Livrable 4 - Rapport final

Travail présenté à

Dr Hussein Al Osman

Dans le cadre du cours SEG 2505:

Introduction au génie logiciel

Soumis par:

Les membres de l'équipe 24 :

Utku Cerit, 300247866

Kheswari Gukhool, 300145273

Olivia Jia, 300215786

Alexis Laplante, 300220658

Ghait Ouled Amar Ben Cheikh, 300226766
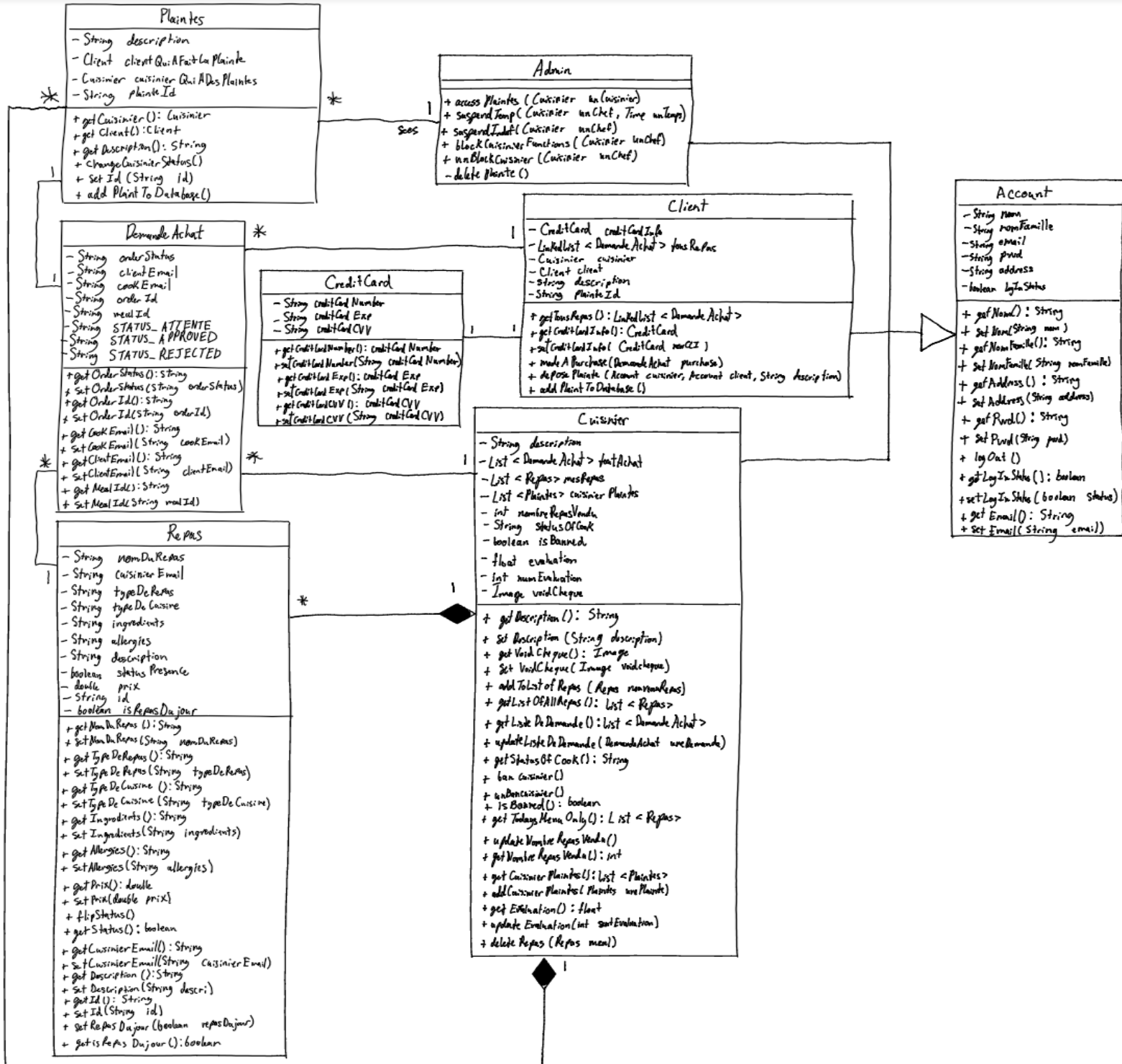
Alexandre Ringuette, 300251252

Université d'Ottawa

Mercredi le 7 décembre 2022

Introduction

Dans le cadre du cours de SEG2505, nous avons créé une application qui a pour but de connecter ceux qui vendent des repas, les cuisiniers, et ceux qui voudrait acheter ces repas, les clients, et qui a tous les fonctionnalités nécessaires pour accommoder la vente et l'achat des repas, incluant un administrateur pour gérer les situations de conflit. Sur ce sujet, dans ce rapport nous allons vous présenter le résultat final de notre application "Mealer", les contributions des membres de l'équipe et les leçons apprises.
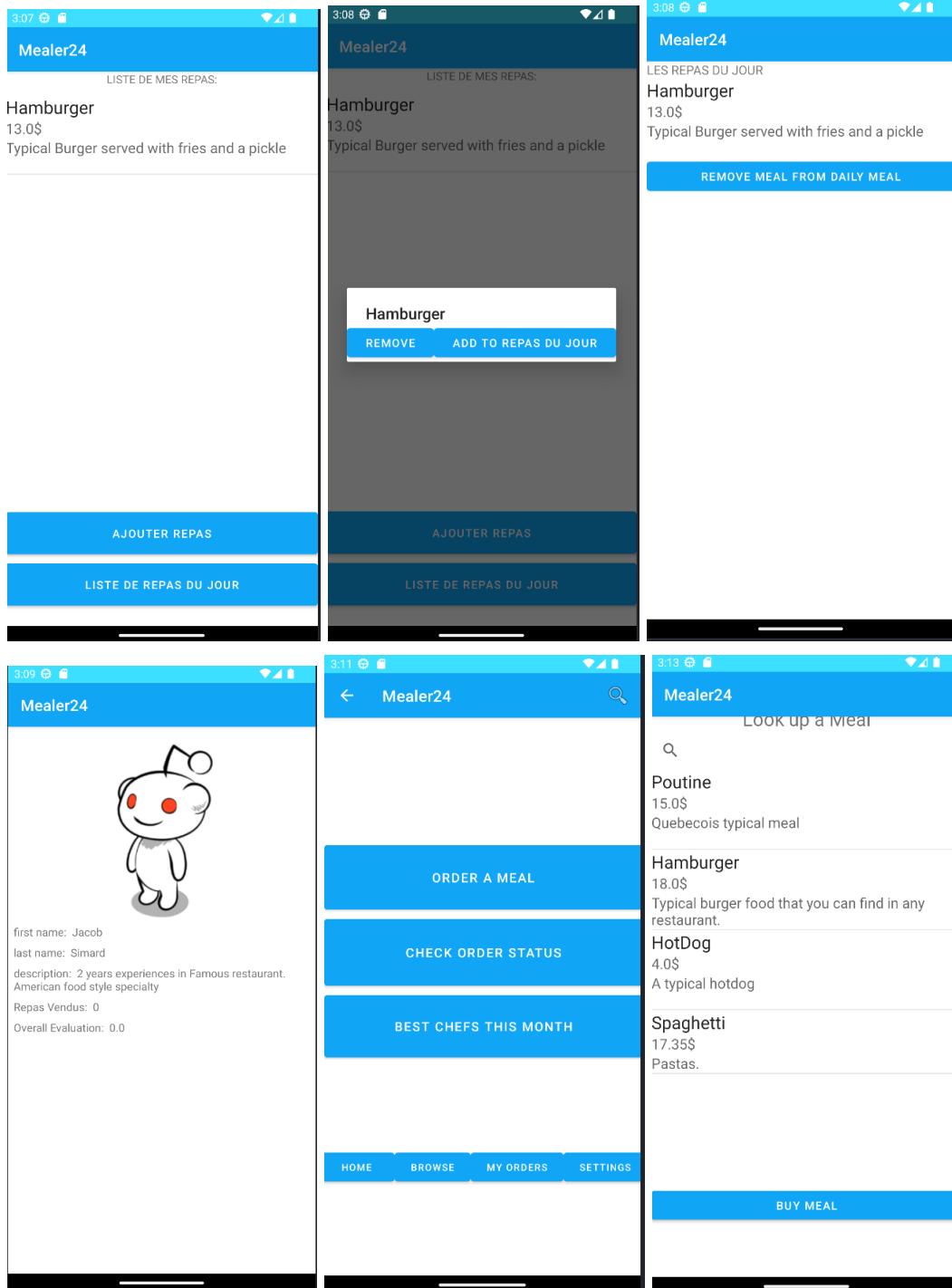
UML final

Plaintes
- String description
- Client clientQuiAFaitLaPlainte
- Cuisinier cuisinierQuiADesPlaintes
- String plainteId

+ getCuisinier(): Cuisinier
+ getClient(): Client
+ getDescription(): String
+ changeCuisinierStatus()
+ setId(String id)
+ addPlaintToDatabase()

Admin

+ access Plaintes (Cuisinier unCuisinier)
+ suspendTemp(Cuisinier unChef, Time unTemps)
+ suspendIndef(Cuisinier unChef)
+ blockCuisinierFonctions(Cuisinier unChef)
+ unBlockCuisinier(Cuisinier unChef)
- delete plainte()

Client
- CreditCard creditCardInfo
- LinkedList <DemandeAchat> tousRepas
- Cuisinier cuisinier
- Client client
- String description
- String plainteId

+ getTousRepas(): LinkedList <DemandeAchat>
+ getCreditCardInfo(): CreditCard
+ setCreditCardInfo(CreditCard newCCI)
+ madeAPurchase(DemandeAchat purchase)
+ deposePlainte(Account cuisinier, Account client, String description)
+ addPlaintToDatabase()

Account
- String nom
- String nomFamille
- String email
- String pwd
- String address
- boolean logInStatus

+ getNom(): String
+ setNom(String nom)
+ getNomFamille(): String
+ setNomFamille(String nomFamille)
+ getAddress(): String
+ setAddress(String address)
+ getPwd(): String
+ setPwd(String pwd)
+ logOut()
+ getLogInStatus(): boolean
+ setLogInStatus(boolean status)
+ getEmail(): String
+ setEmail(String email)

Demande Achat
- String orderStatus
- String clientEmail
- String cookEmail
- String orderId
- String mealId
- String STATUS_ATTENTE
- String STATUS_APPROVED
- String STATUS_REJECTED

+ getOrderStatus(): String
+ setOrderStatus(String orderStatus)
+ getOrderId(): String
+ setOrderId(String orderId)
+ getCookEmail(): String
+ setCookEmail(String cookEmail)
+ getClientEmail(): String
+ setClientEmail(String clientEmail)
+ getMealId(): String
+ setMealId(String mealId)

CreditCard
- String creditCardNumber
- String creditCardExp
- String creditCardCVV

+ getCreditCardNumber(): CreditCardNumber
+ setCreditCardNumber(String creditCardNumber)
+ getCreditCardExp(): CreditCardExp
+ setCreditCardExp(String creditCardExp)
+ getCreditCardCVV(): CreditCardCVV
+ setCreditCardCVV(String creditCardCVV)

Cuisinier
- String description
- List <DemandeAchat> toutAchat
- List <Repas> mesRepas
- List <Plaintes> cuisinierPlaintes
- int nombreRepasVendu
- String statusOfCook
- boolean isBanned
- float evaluation
- int numEvaluation
- Image voidCheque

+ getDescription(): String
+ setDescription(String description)
+ getVoidCheque(): Image
+ setVoidCheque(Image voidCheque)
+ addToListOfRepas(Repas nouveauRepas)
+ getListOfAllRepas(): List <Repas>
+ getListeDeDemande(): List <DemandeAchat>
+ updateListeDeDemande(DemandeAchat uneDemande)
+ getStatusOfCook(): String
+ banCuisinier()
+ unBanCuisinier()
+ isBanned(): boolean
+ getTodayMenuOnly(): List <Repas>
+ updateNombreRepasVendu()
+ getNombreRepasVendu(): int
+ getCuisinierPlaintes(): List <Plaintes>
+ addCuisinierPlaintes(Plaintes unePlainte)
+ getEvaluation(): float
+ updateEvaluation(int sonEvaluation)
+ deleteRepas(Repas meal)

Repas
- String nomDuRepas
- String cuisinierEmail
- String typeDeRepas
- String typeDeCuisine
- String ingredients
- String allergies
- String description
- boolean statusPresence
- double prix
- String id
- boolean isRepasDuJour

+ getNomDuRepas(): String
+ setNomDuRepas(String nomDuRepas)
+ getTypeDeRepas(): String
+ setTypeDeRepas(String typeDeRepas)
+ getTypeDeCuisine(): String
+ setTypeDeCuisine(String typeDeCuisine)
+ getIngredients(): String
+ setIngredients(String ingredients)
+ getAllergies(): String
+ setAllergies(String allergies)
+ getPrix(): double
+ setPrix(double prix)
+ flipStatus()
+ getStatus(): boolean
+ getCuisinierEmail(): String
+ setCuisinierEmail(String cuisinierEmail)
+ getDescription(): String
+ setDescription(String descri)
+ getId(): String
+ setId(String id)
+ setRepasDuJour(boolean repasDuJour)
+ getIsRepasDuJour(): boolean

Tableau de contributions des membres

| Membre d'équipe	Nombre total de "commits"	Description des tâches			
		Livrable 1	Livrable 2	Livrable 3	Livrable 4
Utku Cerit	20	-Designed and created the Homescreen classes.	- Tried to help in the implementation of display of the plaintes and the suspension of admin	-Helped in the deletion of the repas from the list -Implemented test cases	-Implemented a method that allows the client to leave a plainte about the cuisinier - Implemented test cases
Kheswari Gukhool	23	-Created and implemented the UML + classes such as: Cuisinier, Repas, Client, … -Few comments	-Implementation of display of the plaintes Admin can see -Helped a bit in suspension of Admin	-Worked on the classes to help add a new repas -Helped in taking in user input from xml	-Tried implementing Search View and display results of Client
Olivia Jia	19	- Task division - Comments in classes (excluding activities) - Added a few "husks" of methods in Admin and Cuisinier - UML	- Task division - UML update	- Task division - Comments for test cases - Test case for get/set methods for the cooks void cheque - UML update	- Task division - Comments for most of the app - Rapport final (minus "description des tâches" of other team members in this table and "captures d'écran") - UML update
Alexis Laplante	11	- XML creation of the sign up page - Dumb proofing of all the sign up entries	- Tried to implementation of Test cases	-XML creation of add/remove repas and repas of the day - Dumb proofing of all entries for the add repas	-Helped on the implementation of the search View and display results of Client -Add the "captures d'écran" for the Rapport Final
Ghait Ouled Amar Ben Cheikh	23	-Implemented the login - logout functionality of users	-Added support for banning cooks permanently/ temporarily.	-Tried to help everyone with any sort of issues they were facing.	-Cooks can see their profile which showed their final evaluation -Client see the status of their orders. The orders also included information about the cook and the meal itself.
Alexandre Ringuette	37	- Implemented the creation of accounts & storing user data in Firebase	- Helped with adding and retrieving complaints from Firebase	- Implemented adding meal to the cooks menu in Firebase - Cook can add meal in menu of the day. - Added test cases	- Clients could place orders. - Cook able to see orders placed for them.

Captures d'écran
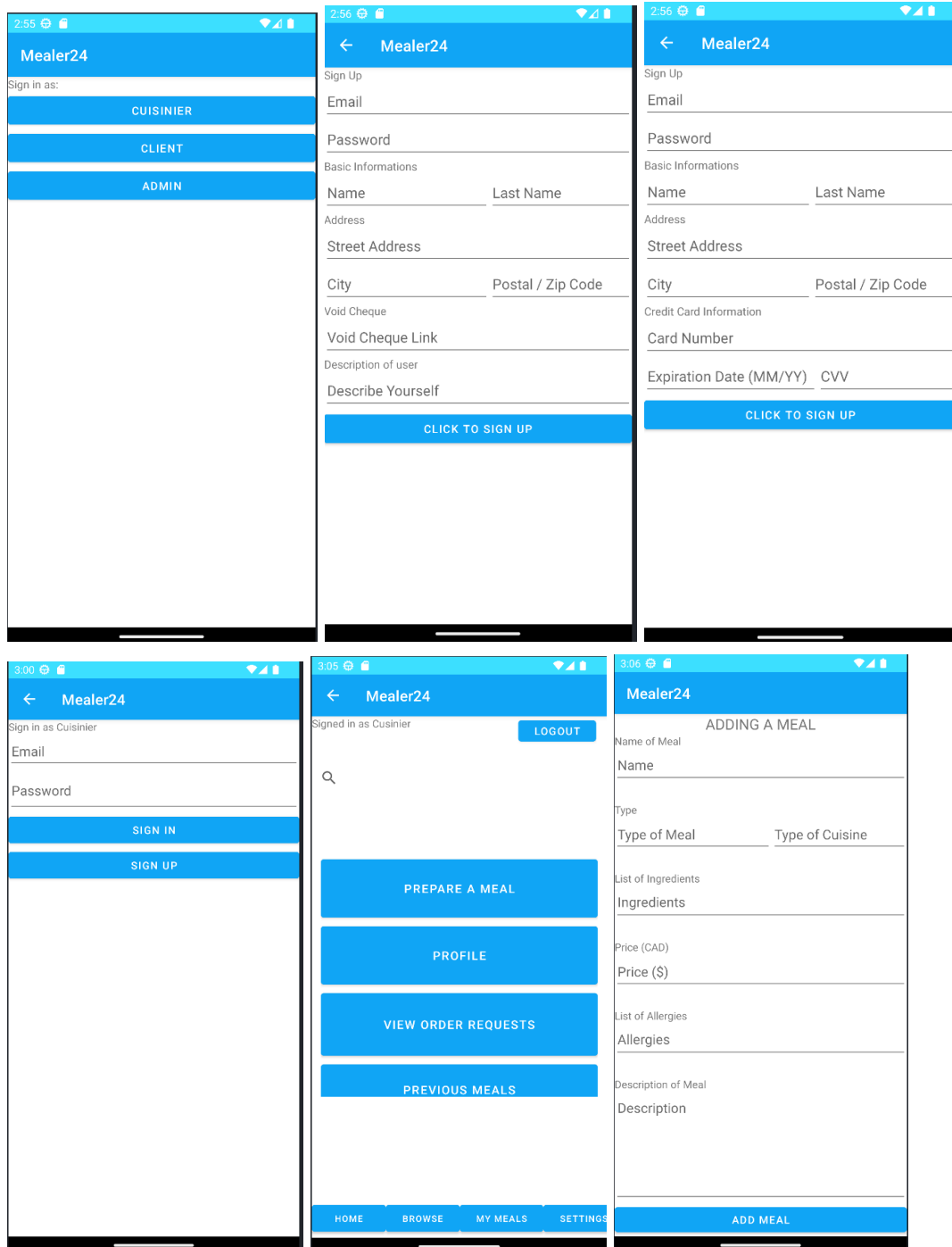
1   Main sign in/up page
2   Cook sign up page
3   Client sign up page
4   General sign in page
5   Cook homescreen page
6   Add meal page

Screen 1

LISTE DE MES REPAS:

Hamburger
13.0$
Typical Burger served with fries and a pickle

AJOUTER REPAS

LISTE DE REPAS DU JOUR

Screen 2

3:08 Mealer24

LISTE DE MES REPAS:

Hamburger
13.0$
Typical Burger served with fries and a pickle

Hamburger
REMOVE   ADD TO REPAS DU JOUR

AJOUTER REPAS

LISTE DE REPAS DU JOUR

Screen 3

3:08 Mealer24

LES REPAS DU JOUR

Hamburger
13.0$
Typical Burger served with fries and a pickle

REMOVE MEAL FROM DAILY MEAL

Screen 4

3:09 Mealer24

first name: Jacob
last name: Simard
description: 2 years experiences in Famous restaurant. American food style specialty
Repas Vendus: 0
Overall Evaluation: 0.0

Screen 5

3:11 Mealer24

ORDER A MEAL

CHECK ORDER STATUS

BEST CHEFS THIS MONTH

HOME   BROWSE   MY ORDERS   SETTINGS

Screen 6

3:13 Mealer24

Look up a Meal

Poutine
15.0$
Quebecois typical meal

Hamburger
18.0$
Typical burger food that you can find in any restaurant.

HotDog
4.0$
A typical hotdog

Spaghetti
17.35$
Pastas.

BUY MEAL

[1] Cook list of repas
[2] Cook remove/add to repas du jour
[3] Repas du jour list
[4] Cook Profile
[5] Client home screen page
[6] Meal order/search page
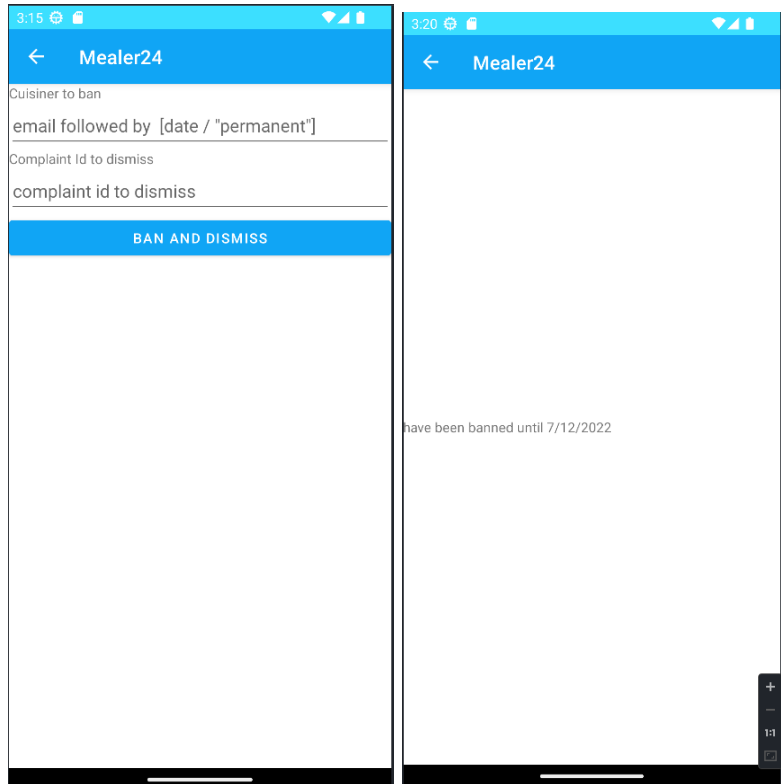
[1] Admin band and dismiss complaint page
[2] Banned cooked homescreen page

Lessons learned

As the project went on and as our team became more and more busy with other classes and such, it was increasingly more difficult to completely stick to our rigid plan. As this was the case, we had to be pretty flexible with helping and supporting each other or exchanging a few tasks here and there which is one of the first lessons. On that subject, to successfully swap tasks or ask for help, we became very efficient in communicating with one another with the help of Discord. We found that for small and quicker subjects it was more efficient to use the platform's text chat to signal any issues. On the other hand, for things like the implementation with code or to make sure there is no ambiguity, it was far simpler to use Discord's voice chat. This feature was especially helpful when we were troubleshooting, another subject upon which we learned more, with one or two other teammates that were either assigned the same task or have more expertise in the part with which we had trouble. Because our time was limited, the only way to get the deliverables done within the allotted time period was to break down tasks. At first it was a little hazy how to split the work so that it was as fair as possible, but with the following deliverables and time, we grasp this topic. Speaking of time, with the aforementioned time restraints coming from both our personal lives and this course, time management was a big lesson on which to gain further knowledge. Though time was a very finite resource, we had plenty of online and in person ressource. Nevertheless, even with an infinite amount of ressource, if you did not know how to use it, it would have been wasted which is why this was an additional lesson we had to latch onto. To complete this project, importantly, we had to learn how to use what we were taught in class and implement thoses concepts into our application, in turn learning by doing the implementation of this software. Finally, none of it would have been possible without learning how to work with each other as a team.

Conclusion

Pour conclure, nous avons conçu et implémenté une application "Mealer" qui offre des services alimentaires et relie les vendeurs et une clientèle appropriée en fonction des repas de ces derniers.