FACULTY OF COMPUTING
UTM Johor Bahru

Semester 2 2023/2024
Project: Phase 2

SECJ1023 PROGRAMMING TECHNIQUE 2
LECTURER: DR. LIZAWATI BINTI MI YUSOF

Group Member:

Name	**Matric No.**
Angie Wong Siaw Thing	A23CS0048
Tan Jia Ying	A23CS0274
Tan Zheng Yu	A23CS5017

Table of Contents

1.0 Introduction 1

2.0 Objectives 1

3.0 Background of the Project 2

4.0 Flow Chart 3

5.0 UML Class Diagram 9

6.0 OOP concepts 10

6.1 Encapsulation 10

6.2 Composition 11

6.3 Aggregation 12

6.4 Inheritance 13

6.5 Polymorphism 14

6.6 Array of objects 15

7.0 Conclusion 16

1.0 Introduction

The restaurant recommendation system is a software application designed to suggest restaurants to users based on their preferences, incorporating a blend of user feedback. The application collects data using a rating system and is developed using the C++ programming language.

The main aim of this application is to propose food options tailored to users' tastes and habits. Additionally, the application provides convenience for users to order food from their favourite restaurants.

2.0 Objectives

1. Propose food options tailored to users' tastes and habits.
2. Allow users to evaluate restaurants.
3. Arrange restaurant details efficiently.
4. Deliver precise restaurant details to users.
5. Offer personalized dining suggestions.
6. Enhance user contentment and involvement.

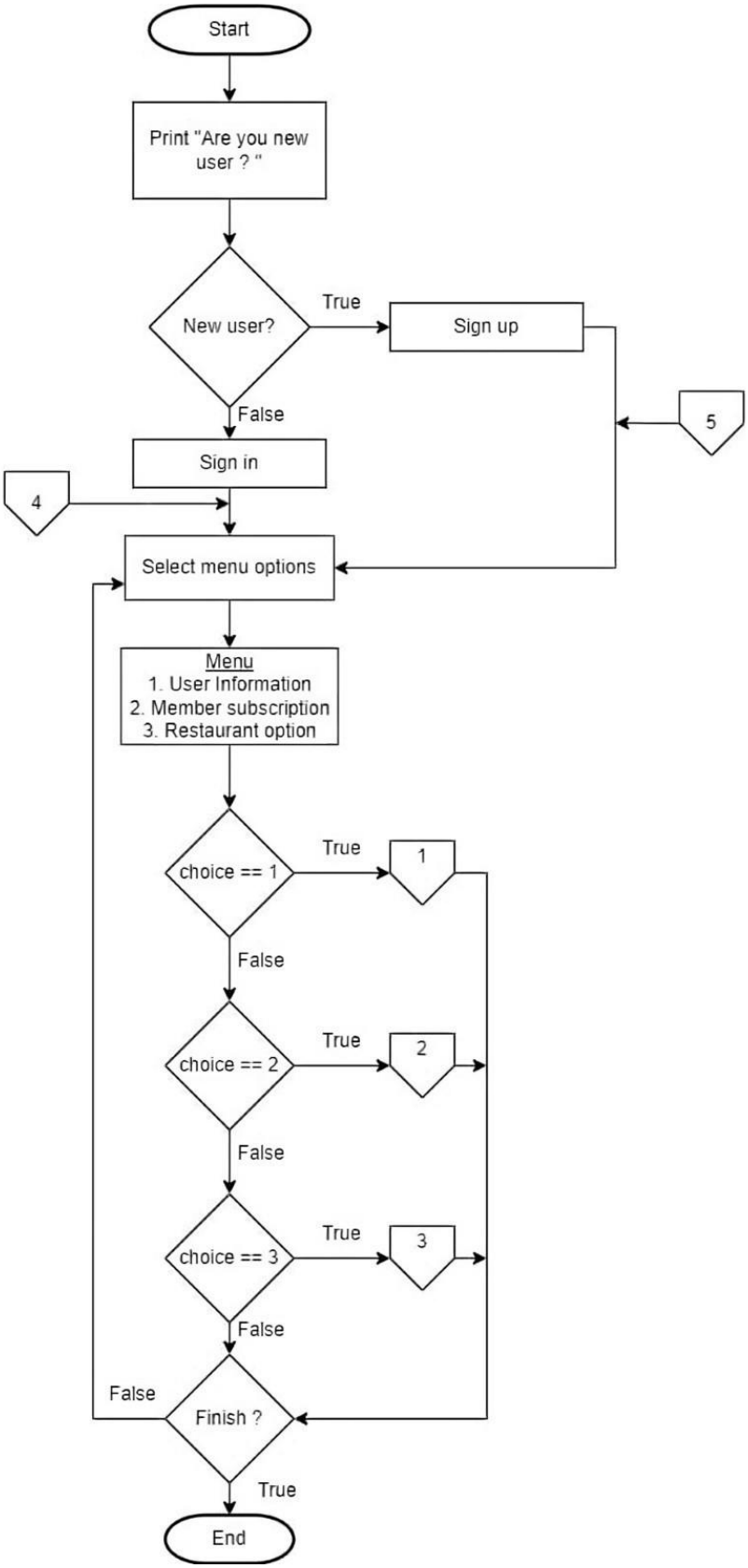
3.0 Background of the Project

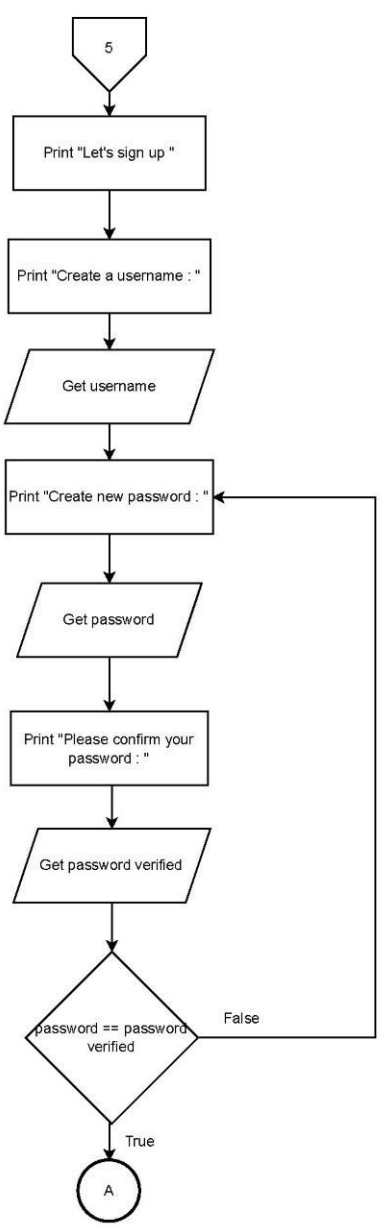
In today's fast-paced world, finding a good restaurant that matches one's culinary preferences and geographical convenience can be a challenge. While existing platforms like Google Maps offer ratings, they often fall short in providing comprehensive information such as detailed menus. On the other hand, services like Foodpanda focus on food delivery, yet they frequently lack essential information about the restaurant's physical address and other relevant details, creating a gap in the user experience.

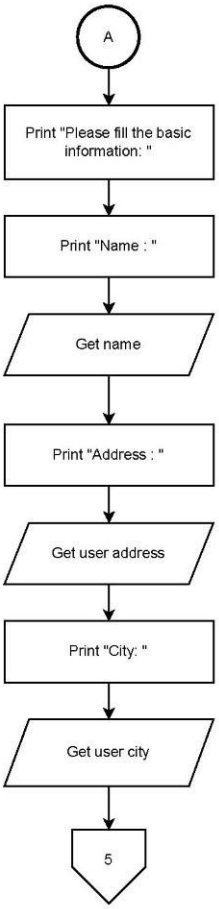
To address this gap, our project aims to develop a robust restaurant recommendation system that provides users with an integrated solution. This system will combine detailed restaurant information, including menus, user ratings, and precise locations. By leveraging data from multiple sources and employing advanced algorithms, we intend to offer personalized recommendations that cater to individual tastes and preferences.

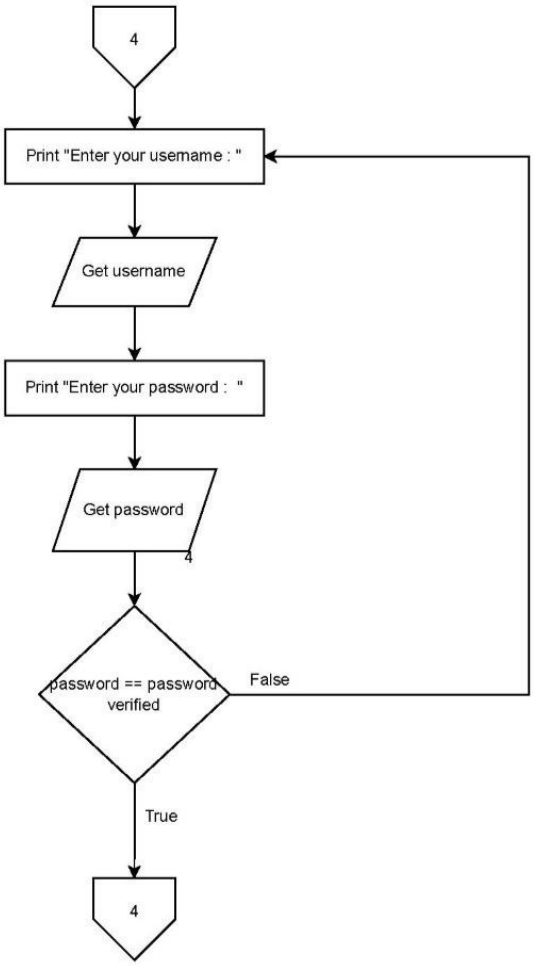
Ultimately, our goal is to simplify the decision-making process for users when selecting a dining venue. By offering a comprehensive view of available options, we hope to enhance the dining experience, making it easier for people to discover and enjoy new culinary delights in their vicinity.

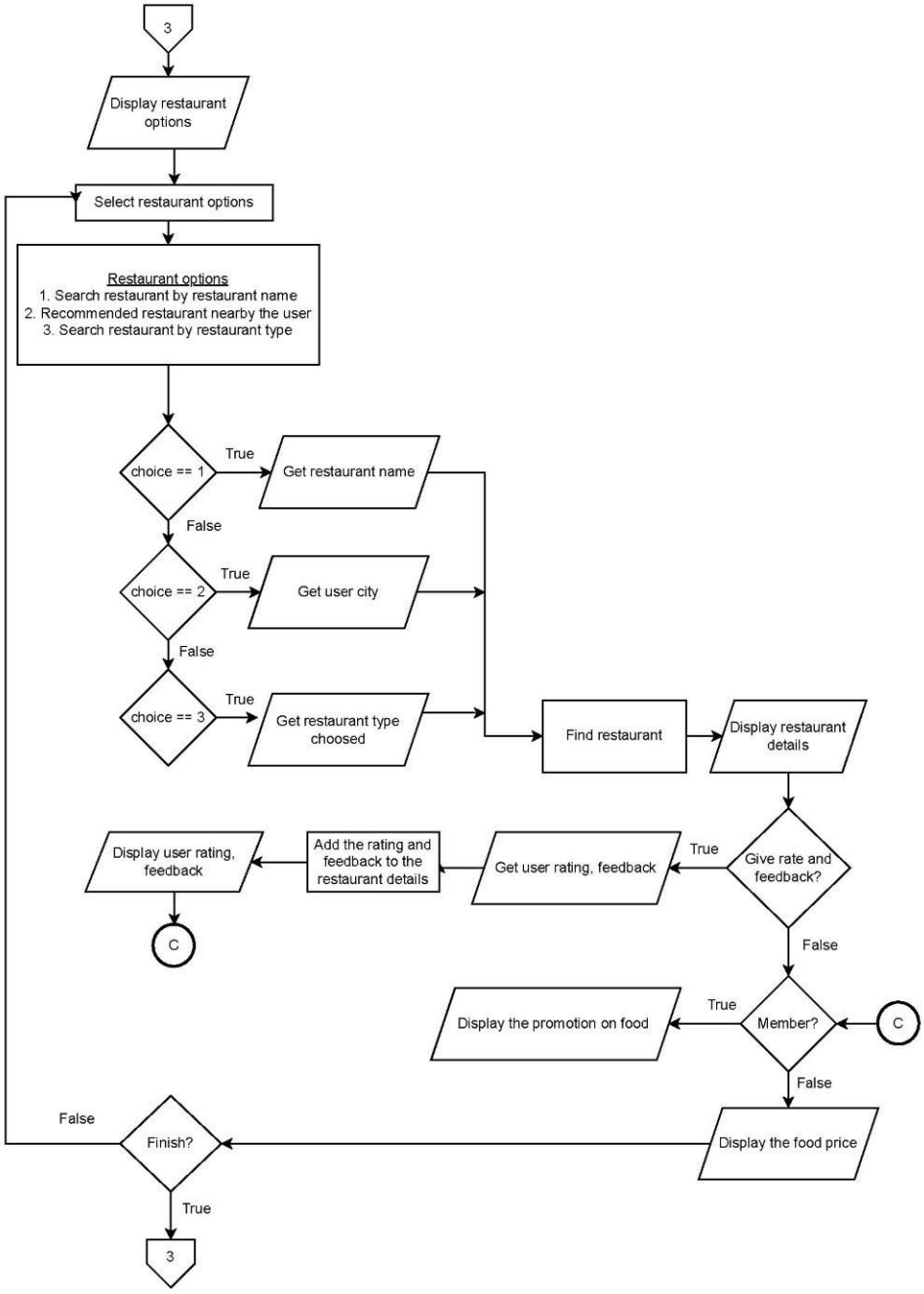
4.0 Flow Chart

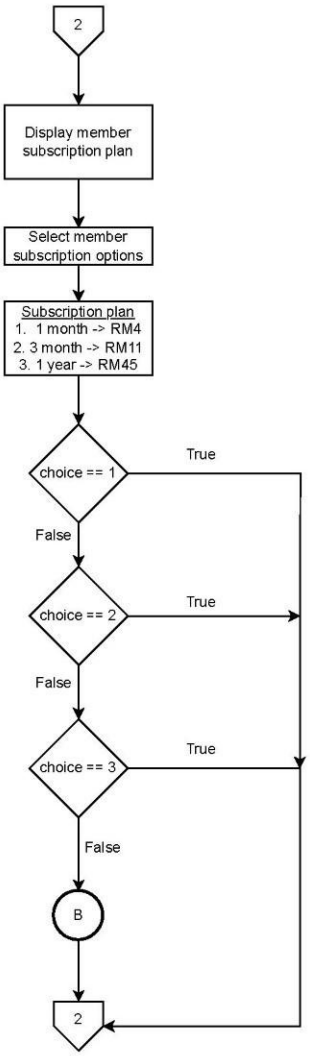


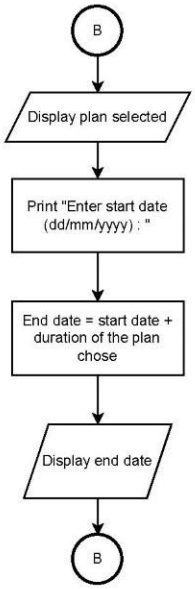

5
Print "Let's sign up "
Print "Create a username : "
Get username
Print "Create new password : "
Get password
Print "Please confirm your password : "
Get password verified
password == password verified
False
True
A

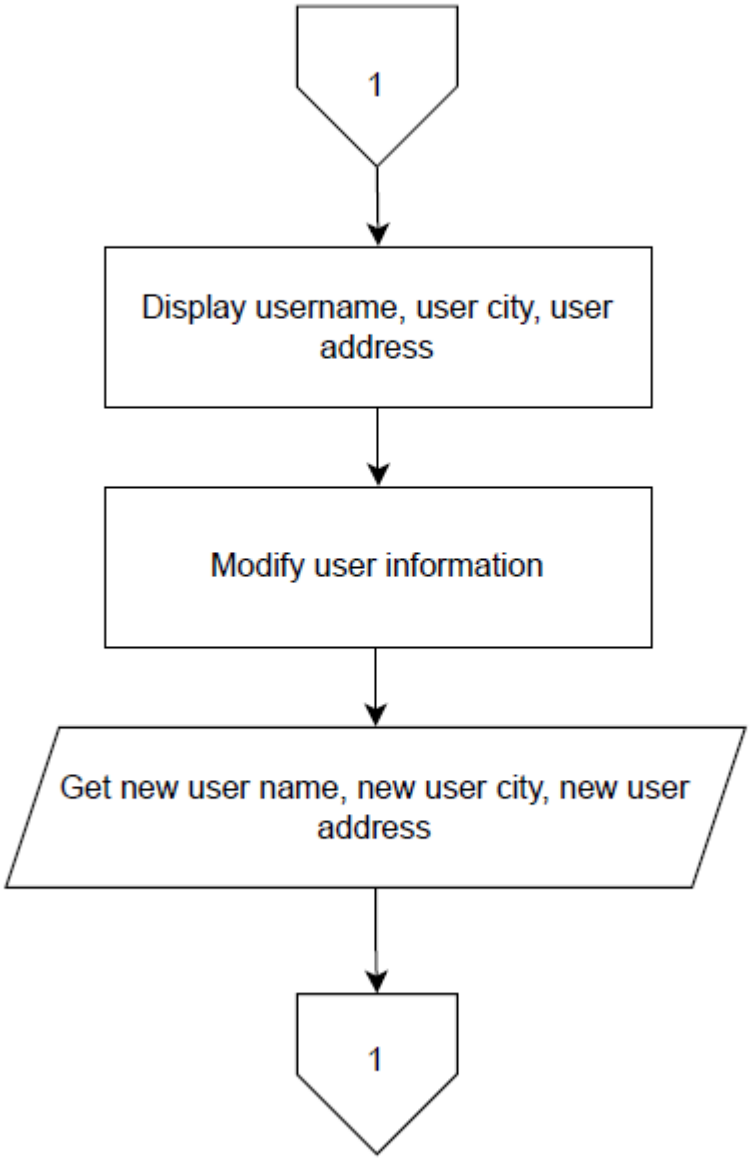

A
Print "Please fill the basic information: "
Print "Name : "
Get name
Print "Address : "
Get user address
Print "City: "
Get user city
5


4
Print "Enter your username : "
Get username
Print "Enter your password : "
Get password
4
password == password verified
False
True
4

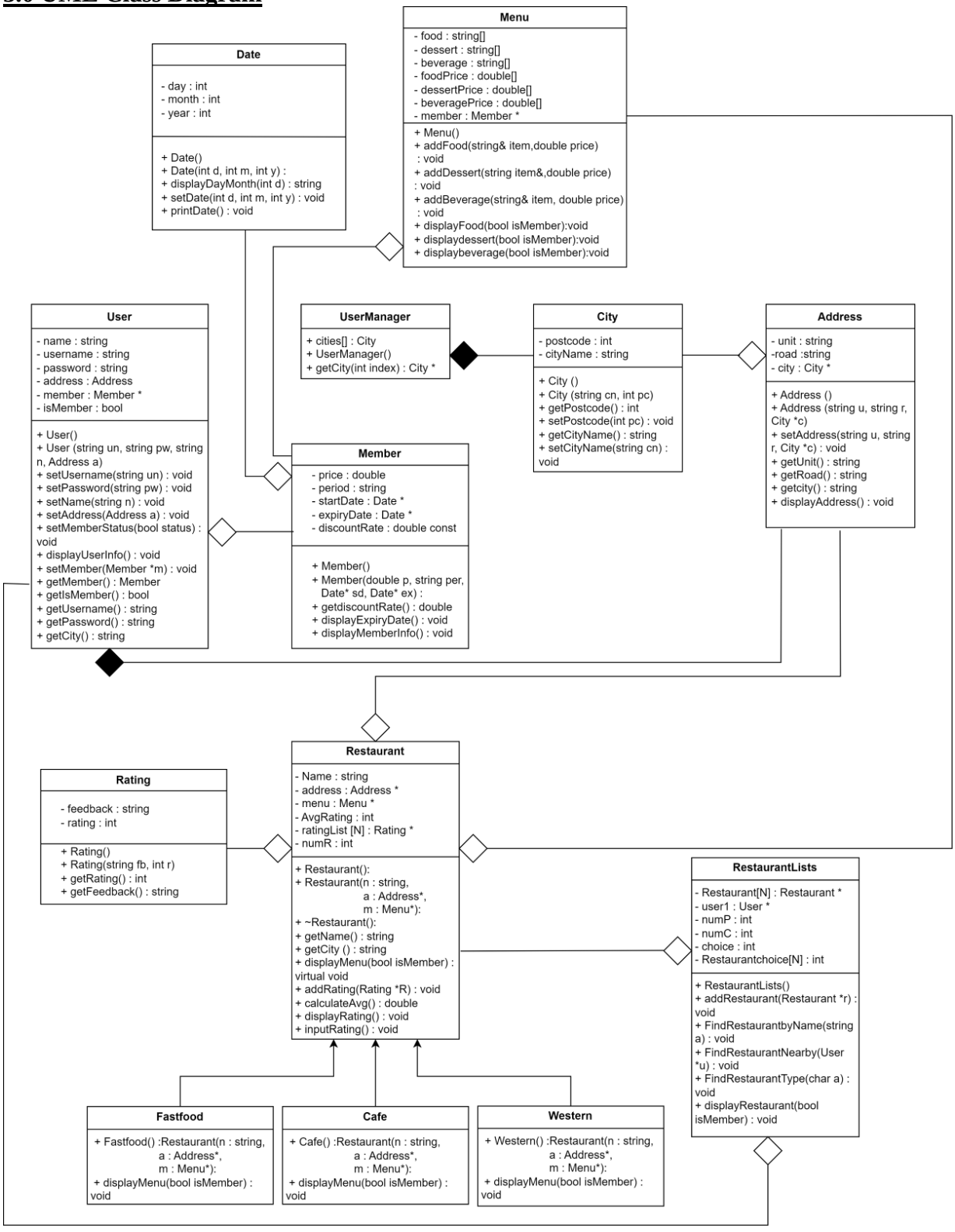

3
Display restaurant options
Select restaurant options
Restaurant options
1. Search restaurant by restaurant name
2. Recommended restaurant nearby the user
3. Search restaurant by restaurant type
choice == 1
True
Get restaurant name
False
choice == 2
True
Get user city
False
choice == 3
True
Get restaurant type choosed
Find restaurant
Display restaurant details
Give rate and feedback?
True
Get user rating, feedback
Add the rating and feedback to the restaurant details
Display user rating, feedback
C
False
Member?
True
Display the promotion on food
C
False
Display the food price
Finish?
False
True
3


2
Display member subscription plan
Select member subscription options
Subscription plan
1. 1 month -> RM4
2. 3 month -> RM11
3. 1 year -> RM45
choice == 1
True
False
choice == 2
True
False
choice == 3
True
False
B
2


B
Display plan selected
Print "Enter start date (dd/mm/yyyy) : "
End date = start date + duration of the plan chose
Display end date
B

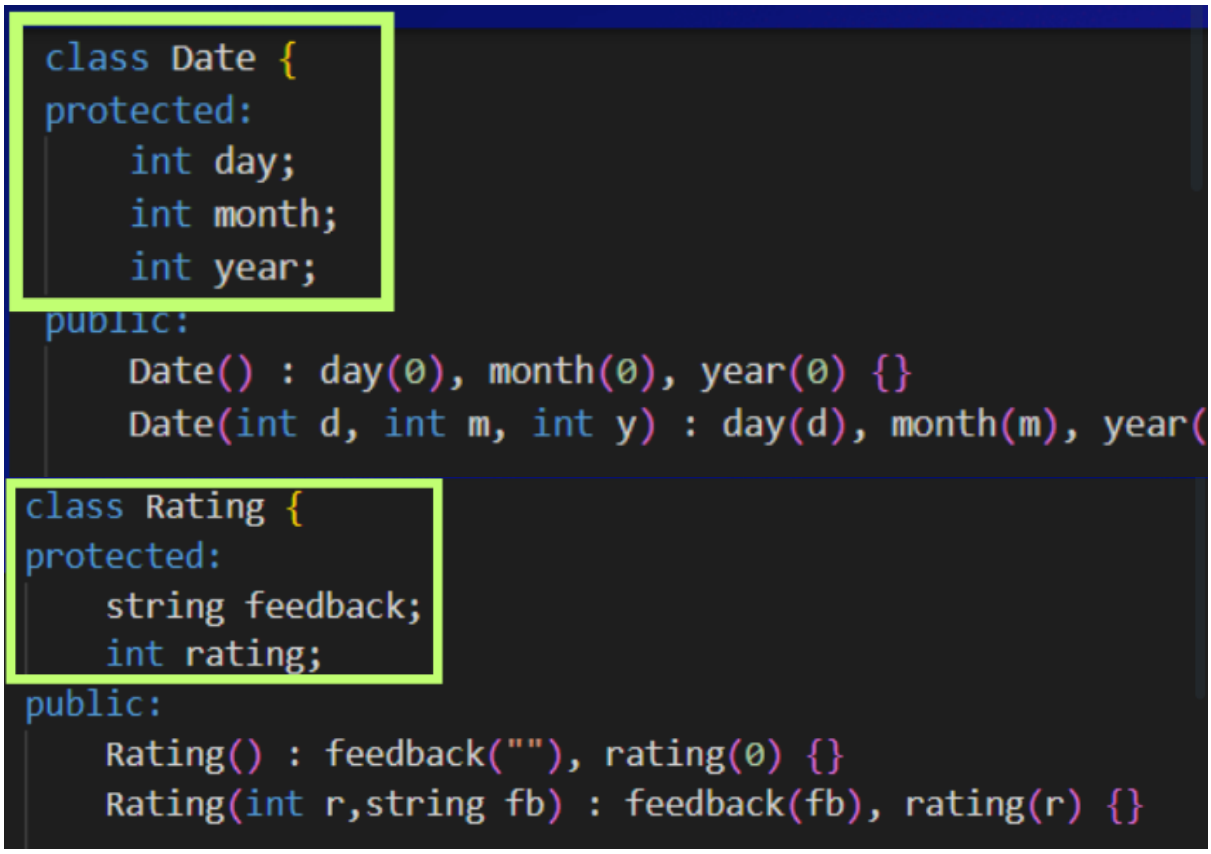

1
Display username, user city, user address
Modify user information
Get new user name, new user city, new user address
1

5.0 UML Class Diagram



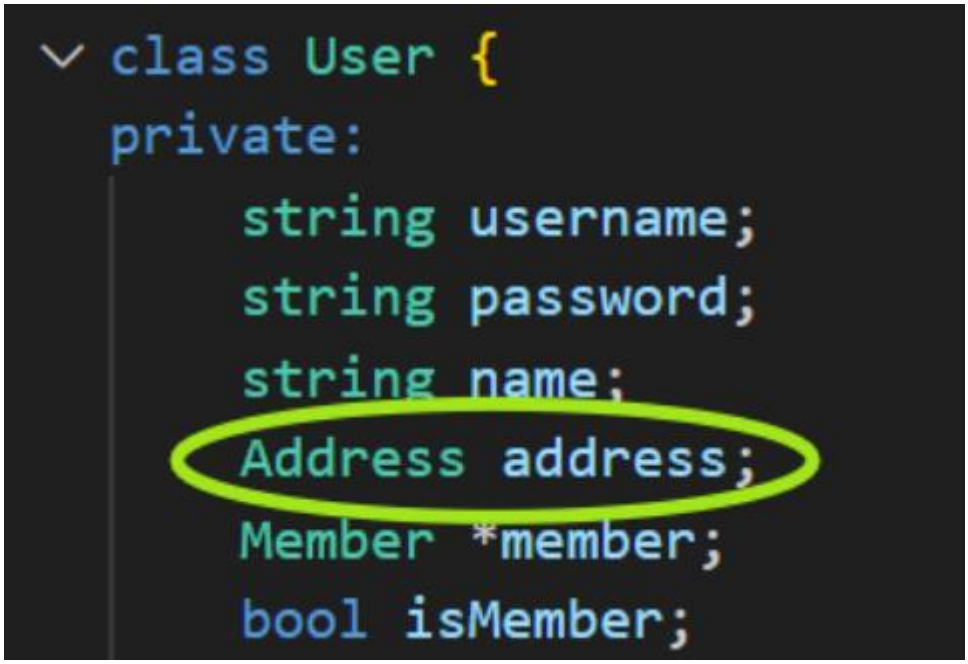
6.0 OOP Concepts
6.1 Encapsulation

Encapsulation is the bundling of data (attributes) and methods (functions) that operate on the data into a single unit, known as a class. It also restricts direct access to some of an object's components, which is a means of preventing accidental interference and misuse of the data.



6.2 Composition

Composition is a design principle where a class is composed of one or more objects of other classes, implying a "has-a" relationship.



6.3 Aggregation

Aggregation is a specialized form of composition where the lifetime of the contained objects does not depend on the lifetime of the container object, implying a "part-of" relationship but with independent lifetimes.

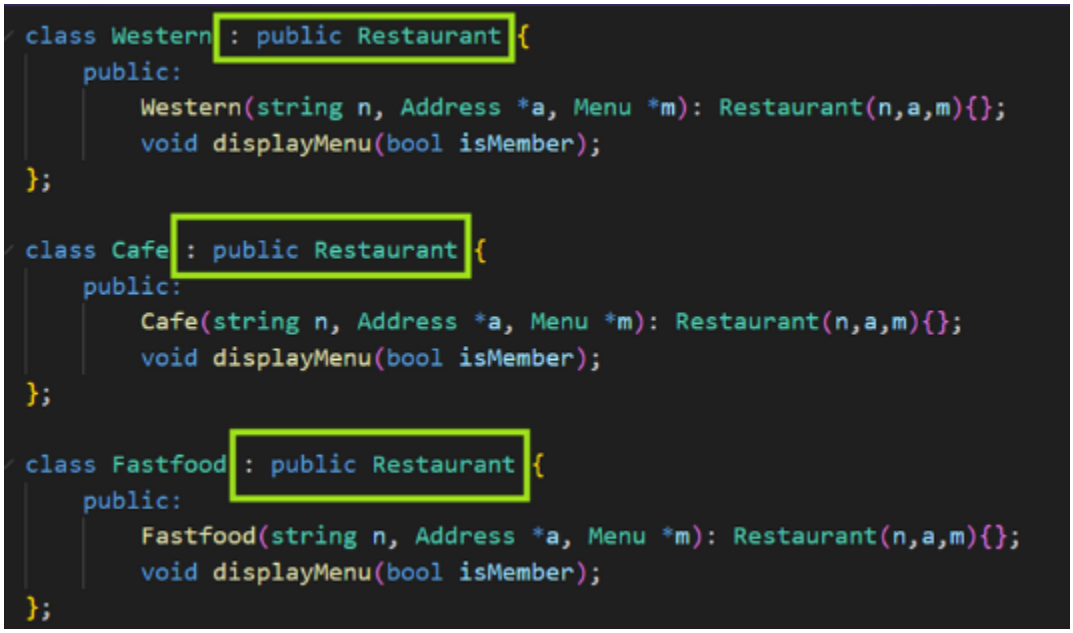
```
class Menu {
protected:
    string food[MAX_ITEMS],dessert[MAX_ITEMS],beverage[MAX_ITEMS];
    double foodPrice[MAX_ITEMS],dessertPrice[MAX_ITEMS],beveragePrice[MAX_ITEMS];
    int foodCount,dessertCount,beverageCount;
    Member *member;
```

```
class Member {
protected:
    double price;
    string period;
    Date *startDate;
    Date *expiryDate;
    double const discountRate;
```

```
class Address {
public:
    string unit;
    string road;
    City *city;
```

6.4 Inheritance

Inheritance is a mechanism where a new class (child class) is derived from an existing class (parent class). The child class inherits attributes and methods from the parent class and can also have additional attributes and methods.



```
class Western : public Restaurant {
    public:
        Western(string n, Address *a, Menu *m): Restaurant(n,a,m){};
        void displayMenu(bool isMember);
};

class Cafe : public Restaurant {
    public:
        Cafe(string n, Address *a, Menu *m): Restaurant(n,a,m){};
        void displayMenu(bool isMember);
};

class Fastfood : public Restaurant {
    public:
        Fastfood(string n, Address *a, Menu *m): Restaurant(n,a,m){};
        void displayMenu(bool isMember);
};
```

6.5 Polymorphism

Polymorphism allows objects of different classes to be treated as objects of a common superclass. It is often used to perform a single action in different ways.

```
class Restaurant {
        virtual void displayMenu(bool isMember);
```

```
void Restaurant::displayMenu(bool isMember) {
    cout << "Here is the menu of Restaurant" << endl;
    cout << "----------------------------------------------------------------\n";
}
```

```
void Western::displayMenu(bool isMember){
    Restaurant::displayMenu(isMember);
    menu->displayFood(isMember);
    menu->displaybeverage(isMember);
}
```

```
void Cafe::displayMenu(bool isMember){
    Restaurant::displayMenu(isMember);
    menu->displaydessert(isMember);
    menu->displaybeverage(isMember);
}
```

```
void Fastfood::displayMenu(bool isMember){
    Restaurant::displayMenu(isMember);
    menu->displayFood(isMember);
    menu->displaybeverage(isMember);
}
```

6.6 Array of Objects

An array of objects is a collection of instances of a class.

```
class UserManager {
public:
    City cities[10];

    UserManager() {
        cities[0] = City(0, "batu pahat");
        cities[1] = City(0, "johor bahru");
        cities[2] = City(0, "kluang");
        cities[3] = City(0, "kota tinggi");
        cities[4] = City(0, "kulai");
        cities[5] = City(0, "mersing");
        cities[6] = City(0, "muar");
        cities[7] = City(0, "pontian");
        cities[8] = City(0, "segamat");
        cities[9] = City(0, "tangkak");
    }
```

```
City* selectedCity = userManager.getCity(cityChoice - 1);
selectedCity->setPostcode(postcode);
if (selectedCity != nullptr) {
    users[userCount++] = User(username, password, name, Address(unit, road, selectedCity))
```

```
cities[newCityChoice-1].setPostcode(newPostcode);
Address newAddress(newUnit, newRoad, &cities[newCityChoice - 1]);
```

7.0 Conclusion

In this project, we have gained valuable insights and practical experience in software development and team collaboration. We successfully implemented key C++ concepts such as association, composition, inheritance, encapsulation, polymorphism, and array of objects, which significantly enhanced our understanding of object-oriented programming and design principles. This hands-on experience allowed us to apply theoretical knowledge to a real-world problem, reinforcing our learning and boosting our technical proficiency.

Moreover, we have experienced the dynamics of working in a team to develop a comprehensive program. Through effective communication and collaboration, we managed to integrate our individual contributions into a cohesive system, demonstrating the importance of teamwork in achieving complex project goals. This process also taught us how to troubleshoot and resolve errors efficiently, utilizing various resources to overcome challenges and improve our problem-solving skills.

We hope that our restaurant recommendation system will provide users with a convenient and reliable tool to discover their desired dining options, ultimately enhancing their overall dining experience. Through this project, we have not only developed a useful application but also grown as programmers and collaborators, preparing us for future endeavors in the field of software development.