The Mealer App

SEG2105 - Introduction to Software Engineering

Fall 2022

University of Ottawa

Nikolah Danchuk-Lauzon 300233204
Meekyle Mondal 300246544
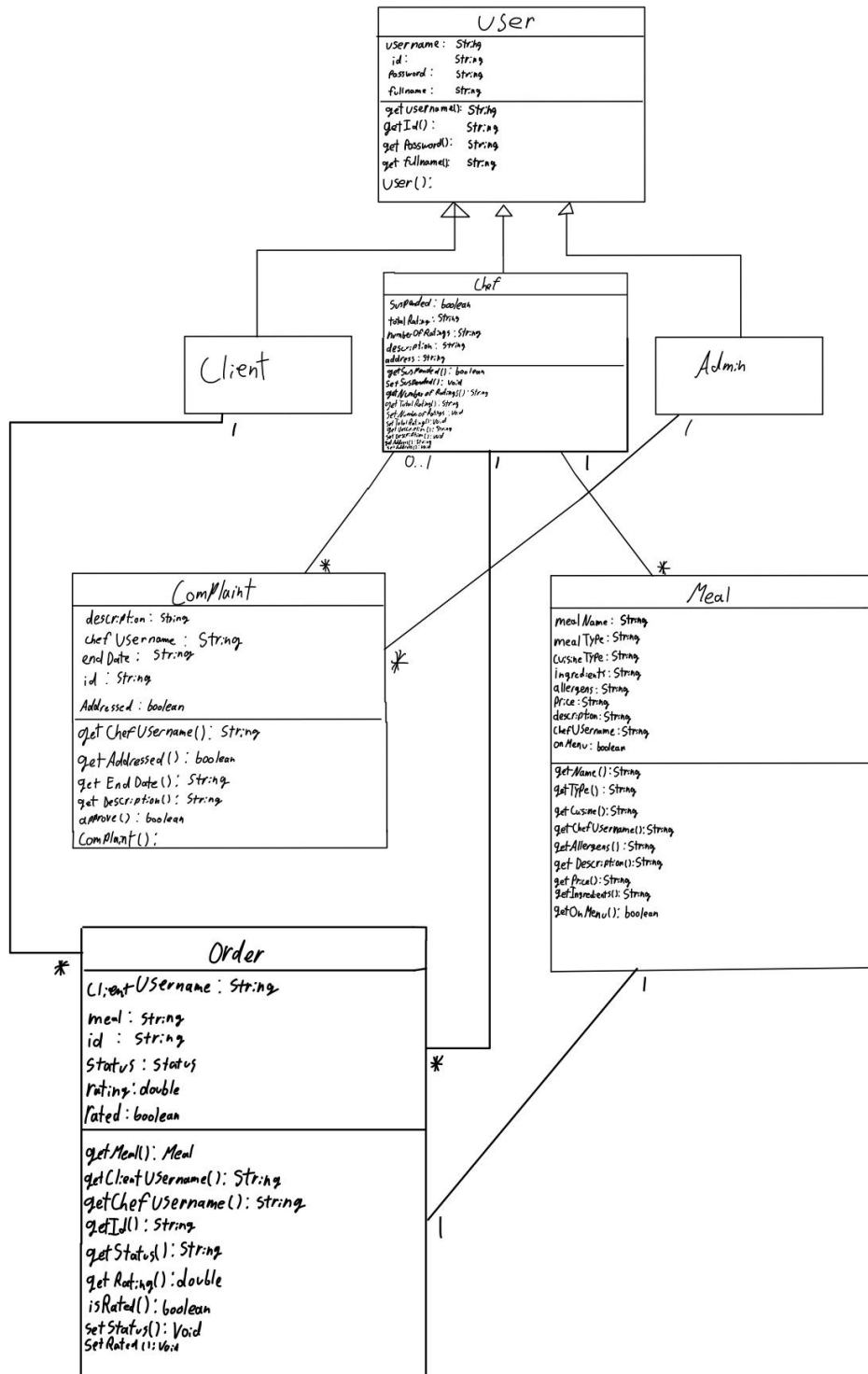**Luke DuSautoy 300239507**
Nodshley Marcelin 300229763

Introduction:

We built the app with the sole purpose of doxing users. The SEG2105 Mealer project is a project about creating a meal sharing application where local cooks can sell meals to clients from their home. The application was built on the foundation of 3 user types: Cooks, Clients and the Administrator. The Cook user type is a user who is able to make meals at their home and sell them to clients. The Client user type is a user who buys meals from the Cooks. They order meals through the Mealer application and they pick up said meal from the Cook's home. The Administrator user type is a user who manages the Clients' complaints about Cooks, and has the ability to suspend the Cook if necessary. The application was developed over the course of the Fall 2022 semester, and was finally finished on December 7th, 2022.

Lessons Learned:

The development of the project allowed us to experiment and utilize many new development tactics and technologies. The first main lesson learned was learning how to develop an application with a team. The Mealer project was the first large scale application based project that any of us had worked on. This meant that communicating and developing an application over a long period of time was a new challenge to face for us all. During the first few deliverables, we knew that assigning a work role to everyone would be crucial, as this allowed the individuals of the group to become "experts" in said role. Over the course of the semester, we utilized Discord as our main method of communication as our group was familiar and comfortable with the application. This allowed for quick and easy development updates/questions to be posted, allowing team members to push/pull accordingly without the fear of merge conflicts. The final lesson that our group learned was that utilizing improper coding practices eventually causes easily avoidable problems. We saw this when implementing the deliverable 4 features. Lots of time was wasted on re-coding some parts of already built code, which all could have been avoided if we had utilized proper coding practices.

Updated UML:

User

| username : String |
| id : String |
| Password : String |
| fullname : String |

get Username(): String
getId() : String
get Password(): String
get fullname(): String
User():

Client

Chef

| Suspended : boolean |
| total Rating: String |
| Number Of Ratings : String |
| description : String |
| address : String |

getSuspended(): boolean
Set Suspended(): void
get Number of Ratings(): String
get Total Rating(): String
Set Number of Ratings : void
Set Total Rating(): Void
Set Description(): void
get Description(): String
Set Address(): String
Set Address(): void

Admin

Complaint

| description : String |
| chef Username : String |
| end Date : String |
| id : String |
| Addressed : boolean |

get ChefUsername(): String
get Addressed() : boolean
get End Date() : String
get Description(): String
approve() : boolean
Complaint():

Meal

| meal Name : String |
| meal Type : String |
| Cuisine Type : String |
| ingredient : String |
| allergens : String |
| Price : String |
| description : String |
| ChefUsername : String |
| on Menu : boolean |

get Name(): String
get Type(): String
get Cuisine(): String
get ChefUsername(): String
get Allergens() : String
get Description(): String
get Price(): String
get Ingredients(): String
get On Menu(): boolean

Order

| Client Username : String |
| meal : String |
| id : String |
| Status : Status |
| rating : double |
| rated : boolean |

get Meal(): Meal
get Client Username(): String
get Chef Username(): String
get Id(): String
get Status(): String
get Rating(): double
isRated(): boolean
Set Status(): Void
Set Rated(): Void

Table of Contributions based on a scale of 10:
1 (Helped a Little) - 10 (Did everything)

<u>Table 1:</u> Deliverable 1

Tasks	Contributors			
	Meekyle	**Nikolah**	**Luke**	**Nodshley**
Chef Functionality	4	2	4	0
Client Functionality	4	2	4	0
Admin Functionality	4	2	4	0
User Interface	0	0	0	10
UML	1	8	1	0
Firebase Setup	0	0	10	0

- **Luke setup the Github**
- **Commits: Everyone**
- **The UML was created by Nikolah with help from Luke and Meekyle**
- **Submission was done by Nikolah and Meekyle**
- **Create client and cook done by Meekyle, Luke and Nikolah**
- **The welcome screen was done by Meekyle and Luke**
- **The log off functionality was done by Meekyle and Luke**
- **Valid inputs was done by Meekyle, Luke and Nikolah**
- **Firebase was setup by Luke**

Table 2: Deliverable 2

Tasks	Contributors			
	Meekyle	Nikolah	Luke	Nodshley
Chef Functionality	3.5	3	3.5	0
Client Functionality	3.5	3	3.5	0
Admin Functionality	5	0	5	0
User Interface	0	0	0	10
UML	1	8	1	0
Firebase Additions	0	0	10	0

- **Nikolah did the UML diagram**
- **Unit tests weren't made for this deliverable, but we did provide the TA with all possible cases of chefs being banned (complaint hasn't been addressed, banned but end date has passed, banned and end date still hasn't come and indefinite ban) done by Nikolah and Meekyle**
- **Database set up and addition was done by Luke**
- **Account creation and sign up was a joint effort between Luke, Meekyle and Nikolah**
- **Complaint functionality set up Meekyle and Luke**
- **Administrator being able to view complaints set up by Luke and Meekyle**
- **Message informing cook they've been suspended was done by Meekyle**
- **Submission was done by Nikolah and Meekyle**

Table 3: Deliverable 3

Tasks	Contributors			
	Meekyle	**Nikolah**	**Luke**	**Nodshley**
Chef Functionality	4	2	4	0
Client Functionality	4	2	4	0
Admin Functionality	4	2	4	0
User Interface	0	2	0	8
UML	1	8	1	0
Firebase Additions	4.5	1	4.5	0
Junit Test Cases	0	10	0	0

- **UML was mostly done by Nikolah with help from Meekyle and Luke**
- **The submission was done by Meekyle and Nikolah**
- **The Local tests were done by Nikolah**
- **Add meal functionality was mostly done by Meekyle with some help from Nikolah**
- **Deleting meal functionality was mostly done by Meekyle and Luke, with some help from Nikolah**
- **Removing meal functionality was done by Meekyle and Luke**
- **Suspended cook functionality was done by Meekyle and Luke**
- **Valid error messages was done by Meekyle and Luke**

Table 4: Deliverable 4

Tasks	Contributors			
	Meekyle	Nikolah	Luke	Nodshley
Chef Functionality	5	0	5	0
Client Functionality	5	0	5	0
Admin Functionality	5	0	5	0
User Interface	0	0	2	8
UML	0	10	0	0
Firebase Addition	5	0	5	0
Report	2	8	0	0
Junit Test Cases	0	10	0	0

- UML was done by Nikolah
- Submitted by Nikolah and Meekyle
- Unit test cases were done by Nikolah
- Final report was mostly done by Nikolah with some help from Meekyle
- Search functionality was implemented by Meekyle
- Client being able to view the cook's and meal's information was joint effort by Meekyle and Luke
- Client submitting purchase request was done by Luke
- Client viewing the status of their orders was done by Luke
- Chefs being able to approve or deny orders was done by Meekyle
- Client rating was done by Meekyle and Luke
- Client complaints was done by Luke
- Cook being able to view their rating and profile was done by Luke
- Field validation was done by Meekyle and Luke at various points

Mealer Screenshots:
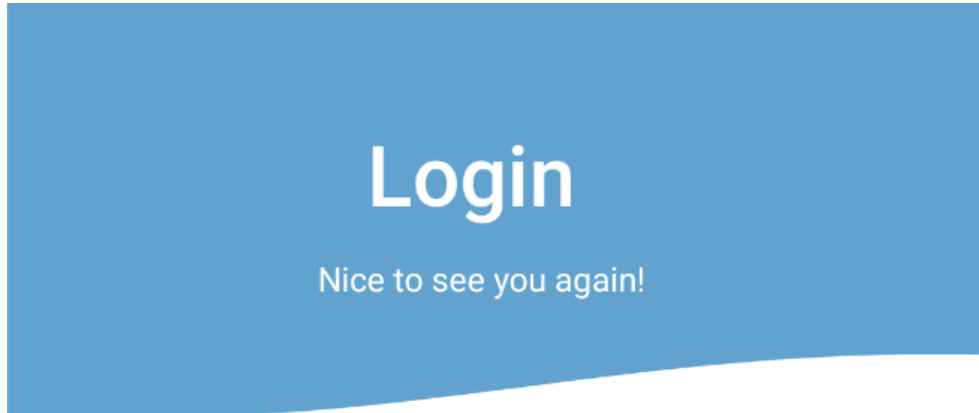
123

What would you like to do?

ORDER MEAL

VIEW ORDERS

Meal

🔍

Name ▾

Pasta

Orders

eggrolls

approved

Chef

Let's get Cookin!

MENU

ORDERS

PROFILE

Menu

Pasta

❌

eggrolls

⭐

ADD

Orders

eggrolls

Profile

Username: 321   Password: 321
Suspended: false   Address: null
Description: testing   Name: egsages
Number Of Orders: 18   Rating: 1

Welcome

Click below to attach an image of a blank cheque

Username

Full Name

Password

Confirm Password

Enter Description

Enter Address

REGISTER

Already Have an Account?
Sign In

Welcome

Fill the Form Below

Full Name

Username

Password

Confirm Password

Enter Credit Card Info

Enter Address

REGISTER

Already Have an Account?

Sign In

Order Meal

Meal Name:Pasta

Allergens:None

Price: 19.99

Chef Name: 1

Pick Up Location:1

Meal Description:Pasta

Chef Description:1

Ingredients:Pasta

Meal Type: Side

Cuisine: Italian

Rating:0

ORDER