notre stage.

L'après-midi, nous avons travaillé avec un autre stagiaire pour **créer notre propre application mobile (Portfolio)** en utilisant Flutter. Cela nous a permis de nous familiariser davantage avec ce framework et de tester notre application sur un émulateur Android généré par Android Studio. Pour le développement, j'ai utilisé VS Code.

Cela nous a donné l'occasion d'apprendre davantage sur ce framework et son langage, et d'appliquer nos connaissances en développement mobile avec Flutter.

#### Jour 2 : 31/05/2023

Nous avons commencé à développer un projet avancé lors de notre stage : **une application de shopping**. L'objectif était de maîtriser en profondeur **l'utilisation de Flutter pour le développement du front-end**.

L'application que nous avons développée comprend une interface utilisateur avec une page dédiée aux produits. Les utilisateurs peuvent ajouter des articles au panier. Notre tuteur nous a proposé ce projet pour nous plonger complètement dans Flutter et renforcer notre expertise.

Pendant le développement, notre tuteur nous a fourni un accompagnement et des conseils. Ces échanges nous ont aidés à surmonter les difficultés techniques et à progresser.

#### Jour 3 : 01/06/2023

Aujourd'hui, notre objectif était de continuer à travailler sur l'application de shopping et d'apprendre à utiliser une **API**.

Nous avons réussi à intégrer **l'interaction avec une API** dans notre application en utilisant **le package HTTP**. Pour cela, nous avons utilisé l'URL spécifique fournie par JSONPlaceholder.

En nous connectant à cette URL à l'aide des requêtes HTTP appropriées, nous avons pu récupérer les données nécessaires pour enrichir notre application de shopping.

Cela nous a permis d'afficher des produits dynamiques à partir des informations fournies par JSONPlaceholder.

#### **Jour 4 : 02/06/2023**

Aujourd'hui, nous avons travaillé sur la création d'une **application mobile de gestion de tâches appelée To-do List**. L'objectif principal était d'intégrer l'interaction avec une API en utilisant le package HTTP, ce que nous avons déjà abordé précédemment.

Voici les fonctionnalités principales que nous avons implémentées dans cette application sont la récupération des données à partir d'une API, l'affichage des tâches dans une liste, l'affichage du nombre des tâches réalisées ou non réalisées, la suppression des tâches et la modification des tâches.

### **SEMAINE 2 :**

#### **Jour 5 : 05/06/2023**

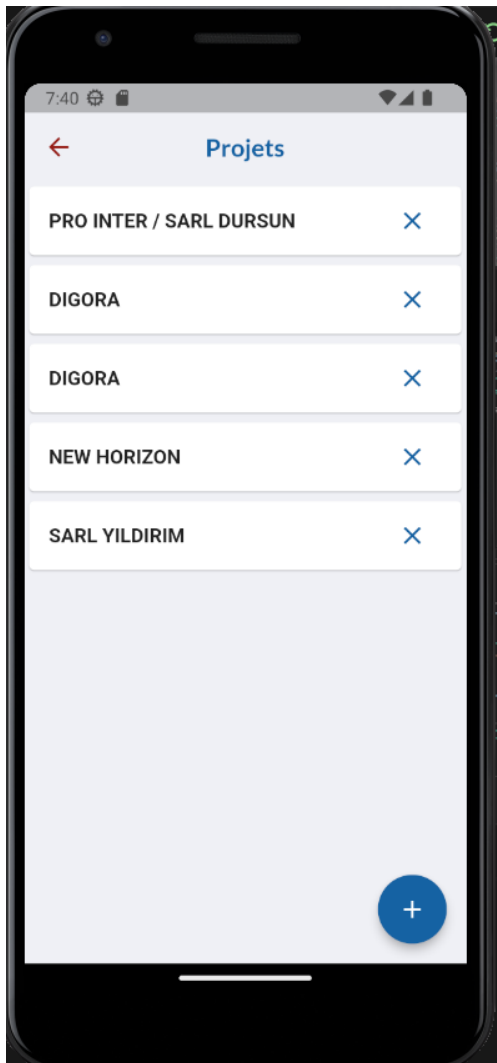
Après une semaine d'apprentissage et de pratique avec Flutter, on va maintenant travailler sur l'application mobile Akead Tech. Mon tuteur m'a guidé dans l'utilisation de GitHub pour la gestion du code source et m'a expliqué les différentes parties du code existant, ainsi que les éléments spécifiques dont nous aurons besoin pour cette fonctionnalité. J'ai également été initié à l'utilisation de la base de données.

Notre objectif principal était de créer une section dans l'application mobile qui permettrait aux utilisateurs de visualiser les tâches à effectuer pour un client spécifique. Cela nous permettrait de suivre l'avancement des tâches pour chaque client.

Pour commencer, nous avons ajouté les tables nécessaires à la base de données en utilisant Symfony, un framework de développement web PHP. Ensuite, nous avons créé les modèles correspondants à ces tables, tels que "Project.dart", "SubProject.dart" et "Todo.dart". Par la suite, nous avons développé les API requises pour récupérer les données nécessaires à partir de la base de données.

#### **Jour 6 : 06/06/2023**

Aujourd'hui, nous avons poursuivi le travail entamé hier. J'ai terminé la partie de récupération des projets, ainsi que celle permettant de récupérer les sous-projets liés à ces projets. J'ai également apporté des améliorations esthétiques pour rendre l'interface plus agréable visuellement.

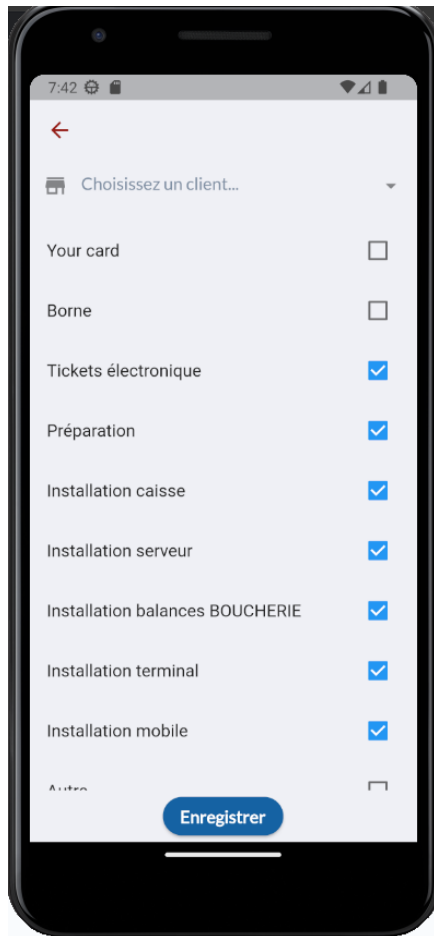


J'ai créé la page dédiée à l'ajout des sous-projets, mais je n'ai pas encore terminé son implémentation à la fin de cette journée. Je me suis concentré sur la mise en place des fonctionnalités principales et j'ai pris le temps de peaufiner le design et l'apparence générale de l'application.

Malgré le fait que la page d'ajout des sous-projets ne soit pas complètement terminée, je suis satisfait de mon avancement aujourd'hui. J'ai pu accomplir les tâches prévues et j'ai hâte de continuer à travailler demain pour finaliser cette fonctionnalité.

### **Jour 7 : 07/06/2023**

Aujourd'hui, j'ai poursuivi la création de la page permettant de créer des sous-projets après avoir sélectionné un projet. J'ai également mis en place le fournisseur de données (provider) et le service pour la gestion des projets, afin d'ajouter à la fois des projets et des sous-projets. J'ai ajouté une condition pour éviter l'ajout de projets en double dans la liste.



Ensuite, j'ai consacré une grande partie de ma journée au débogage, en vérifiant et en corrigeant les erreurs dans mon code. Cela m'a permis d'améliorer le fonctionnement global de l'application et d'assurer une meilleure expérience utilisateur.

Je suis satisfait de mes progrès aujourd'hui et je suis prêt à continuer à travailler sur les fonctionnalités restantes.

### **Jour 8 : 08/06/2023**

Aujourd'hui, j'ai travaillé sur l'ajout des fonctionnalités de suppression des projets dans le Provider et le Service. Mon tuteur a également mis en place l'API nécessaire pour que mon code fonctionne correctement. Ensuite, j'ai commencé à travailler sur la fonctionnalité d'ajout des sous-projets pour un projet existant. J'ai créé la page EditProjectScreen pour permettre la modification du projet sélectionné.

En fin de journée, nous avons eu une réunion avec les employés de l'entreprise. Au cours de cette réunion, ils nous ont présenté d'autres fonctionnalités et automatisations qu'ils aimeraient voir mises en place dans l'application. Nous avons discuté des exigences et des priorités, afin de bien comprendre les attentes de l'entreprise.

### **Jour 9 : 09/06/2023**

Ce matin, notre tuteur n'était pas disponible, ce qui nous a empêché d'accéder à la base de données et de poursuivre le développement de l'application Akead Tech. En conséquence, nous avons décidé de profiter de ce temps libre pour regarder des tutoriels sur Flutter et approfondir nos connaissances dans le domaine.

Nous avons également saisi l'occasion pour discuter de la suite du projet avec mon collègue. Nous avons échangé sur les fonctionnalités à implémenter, les défis techniques à relever et les prochaines étapes de développement. Cette réunion nous a permis de mieux planifier notre travail et de clarifier nos objectifs pour les jours à venir.

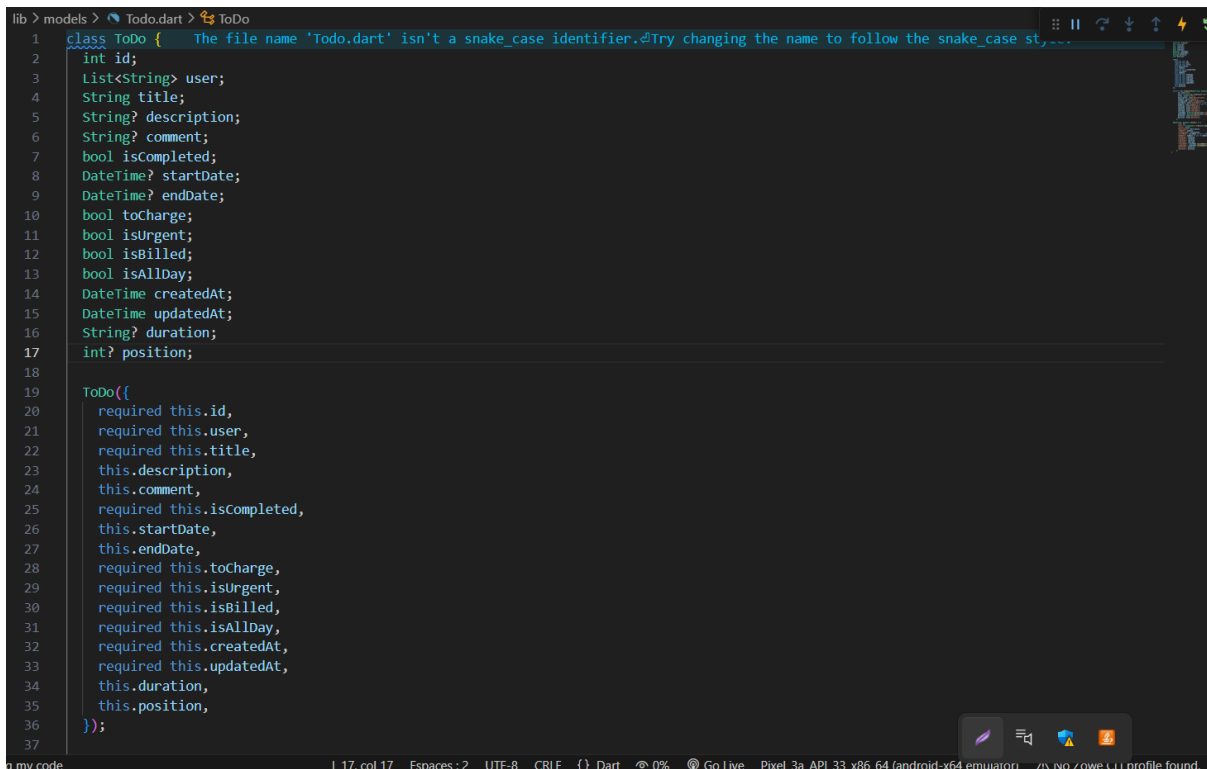
## SEMAINE 3 :

### Jour 10 : 12/06/2023

Ce matin, nous avons eu une petite réunion générale suivie d'une réunion spécifique à notre projet de "Gestion de projet". Chacun a eu l'occasion de proposer des idées pour améliorer notre application, tant pour les employés que pour le patron. Après avoir obtenu les instructions nécessaires, nous sommes maintenant prêts à réaliser leurs demandes.

L'après-midi, nous avons travaillé sur la fonctionnalité des Todos prédéfinis pour chaque sous-projet. Notre tuteur a créé l'API nécessaire pour les Todos, et j'ai mis en place le modèle "Todos.dart".

Todos.dart :



```
lib > models > Todos.dart > class Todo
1 class Todo {
2   int id;
3   List<String> user;
4   String title;
5   String? description;
6   String? comment;
7   bool isCompleted;
8   DateTime? startDate;
9   DateTime? endDate;
10  bool toCharge;
11  bool isUrgent;
12  bool isBilled;
13  bool isAllDay;
14  DateTime createdAt;
15  DateTime updatedAt;
16  String? duration;
17  int? position;
18
19  Todo({
20    required this.id,
21    required this.user,
22    required this.title,
23    this.description,
24    this.comment,
25    required this.isCompleted,
26    this.startDate,
27    this.endDate,
28    required this.toCharge,
29    required this.isUrgent,
30    required this.isBilled,
31    required this.isAllDay,
32    required this.createdAt,
33    required this.updatedAt,
34    this.duration,
35    this.position,
36  });
37
```

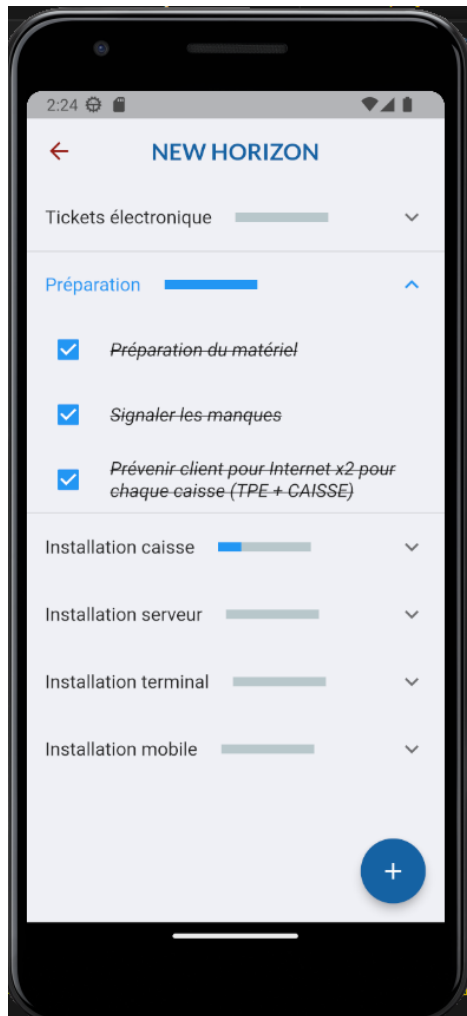
```

factory ToDo.fromJson(Map<String, dynamic> json) => ToDo(
  id: json["id"],
  user: List<String>.from(json["user"].map((x) => x)),
  title: json["title"],
  description: json["description"],
  comment: json["comment"],
  isCompleted: json["isCompleted"],
  startDate: json["startDate"] != null ? DateTime.parse(json["startDate"]) : null,
  endDate: json["endDate"] != null ? DateTime.parse(json["endDate"]) : null,
  toCharge: json["toCharge"],
  isUrgent: json["isUrgent"],
  isBilled: json["isBilled"],
  isAllDay: json["isAllDay"],
  createdAt: DateTime.parse(json["createdAt"]),
  updatedAt: DateTime.parse(json["updatedAt"]),
  duration: json["duration"],
  position: json["position"],
); // ToDo

Map<String, dynamic> toJson() => {
  "id": id,
  "user": List<dynamic>.from(user.map((x) => x)),
  "title": title,
  "description": description,
  "comment": comment,
  "isCompleted": isCompleted,
  "startDate": startDate != null ? startDate!.toIso8601String() : null,
  "endDate": endDate != null ? endDate!.toIso8601String() : null,
  "toCharge": toCharge,
  "isUrgent": isUrgent,
  "isBilled": isBilled,
  "isAllDay": isAllDay,
  "createdAt": createdAt.toIso8601String(),
  "updatedAt": updatedAt.toIso8601String(),
  "duration": duration,
  "position": position,
};
}

```

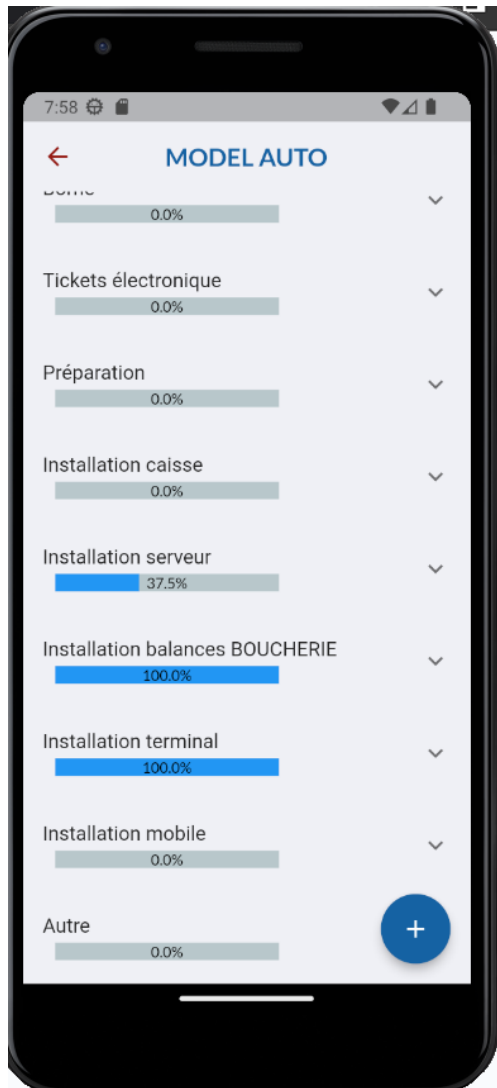
J'ai utilisé le widget ExpansionTile() pour afficher les Todos de chaque sous-projet, avec des cases à cocher. J'ai également intégré le package 'percent\_indicator' pour afficher le pourcentage de cases cochées, ajoutant ainsi une dimension visuelle à notre application.



### Jour 11 : 13/06/2023

J'ai continué de faire la barre de progression pour que ça soit plus esthétique en ajoutant le pourcentage dans la barre de progression.

```
return ExpansionTile(  
  title: Text(subProject.name),  
  subtitle:  
  LinearPercentIndicator( //barre de pourcentage  
    // width: 200.0,  
    // lineHeight: 8.0,  
    // percent: completionPercentage,  
    // progressColor: Colors.blue,  
  
    width: 200.0,  
    lineHeight: 14.0,  
    percent: completionPercentage,  
    center: Text("${(completionPercentage * 100).toStringAsFixed(1)}%",  
      style: new TextStyle(fontSize: 12.0), Use 'const' with the construc  
    ), // Text  
    // backgroundColor: Colors.grey,  
    progressColor: Colors.blue,  
  ), // LinearPercentIndicator  
)
```





Aujourd'hui, nous avons travaillé sur l'enregistrement des boutons cochés dans la base de données en utilisant une API créée par votre tuteur. Nous avons également apporté des modifications à la structure de la base de données en créant un nouveau modèle appelé "ProjectTodo" et en modifiant les modèles "Project" et "SubProject".

Pour mettre à jour la propriété "isCompleted" d'un todo dans la base de données, j'ai ensuite créé un provider et un service. Cela m'a permis de faire une requête vers l'API pour mettre à jour l'état du todo.

Le service :

```
Future<void> updateTodoState(int todoId, bool isCompleted) async {  
  final response = await _dioClient.dio  
    .put('/project_todo/$todoId/status', data: json.encode({'isCompleted': isCompleted}));  
  // erreurs  
  if (response.statusCode != 200) {}  
}
```

### **Jour 12 : 14/06/2023**

Aujourd'hui, nous avons travaillé sur le débogage de la barre de pourcentage dans la partie "Gestion de projet". Nous avons résolu le problème qui faisait que la barre de pourcentage revenait à 100% lorsque vous sortiez puis rentriez dans cette partie. Le problème était dû au fait que l'état "isCompleted" des todos n'était pas correctement renvoyé à la base de données lorsque les todos étaient cochés.

Ensuite, nous avons commencé à ajouter un bouton "Tout sélectionner" qui permet de sélectionner tous les todos d'un sous-projet. Cependant, nous n'avons pas terminé ces implémentations à la fin de la journée.

### **Jour 13 : 15/06/2023**

Aujourd'hui, j'ai ajouté un bouton "Tout sélectionner" pour permettre à l'utilisateur de sélectionner tous les todos d'un sous-projet. J'ai également ajouté un bouton "+" pour ajouter un nouveau todo dans le sous-projet. De plus, j'ai ajouté un bouton "..." pour donner à l'utilisateur la possibilité de modifier et supprimer les sous-projets. J'ai ajusté la taille des boutons pour éviter les erreurs d'affichage.

Nouveau todo :

```
// nouv todo
showDialog(
  context: context,
  builder: (context) {
    return AlertDialog(
      title: Text('Ajouter un nouveau todo'),    Use 'const'
      content: TextField(
        controller: TextEditingController(),
        decoration: InputDecoration(    Use 'const' with the
          labelText: 'Titre du todo',
        ), // InputDecoration
      ), // TextField
      actions: [
        TextButton(
          onPressed: () {
            // Ferme le dialogue
            Navigator.of(context).pop();
          },
          child: Text('Annuler'),    Use 'const' with the con
        ), // TextButton
        TextButton(
          onPressed: () {
            // Ajoute nouv todo a la liste
            final newTodoTitle =
              TextEditingController().text;
            // ajout todo
            context
              .read<ProjectProvider>()
              .addTodos(subProject, newTodoTitle);
            // Ferme dialogue
            Navigator.of(context).pop();
          },
          child: Text('Ajouter'),    Use 'const' with the con
        ), // TextButton
      ],
    ); // AlertDialog
  },
);
```

Modifier, supprimer :

```

    PopupMenuButton(
      itemBuilder: (context) => [
        PopupMenuItem(
          child: Text('Modifier'),      Use 'const' w
          onTap: () {
            // modifier le subProject.name
          },
        ), // PopupMenuItem
        PopupMenuItem(
          child: Text('Supprimer'),      Use 'const'
          onTap: () {
            // supprimer le subProject.name
          },
        ), // PopupMenuItem
      ],
      icon: Icon(Icons.more_vert),      Use 'const' w
    ), // PopupMenuButton
  ],
), // Row

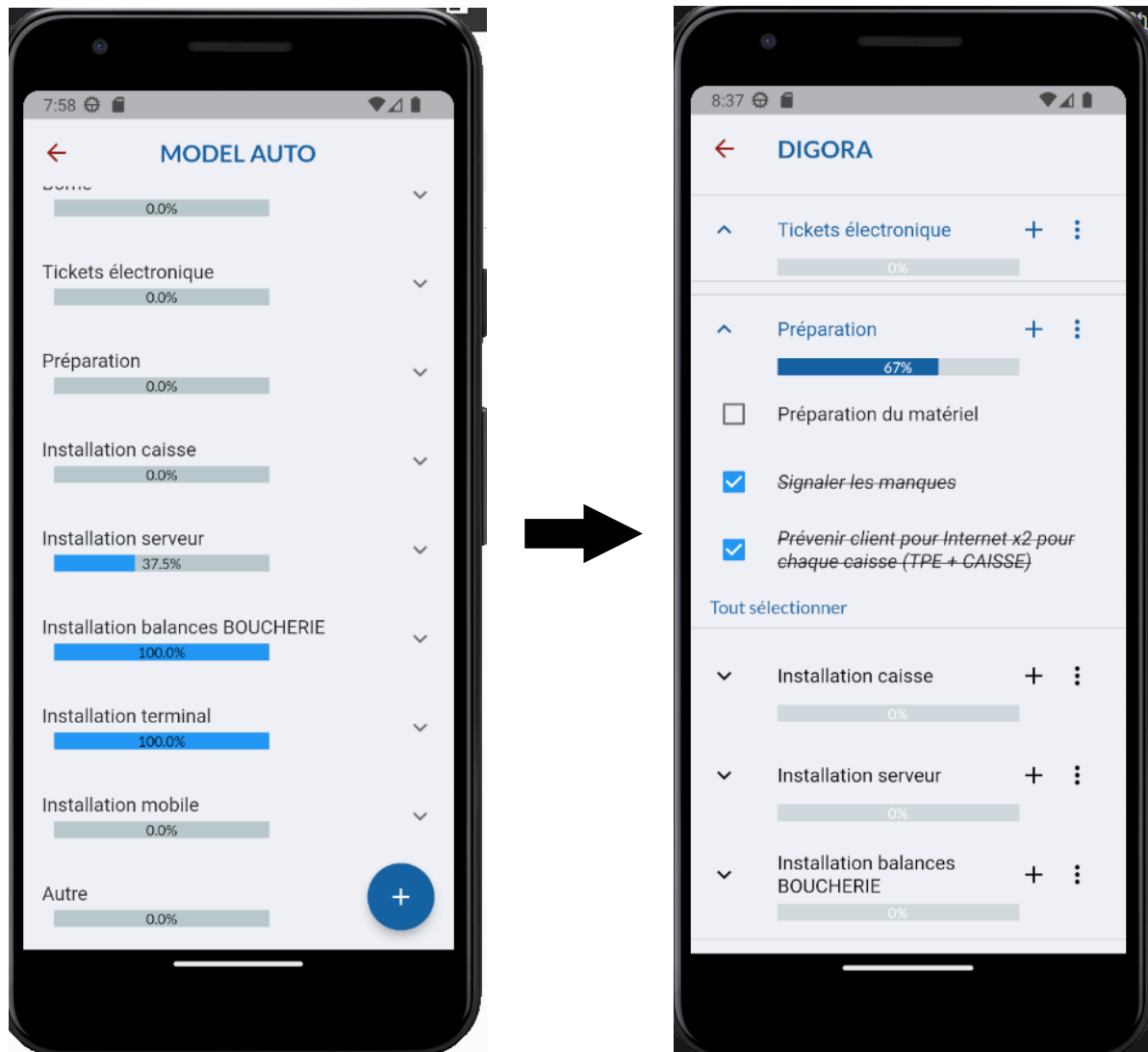
```

En ce qui concerne les providers et services nécessaires pour réaliser ces actions, j'ai réussi à mettre en place celui pour le bouton "Tout sélectionner". Cependant, nous sommes bloqués sur la mise en place de la fonction "addTodos()". Nous avons recherché des solutions sur internet, consulté des forums et même demandé de l'aide à ChatGPT, mais nous n'avons pas réussi à résoudre le problème.

Comme notre tuteur n'était pas disponible aujourd'hui, nous avons fait de notre mieux pour avancer en attendant qu'il puisse nous débloquer demain.

#### **Jour 14 : 16/06/2023**

Le matin, nous avons travaillé avec notre tuteur pour créer une API permettant de supprimer un sous-projet. Nous avons ensuite créé les providers et services nécessaires pour cette fonctionnalité. Nous avons également corrigé des erreurs d'affichage et apporté des modifications à l'apparence des sous-projets et des todos.



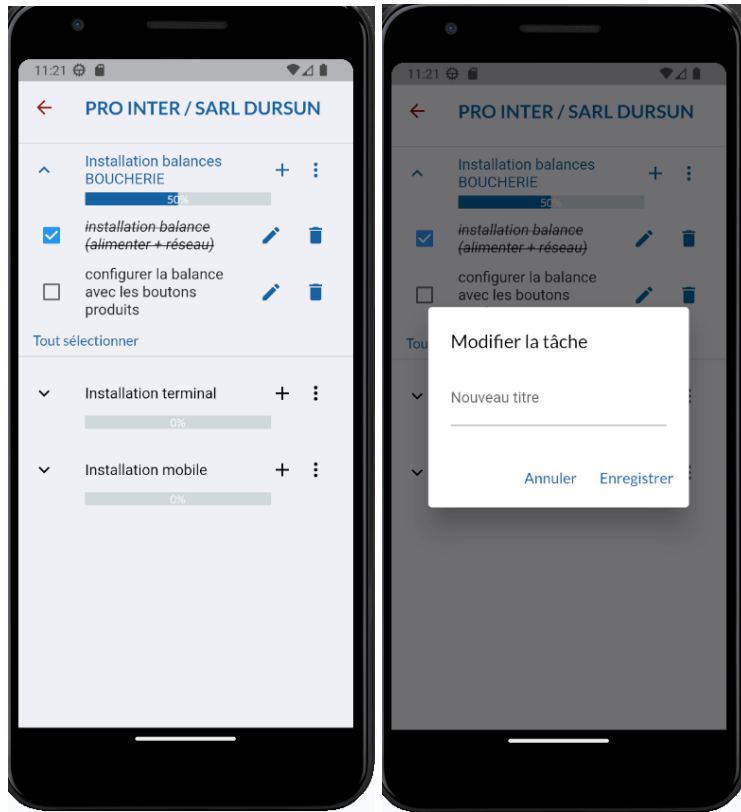
Dans l'après-midi, nous avons ajouté une nouvelle API pour l'ajout de todos, puis nous avons créé le provider et le service correspondants pour gérer cette fonctionnalité. Cependant, nous avons rencontré des problèmes et des messages d'erreur sont apparus, ce qui nous a amenés à demander l'aide de notre tuteur pour déboguer et trouver des solutions.

## **SEMAINE 4 :**

### **Jour 15 : 19/06/2023**

Ce matin, nous avons eu notre réunion de début de semaine habituelle. La réunion a duré jusqu'à 10h30, ce qui m'a laissé du temps pour approfondir mes connaissances sur Flutter. J'avais du mal à comprendre certaines parties, notamment la séparation des providers et des services dans notre code. Après recherche, j'ai compris que cette séparation permet une meilleure organisation, facilite la réutilisation, améliore la testabilité et simplifie la maintenance du code.

L'après-midi, j'ai ajouté un bouton permettant de modifier le titre d'un todo et un autre pour le supprimer. J'ai commencé à travailler sur la partie qui permet de modifier le titre ou de supprimer un todo. Mon tuteur a créé les deux APIs nécessaires, ce qui m'a permis de mettre en place les providers et services correspondants.



De plus, j'ai ajouté une fonctionnalité permettant de visualiser le nombre de tâches réalisées.

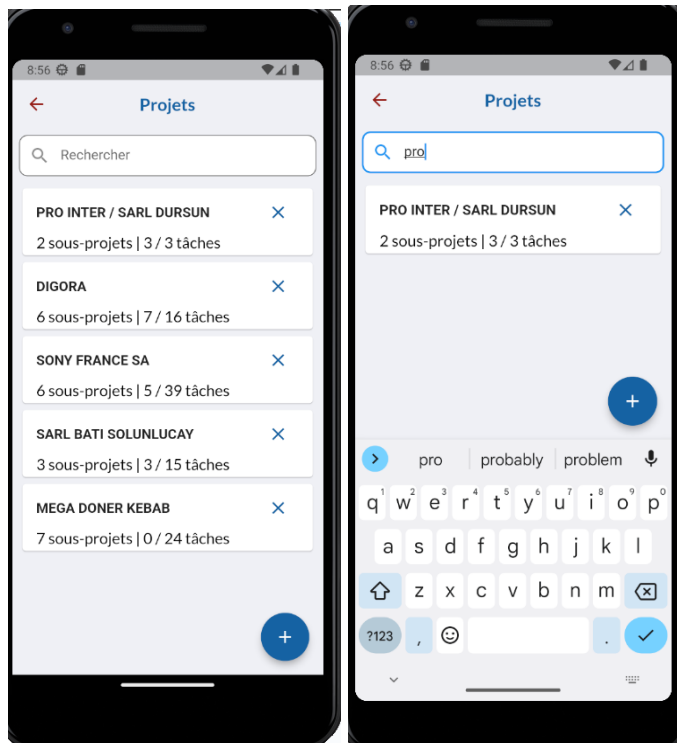
```
//fonction qui compte nb de todos coché
int getCompletedCount(Project project) {
  int completedCount = 0;

  for (var todo in project.projectTodos) {
    if (todo.isCompleted) {
      // setState(() {
      | completedCount++;
      // });
    }
  }
  return completedCount;
}

subtitle: Text(
  '${widget.project.subProjects.length} sous-projets | ${getCompletedCount(widget.project)} / ${widget.project.projectTodos.length} tâches',
```

**Jour 16 : 20/06/2023**

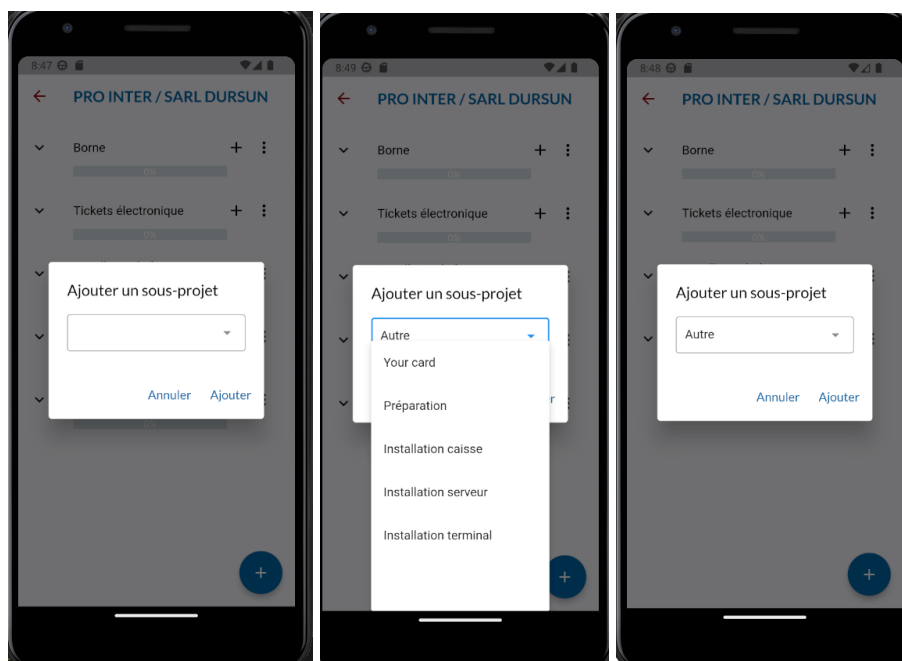
Aujourd'hui, j'ai créé une barre de recherche qui permet de rechercher des projets. Cette barre de recherche trie les projets en fonction des lettres qu'ils contiennent.



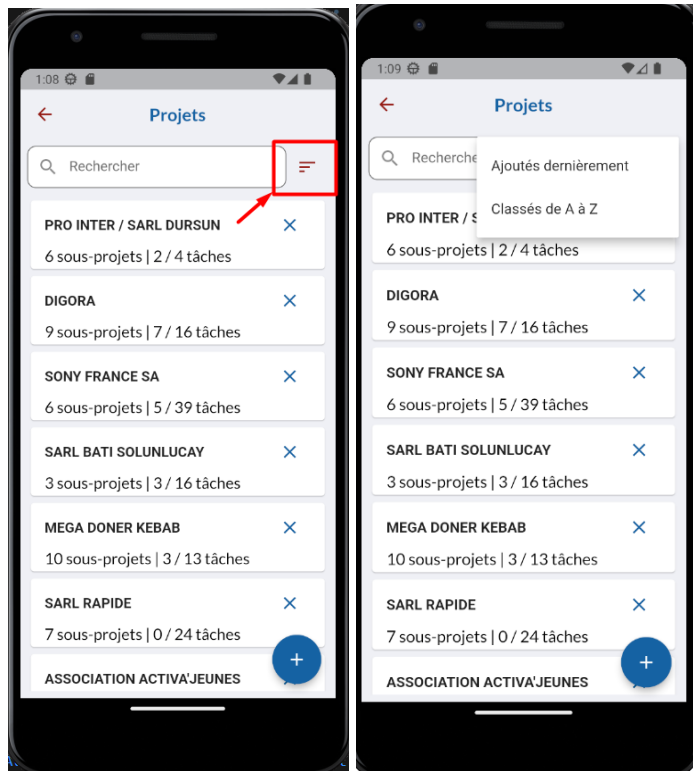
Ensuite, j'ai ajouté un bouton permettant d'ajouter un sous-projet au projet existant.

### Jour 17 : 21/06/2023

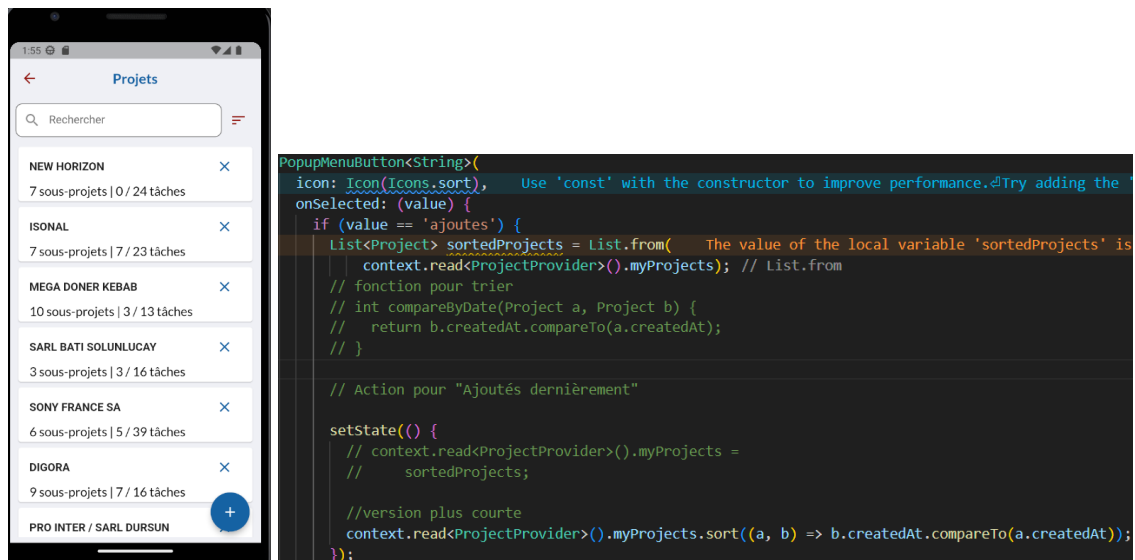
J'ai développé un bouton qui, lorsqu'il est cliqué, affiche une liste des sous-projets pouvant être ajoutés et permet de les sélectionner et de les ajouter. J'ai créé le provider et le service correspondants pour cette fonctionnalité.



Ensuite, j'ai créé un bouton qui permet de filtrer les projets en fonction des projets ajoutés récemment et de les classer par ordre alphabétique. J'ai implémenté ces boutons en recherchant des exemples et en choisissant la meilleure approche. J'ai également commencé à consulter la documentation pour mettre en place les actions nécessaires pour ces types de tri.



J'ai terminé la partie "Ajoutés dernièrement" et la partie "Classés de A à Z".





### Jour 18 : 22/06/2023

Une des fonctionnalités demandées était la possibilité de marquer des projets comme favoris. J'ai commencé à réfléchir et à chercher comment implémenter ce bouton de favoris et comment consulter les projets favoris.

J'ai créé les boutons de favoris (Icons.star) :



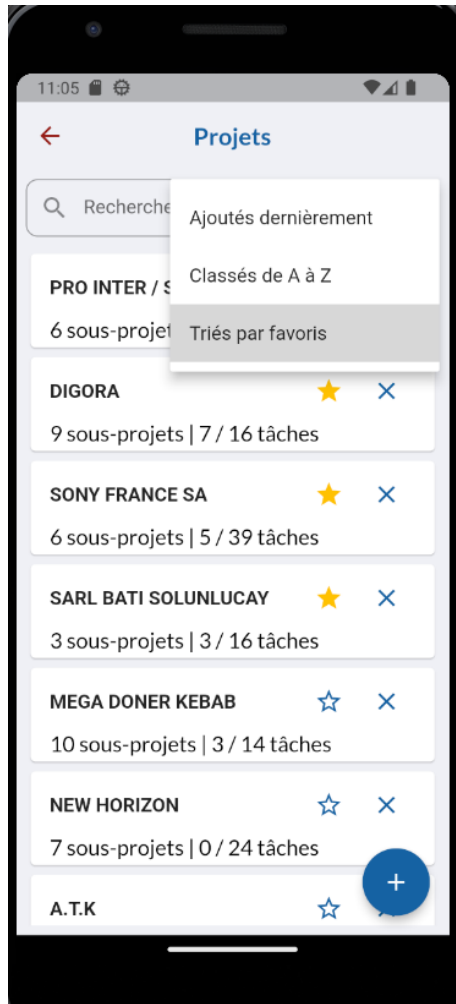
Ensuite, j'ai créé le service et le provider `_updateFavoriteStatus()` et `getFavorite()`. Cependant, j'ai rencontré quelques problèmes dans mon code.

### Jour 19 : 23/06/2023



J'ai commencé par corriger les erreurs et les problèmes liés à la fonctionnalité des favoris. J'ai également corrigé le fait que les données étaient enregistrées dans la base de données mais n'apparaissaient pas correctement.

Ensuite, j'ai ajouté une partie, un `PopupMenuItem()` appelé "Triés par favoris". Cela permettra d'afficher les projets favoris en première position, suivis des projets non favoris.



## SEMAINE 5 :

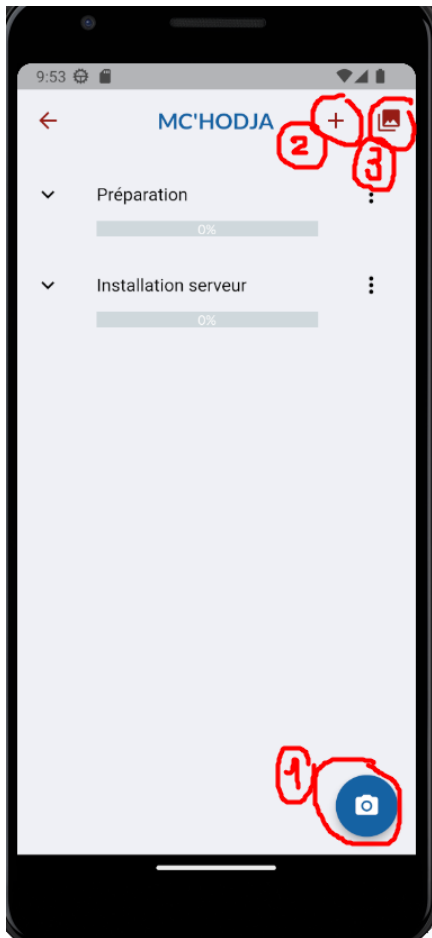
### Jour 20 : 26/06/2023

J'ai assisté à la réunion du matin comme d'habitude.

Ensuite, mon tuteur a appelé le patron et les salariés pour leur montrer ce que nous avons fait. Ils nous ont fait part de leurs demandes de modifications ou d'ajouts.

J'ai commencé par ajouter un bouton permettant d'ajouter des photos au projet.

J'ai également ajouté un autre bouton pour consulter ces images. À côté de ce bouton, nous avons ajouté un autre pour ajouter un sous-projet.

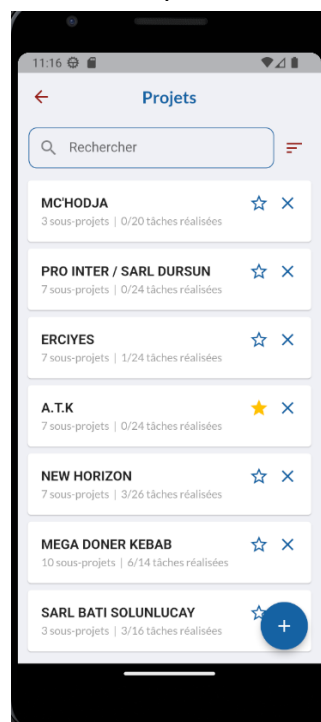


Ensuite, j'ai effectué des ajustements dans le code pour que lorsque nous accédons à la page des projets, les projets soient triés selon le critère "Ajouté dernièrement".

Avant

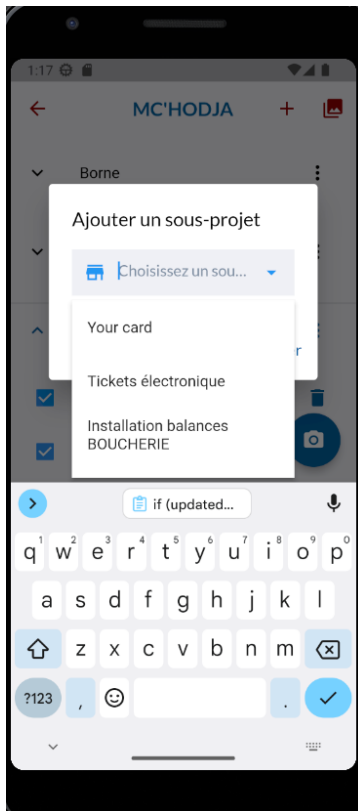


Après



J'ai remarqué un problème lors de l'ajout d'un nouveau sous-projet, les Todos ne s'ajoutent pas correctement. Nous avons donc cherché une solution pour résoudre ce problème. Mon tuteur a ajouté les API nécessaires pour ajouter une image ou une photo dans le projet, un API pour ajouter les Todos lors de l'ajout d'un sous-projet existant, et enfin un API pour ajouter un nouveau sous-projet qui n'est pas défini.

J'ai également ajouté un nouveau widget "dropdown search" pour permettre l'ajout d'un nouveau sous-projet au projet. Ce widget permet d'ajouter un sous-projet à partir de la liste des sous-projets prédéfinis, sinon il permet d'ajouter un nouveau sous-projet.

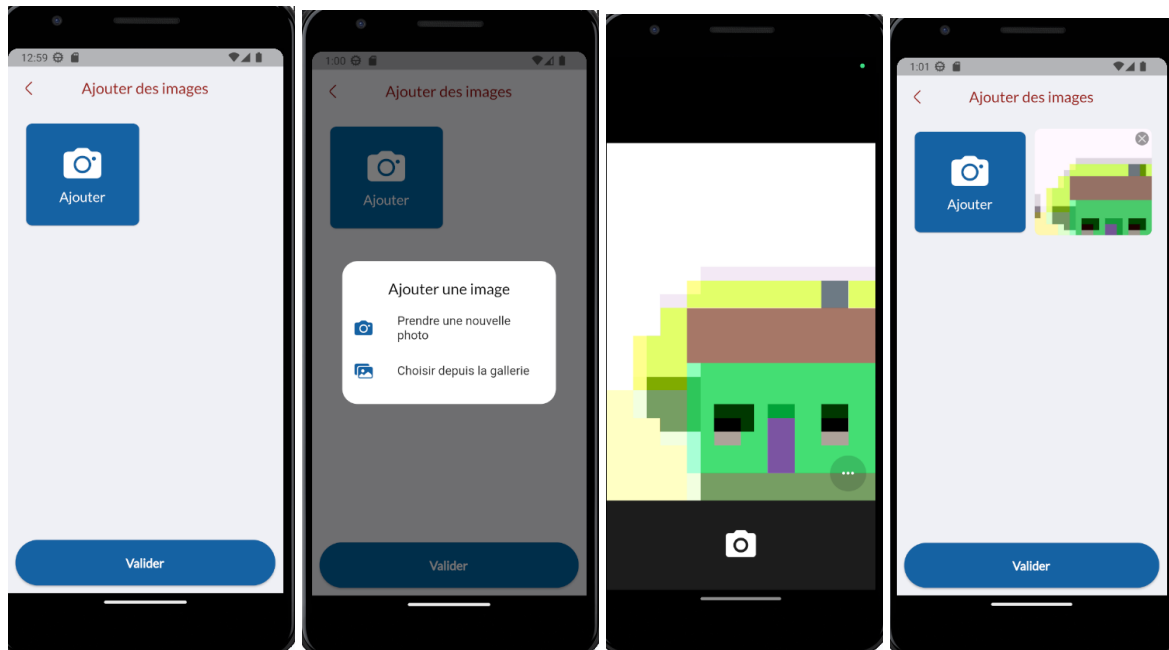


Ensuite, j'ai travaillé sur des optimisations pour la partie où l'on peut modifier le titre de la tâche.

### **Jour 21 : 27/06/2023**

Aujourd'hui, mon tuteur n'était pas disponible, ce qui m'a amené à travailler sur le projet de manière autonome. Malgré cela, j'ai pu faire des avancées significatives.

Dans la partie photothèque, j'ai ajouté une nouvelle page qui permet à l'utilisateur de prendre une photo et de l'enregistrer ensuite.



Cependant, j'ai rencontré des difficultés lorsqu'il s'est agi d'enregistrer cette photo dans la base de données. J'ai tenté différentes approches, mais je n'ai pas réussi à trouver une solution satisfaisante. Face à ce problème, j'ai décidé d'utiliser mon temps libre pour me former davantage sur Flutter.

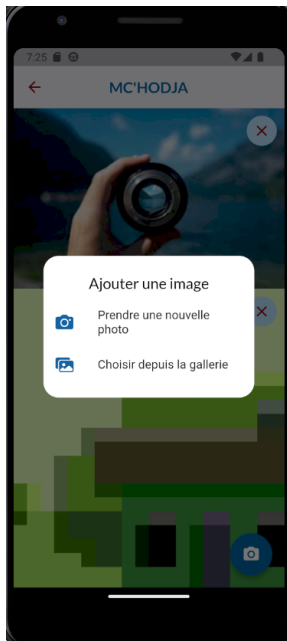
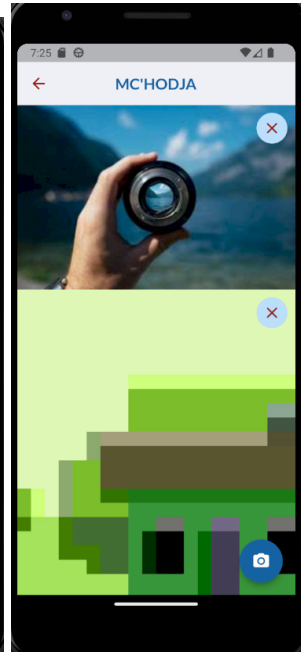
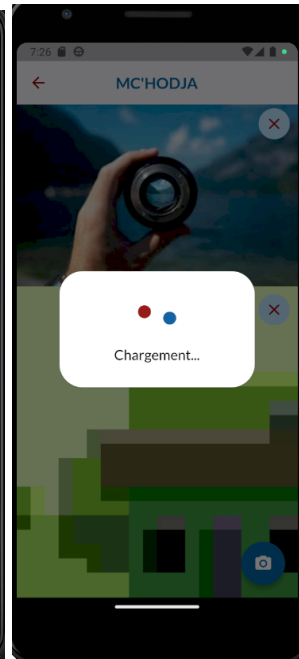
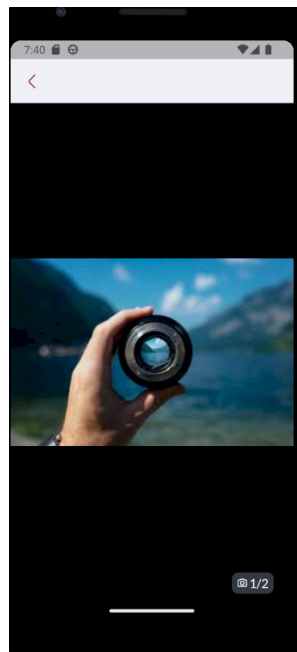
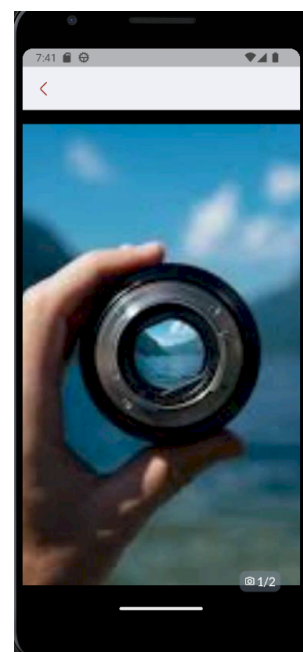
J'ai visionné des vidéos didactiques sur Flutter et j'ai consulté des documents et des sites web pertinents pour approfondir ma compréhension de la plateforme. Cette démarche m'a permis d'acquérir de nouvelles connaissances et de trouver des pistes de résolution pour les problèmes auxquels je faisais face.

### **Jour 23 : 29/06/2023**

C'était l'Aïd, je n'ai pas travaillé ce jour-là.

### **Jour 23 : 29/06/2023**

On a commencé par modifier la partie photothèque (`project_images_screen.dart`) en apportant des changements. Nous avons ajouté un bouton flottant dans la photothèque. La photothèque dispose désormais d'un bouton 'caméra' qui permet de prendre une photo ou de choisir une photo depuis sa galerie avant de l'ajouter à la photothèque. Il est également possible de supprimer une image en appuyant dessus ou de l'agrandir et de zoomer dessus à l'aide du package 'photo view gallery'.

*Ajout images**Bouton supprimer**Chargement**Bouton supprimer**Images en grand**Images en grand sans zoom*

Ensuite, nous avons amélioré la fonctionnalité de sélection multiple pour qu'elle soit unique pour chaque sous-projet. De plus, mon tuteur avait créé un widget spécifique permettant de créer une page de modification. Nous l'avons utilisé pour la partie des images en l'intégrant de la manière suivante :

```

void onDeletePicture(String name) async {
  final isConfirm = await confirmDialog(context,
    title: 'Etes-vous sûr de vouloir supprimer', confirmText: 'Oui');
  if (!isConfirm) return;

  AlertDialog.show(context); Don't use 'BuildContext's across async gap

  final result = await _projectService.deleteImage(widget.project.id, name);

  AlertDialog.hide(context); Don't use 'BuildContext's across async gap
  if (!result) return;

  setState(() {
    _images.remove(name);
  });
}

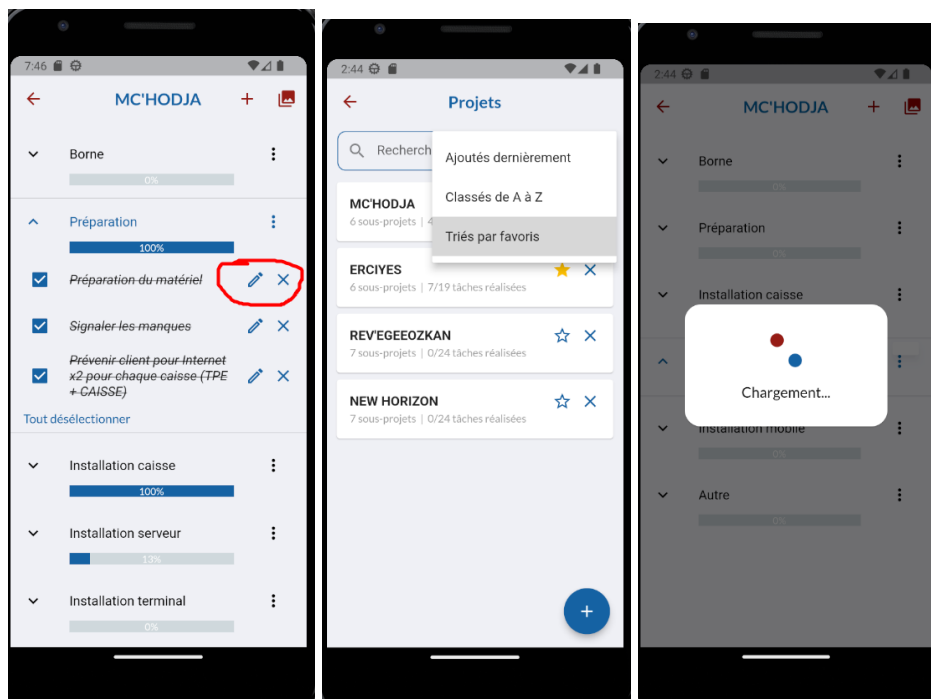
```

### Jour 24 : 29/06/2023

Ensuite, nous avons apporté des modifications au style, à l'emplacement et à la taille des boutons pour les todos afin d'améliorer leur apparence visuelle.

Par la suite, j'ai ajouté une page de chargement pour les opérations de suppression de projets, d'ajout et de suppression de sous-projets, ainsi que de suppression de To-dos. Cette page de chargement permet d'indiquer visuellement à l'utilisateur que ces opérations sont en cours de traitement.

Le reste de ma journée a été consacré à la recherche d'une solution pour gérer les favoris, car la fonctionnalité ne fonctionnait pas correctement. J'ai effectué plusieurs tests et recherches pour trouver des solutions possibles. Heureusement, mon tuteur est intervenu et m'a aidé à résoudre le problème.



## **SEMAINE 6 :**

### **Jour 25 : 03/07/2023**

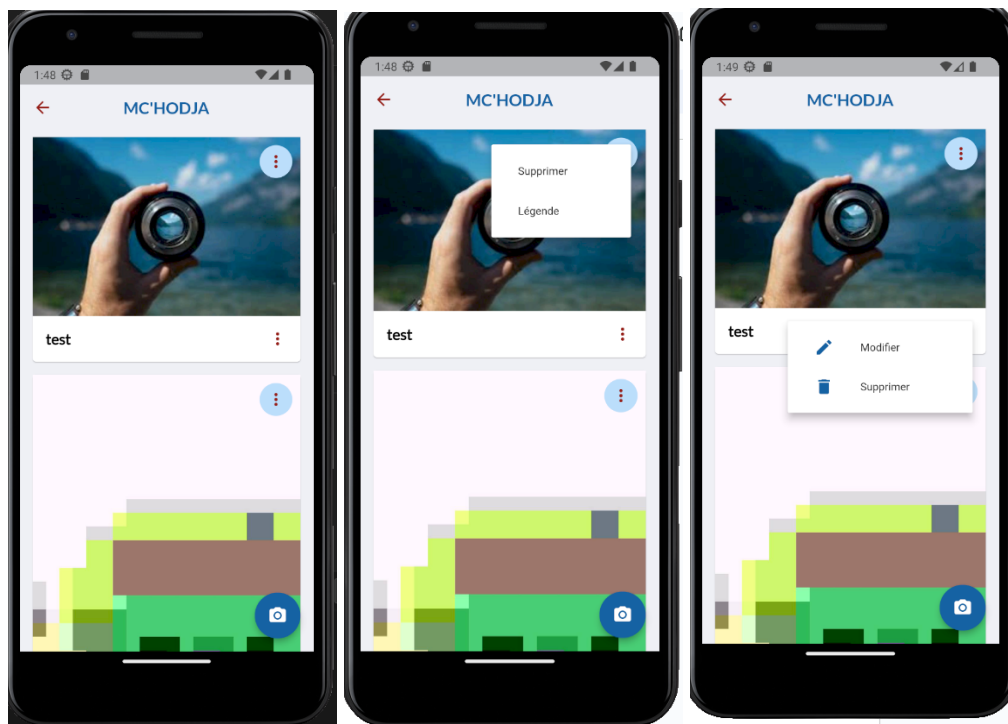
On a fait une réunion ce matin comme d'habitude. Puis, pendant que mon tuteur expliquait sur quoi on travaillait au nouveau stagiaire, de mon côté j'ai commencé à réfléchir à comment on pouvait modifier la partie "images" afin de faire une partie comme ils veulent.

### **Jour 26 : 04/07/2023**

En faisant des tests, j'ai remarqué qu'il y avait un bug quand on voulait ajouter un sous-projet. La page de chargement est chargée à l'infini. Le problème venait de rethrow. Mon tuteur m'a expliqué ce que ça servait et comment on devait l'utiliser en utilisant des exemples.

J'ai créé un modèle Pictures pour les images et un bouton option pour supprimer et ajouter une légende. J'ai ensuite créé le service pour supprimer et ajouter une légende (updatePicture).

Ensuite j'ai géré le cas où la légende doit contenir au moins une lettre, sinon on a une erreur qui s'affiche. Puis si la légende est null, on ne voit pas le bouton.



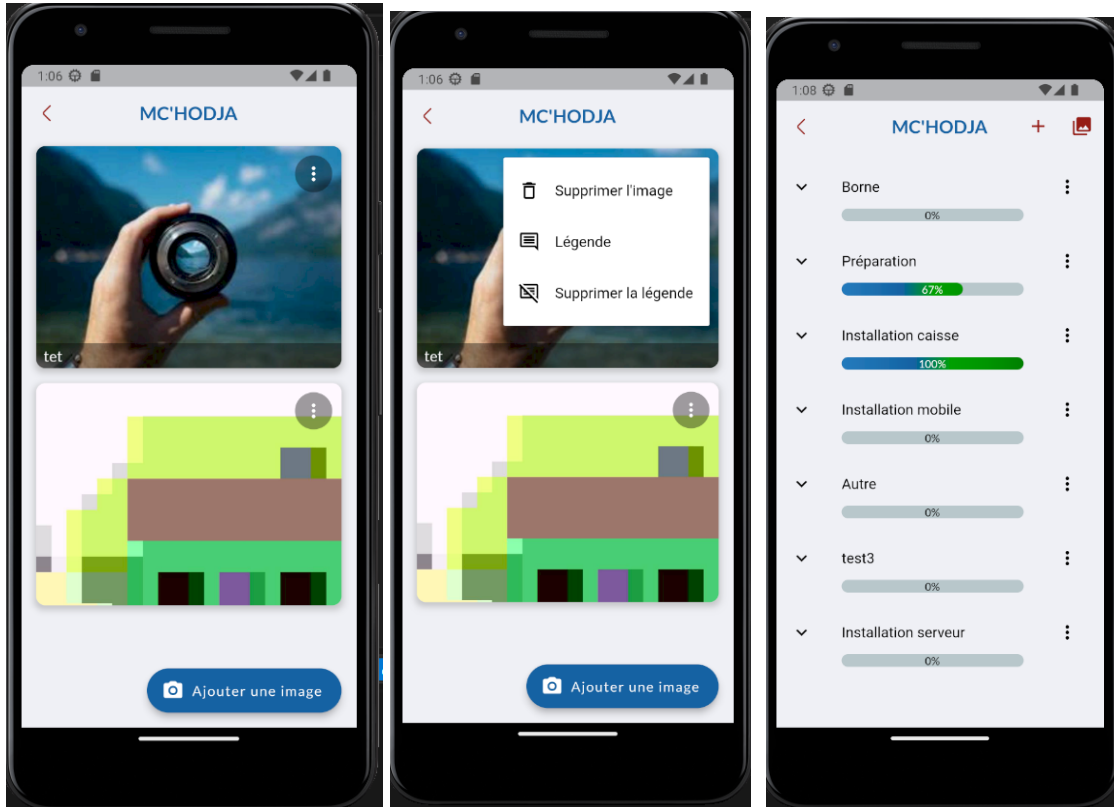
### **Jour 27 : 05/07/2023**

J'ai fait mes dernières modifications, mon tuteur a mis la dernière version de l'application sur App Store.

J'ai regardé des tutos sur youtube pour mieux comprendre pourquoi on fait certaines choses de cette façon sur Flutter.

J'ai ensuite passé le reste de la journée à changer le design pour que ça fasse beau.

Exemples :



La page images

La page sous-projets

Et enfin tout est fini.

**Jour 28 : 06/07/2023**

absente

**Jour 29 : 07/07/2023**

absente