UNIVERSITI TEKNIKAL MALAYSIA MELAKA

FAKULTI TEKNOLOGI MAKLUMAT DAN KOMUNIKASI

W O R K S H O P 1

R E P O R T

Name : NUR SHAHIRA ATILIA BINTI ZAINUDDIN

Matric Number : B032210369

Program : BITD

Project Title : RESIDENTIAL CLEANING SERVICE RESERVATION

Supervisor Name : NOOR AZILAH DRAMAN@MUDA

Supervisor Signature :

Evaluator Name : AMIR SYARIFUDDIN BIN KASIM

EXECUTIVE SUMMARY

The project domain chosen is business and service, which is the cleaning service business. The cleaning companies provide professional service to ensure cleanliness around the client's space. This project will focus on residential cleaning, which is in high demand. Although there are numerous businesses that offer excellent service, the problems faced by employees and clients associated with the service are still growing. This problem starts with the way companies handle their reservation management system. This often leads to issues such as staffs' uncertainty of their actual task, miscalculation of total and overlook of customers' opinion. In order to deal with these problems, the project will develop a system that has clear and informative reservation form, customers' feedback friendly system and has dependable system to make calculation. This project's aim is to help minimize the issues and improve the system to bring a satisfying experience for both employees and clients. This is important to help the business grow and have a reliable management system that is highly useful for everyone involved.

TABLE OF CONTENTS

PAGE

CHAPTER 1: INTRODUCTION

1.1 Introduction

Cleaning Service Business is service which an organisation will assign their employees or cleaners as individual or a team to customer's house that made a reservation through a provided system by the company. The focus of this project will revolve around residential cleaning that involve customer as individual and their property. It is quietly common for people to call for cleaning service instead of doing the house chores themselves as their personal life would prevent them from doing so. The company would build system that require the customer to provide details to help them organise appointment. This system is assessable by both customers and staffs.

1.2 Problem Statement

A few issues would rise up when the system is being use:

i. No clear service details understanding. Common complains in cleaning service is about cleaners' poor performance and quality of job. This is because they don't have the right understanding and detailed explanation about their job.
ii. Lack of customer engagement. It will be challenging for company to grow if they don't consider to get response from their customer as they don't know what drives customers' satisfaction.
iii. Service cost miscalculation. Wrong charging strategy make is one of the major reason why companies lose their customers or let themselves working for free which will cause a great loss for the companies.

1.3 Objective (s) of the project

This project embarks on the following objectives:

a. To develop better job request form template and improve staffs understanding about their job to ensure that excellent service is delivered.

b. To generate a system which accept effective feedback from customer to generate analysis and give opportunity for company to make improvement.
c. To ensure that the system will be able to generate the correct pricing way for every reservation and produce healthy profit for the company.

1.4 Scope

1.4.1 Module to be developed

i. Registration and Login module – Customer and staff can register their new account in the system and login into the system with their registered account.
ii. Account Edit module – Customer can modify or delete their account.
iii. Reservation Request module – Customer will provide their personal details and state their needs from the service which include date and time, selection of package, and extra requirement. When the customer confirm their reservation, the system will calculate the total cost and show the customer.
iv. Reservations View module – Customer can review their reservation.
v. Feedback Adding module – Customers will be able to leave feedback and rate.
vi. Feedbacks View module – Staff and customer may view feedbacks by other customers.
vii. Package Adding module – Staff can add new packages to the service system.
viii. Package Sales Report module – Staff can review number of reservation and rate average for every packages for analysis purpose.

1.4.2 Target User

i. Staff
Staff can use register and login module and access to package adding, feedbacks view and package sales report module.
ii. Customer
Customer may use register and login module and access to reservation request, reservations view, feedback adding, and feedbacks view modules. Customer may also use account edit module to modify their

account.

1.5 Project Significance

This project is important to ensure that the system's user will get good experience when using the system. This will lead to a pleasing performance when the task is done thus it can help the business to grow. Motivation and inspiration that lead to this project is by the current situation and challenges faced by cleaning service business and the people that regularly hire from the service.

1.6 Gantt Chart of Project Activities

No	Task	Week														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Planning - Composing a proposal including executive summary, problem statements, objectives, scope, and structured chart.															
2	Analysis & Design - Designing flowchart, ERD, DFD, data dictionary and console design.															
3	Progress 1 - Building database of five figures and connected it to user interface of C++.															
4	Progress 1 - Developing main function and object class of register and login module for all target users.															
5	Progress 2 - Developing main functions and object class of modules for user customer															
6	Progress 3 - Developing main functions and object class of modules for user staff															
7	Documentation - Writing report to finalize the project.															

CHAPTER 2: ANALYSIS OF PROBLEM

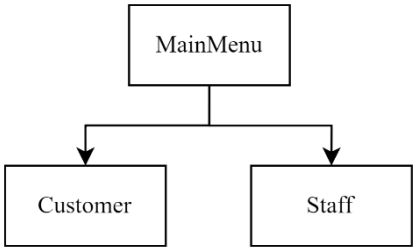
2.1 Problem Decomposition Description

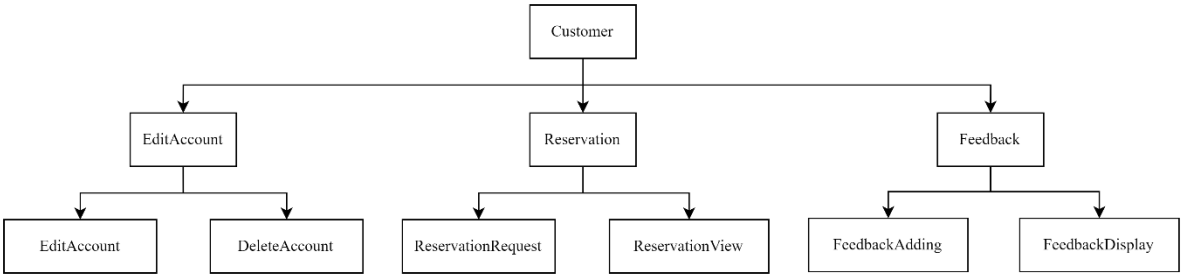
For the first problem where staffs don't clearly understand their service details, it is mainly because the company don't prepare packages that is tailored to the actual needs of customer as well as thorough reservation form. So the system will make sure that customer provides every details on form needed to organize appointment, the details would be such as package needed, extra requirement, date and time. They has to choose the service packages offered by company and additional requirement. Each packages will have its own description. Customers may later review their past reservation request.

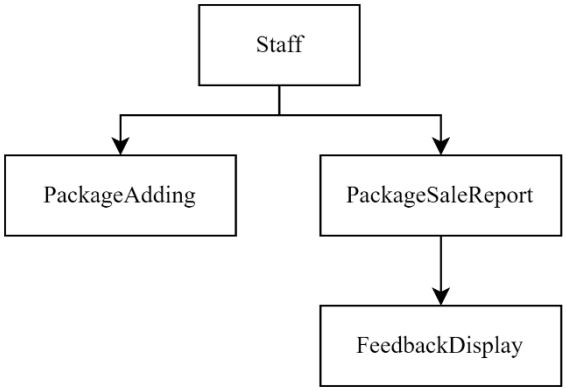
Secondly, the miscalculation of total cost mistake would be from the carelessness of every requirement consideration and inaccurate match with the hourly rate. From the reservation details including the additional requirement, the system will operate the calculation specified. This calculation will later be shown to customers for them to decide whether they will proceed with the request or not.

Lastly, slow company growth would be mainly because of the absence of customer involvement in the business which what made customers unsatisfied with the service will stay as the company flaw forever. When the service is done, customer should be able to leave their opinion and rate on feedback page. Staff can review the feedbacks and the rate average alongside number of reservation of each packages and collect it to make analysis of how customer feel about the package service whether it is positive, negative or mixed.

2.2 Structured Chart



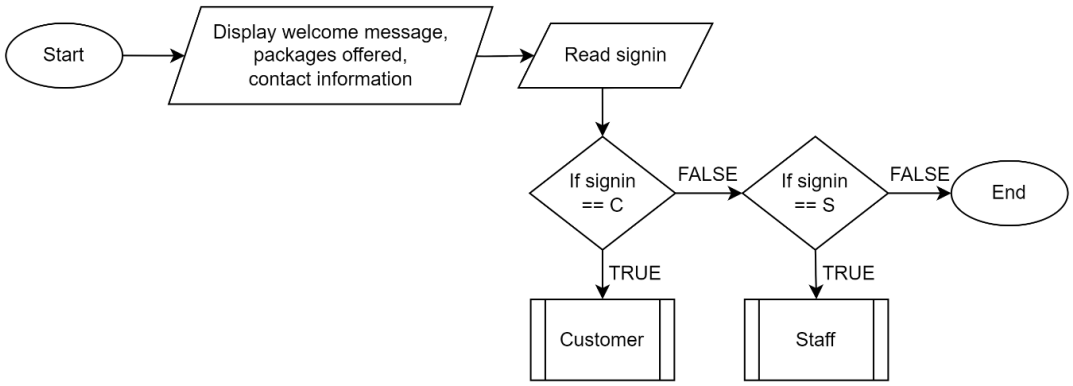




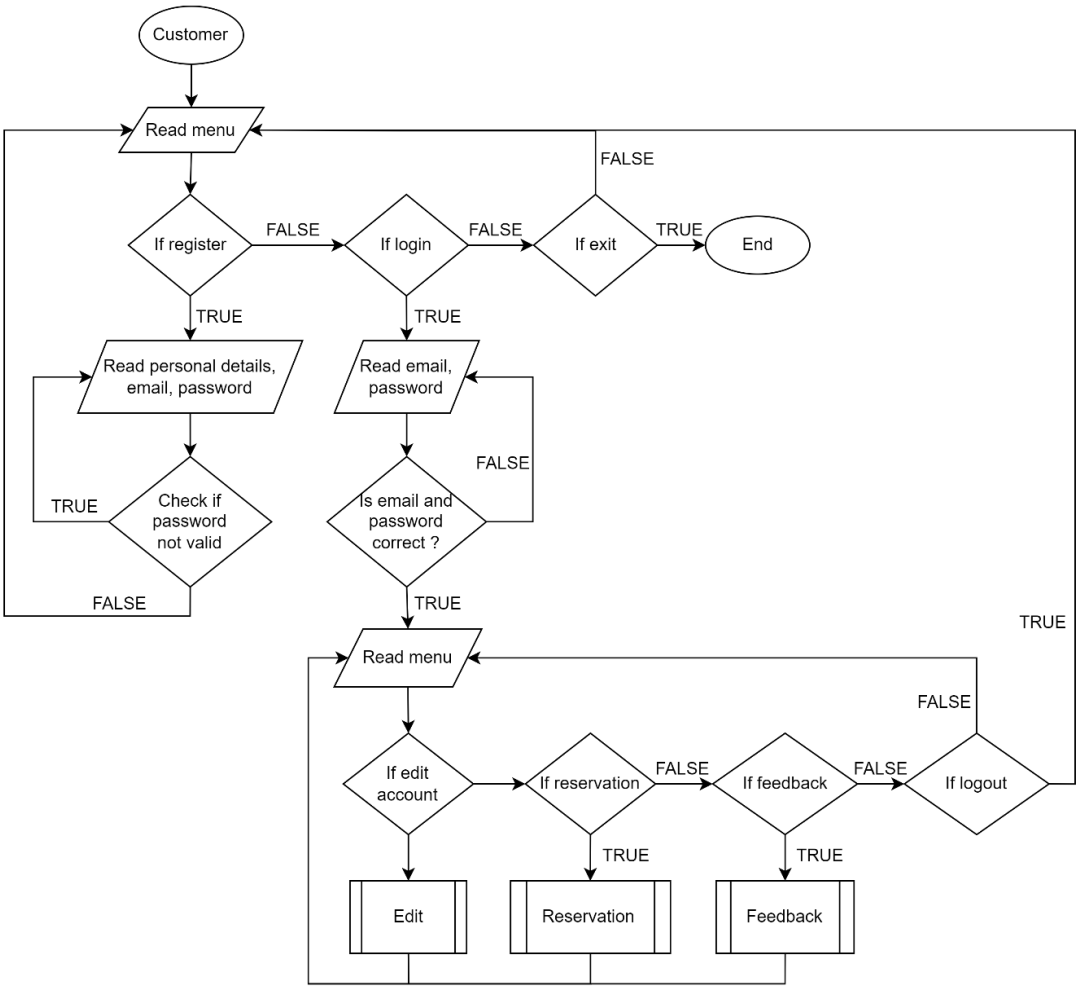
CHAPTER 3: DESIGN

3.1 Flowchart

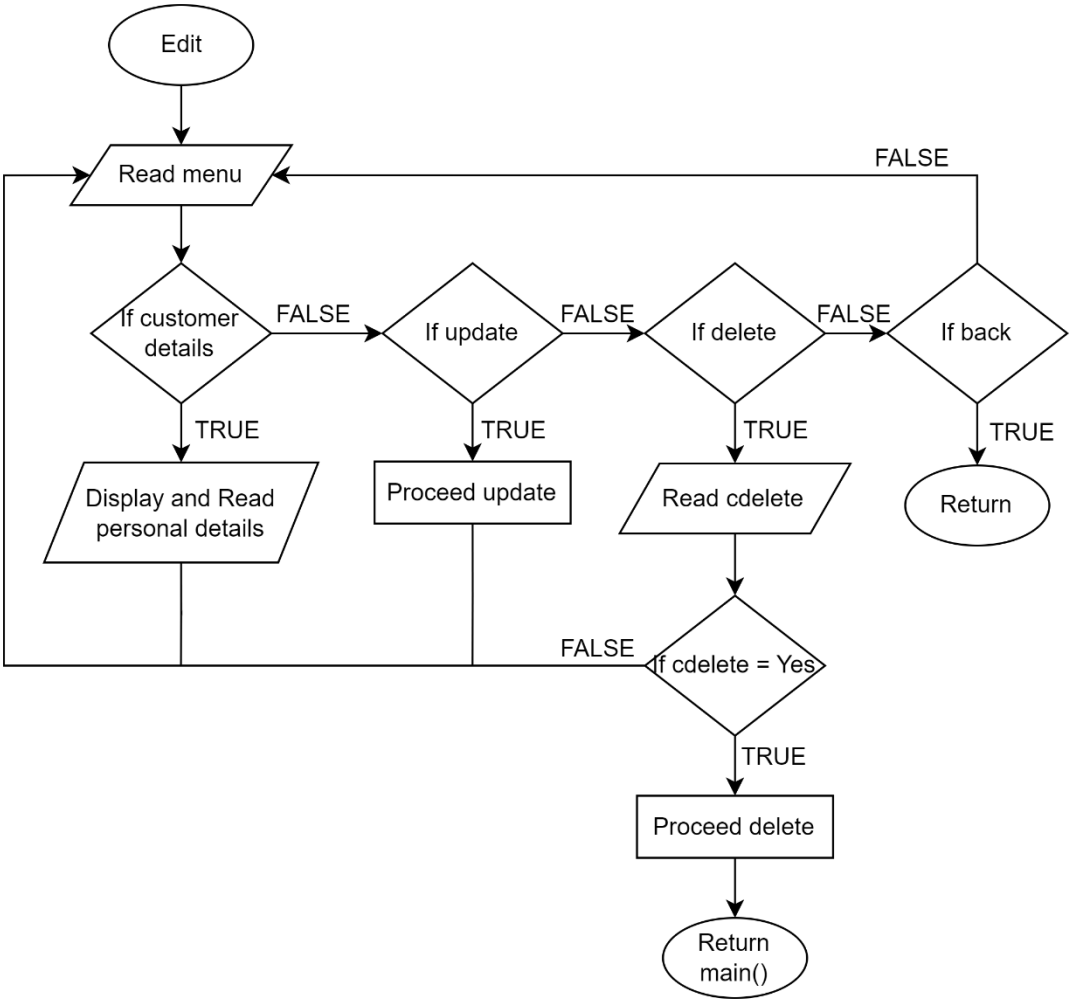
Main Page



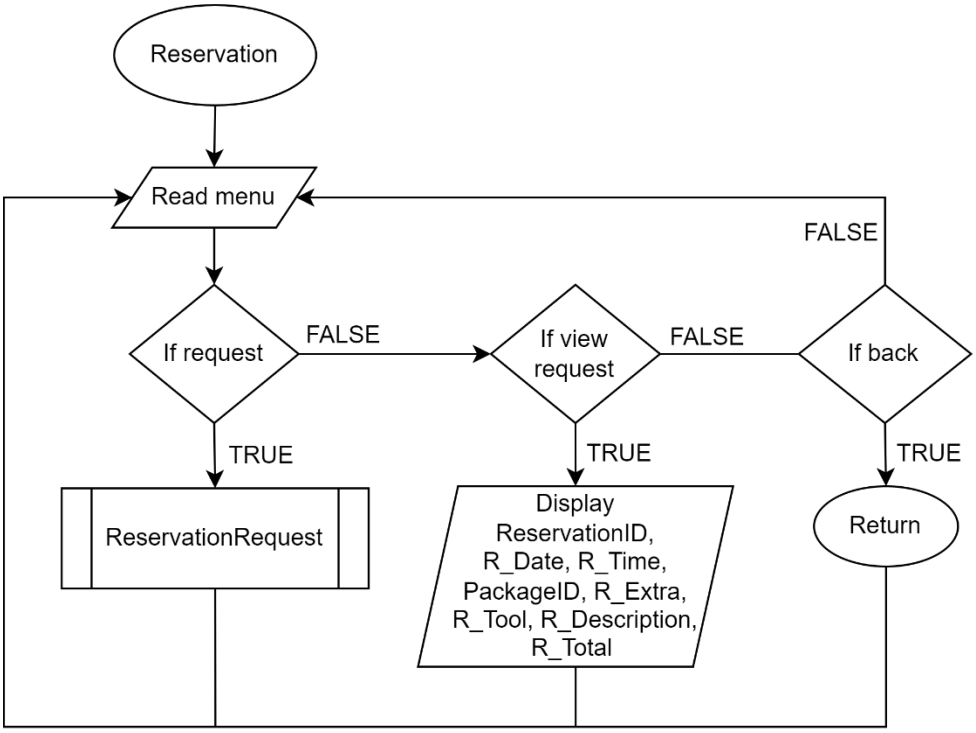
Customer


Customer
Read menu
If register
FALSE
If login
FALSE
If exit
TRUE
End
FALSE
TRUE
TRUE
Read personal details, email, password
Read email, password
TRUE
Check if password not valid
Is email and password correct ?
FALSE
FALSE
TRUE
TRUE
Read menu
FALSE
If edit account
If reservation
FALSE
If feedback
FALSE
If logout
TRUE
TRUE
TRUE
Edit
Reservation
Feedback

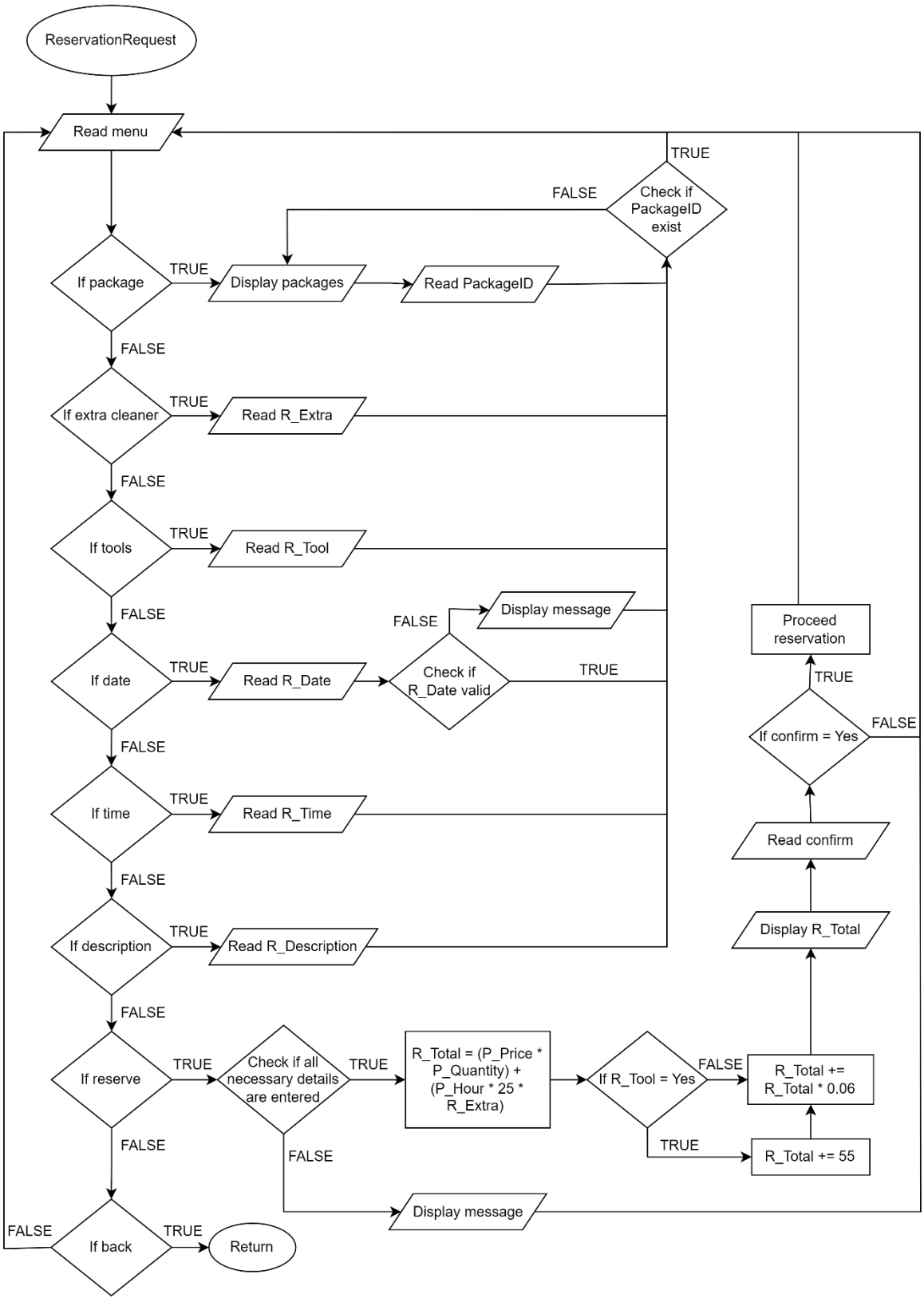
Customer – Edit



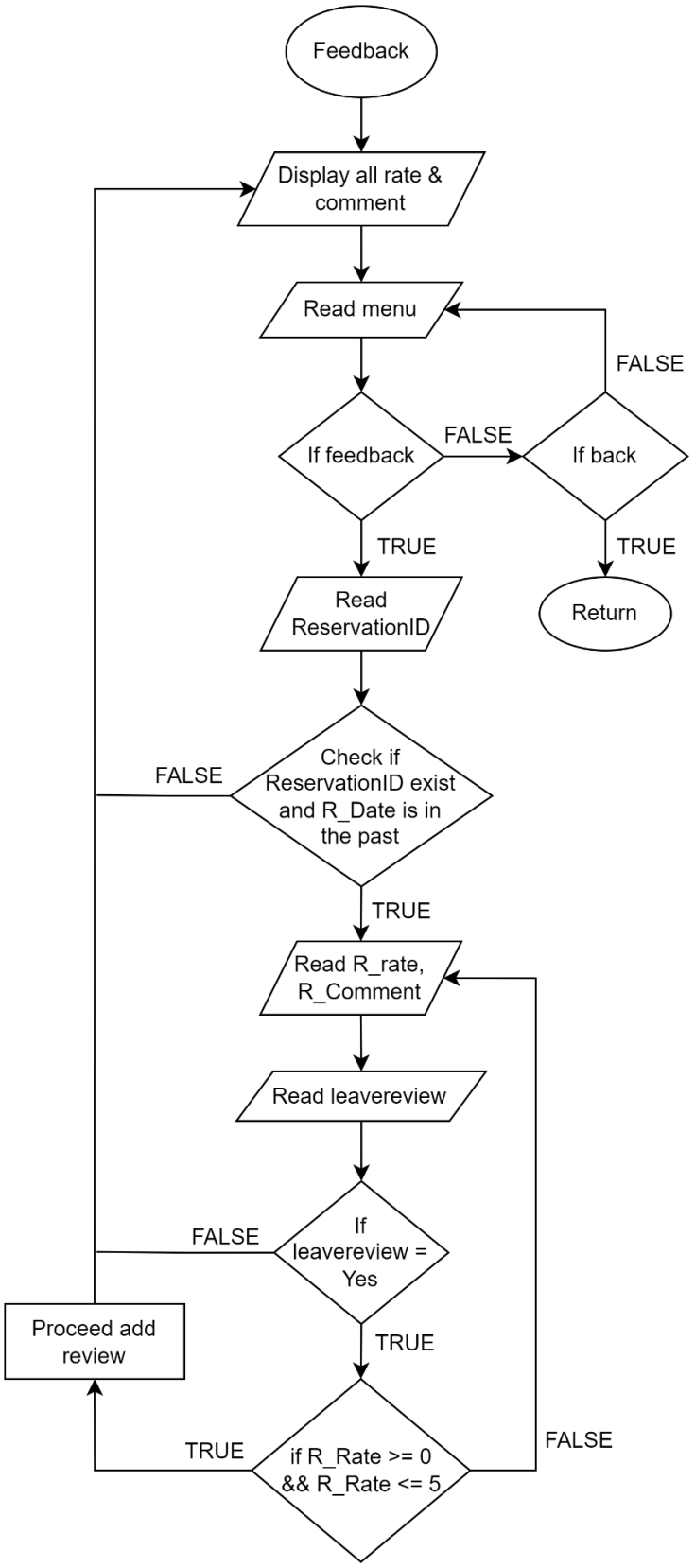
Customer – Reservation



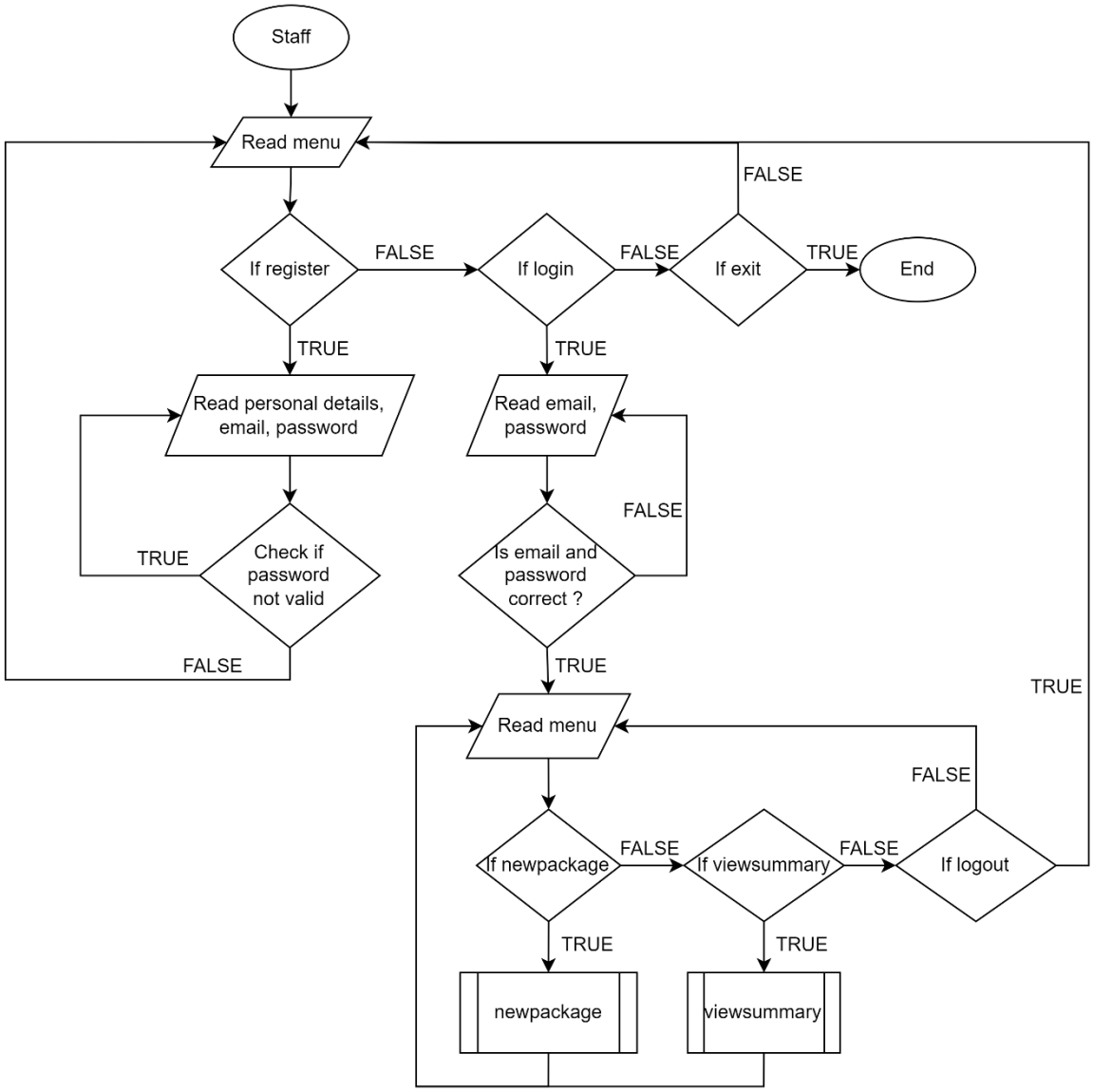
Customer – Reservation – Request



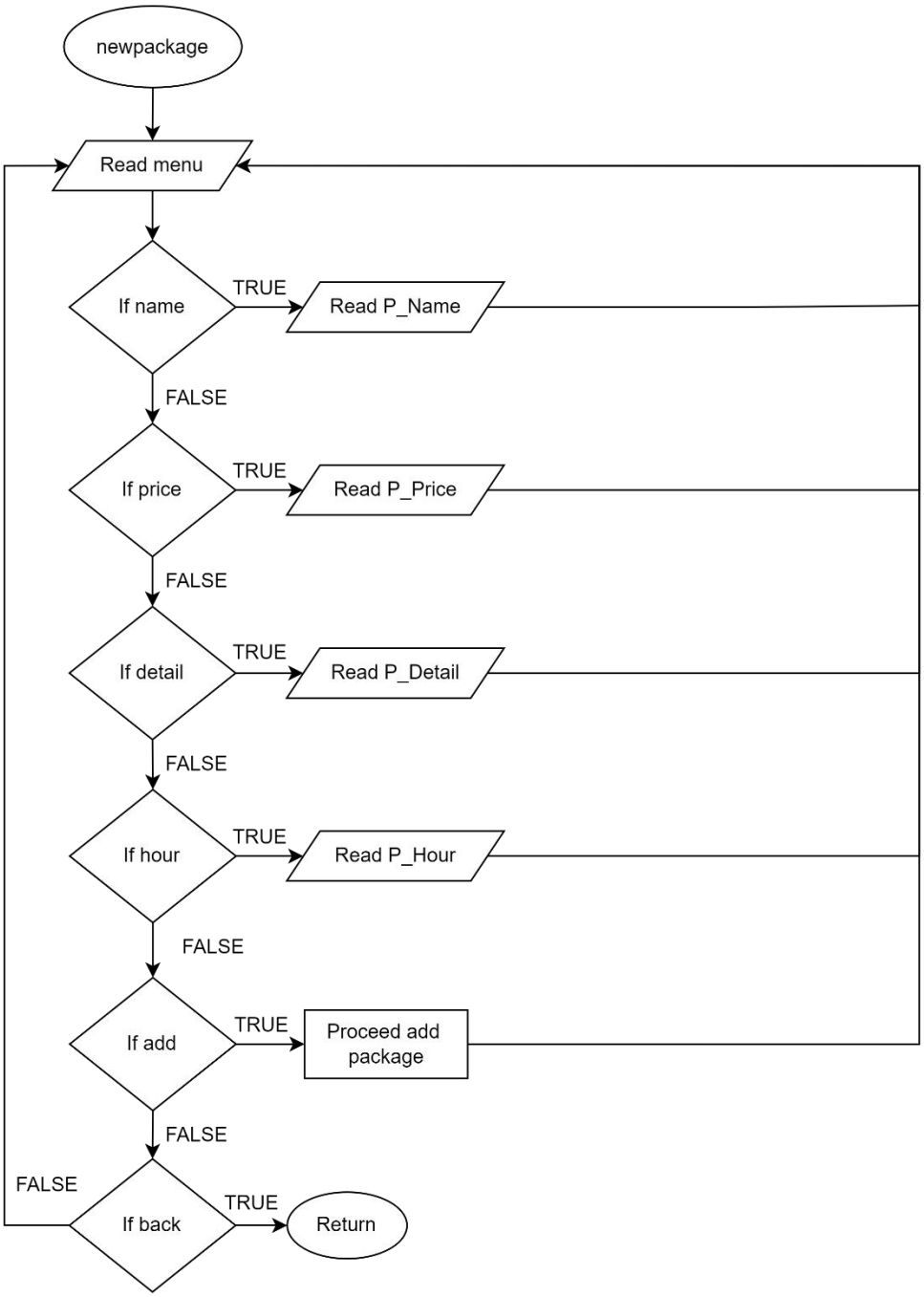
Customer – feedback



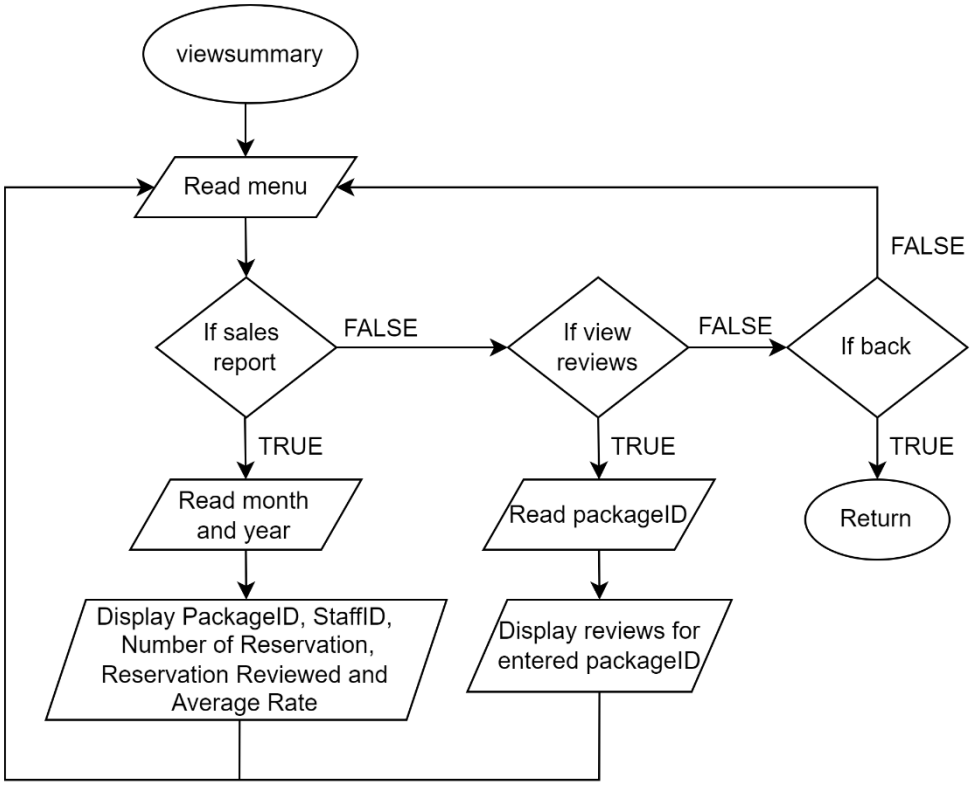
Staff


Staff
Read menu
If register
FALSE
If login
FALSE
If exit
TRUE
End
FALSE
TRUE
TRUE
Read personal details, email, password
Read email, password
Check if password not valid
TRUE
Is email and password correct ?
FALSE
FALSE
TRUE
TRUE
Read menu
FALSE
If newpackage
FALSE
If viewsummary
FALSE
If logout
TRUE
TRUE
TRUE
newpackage
viewsummary

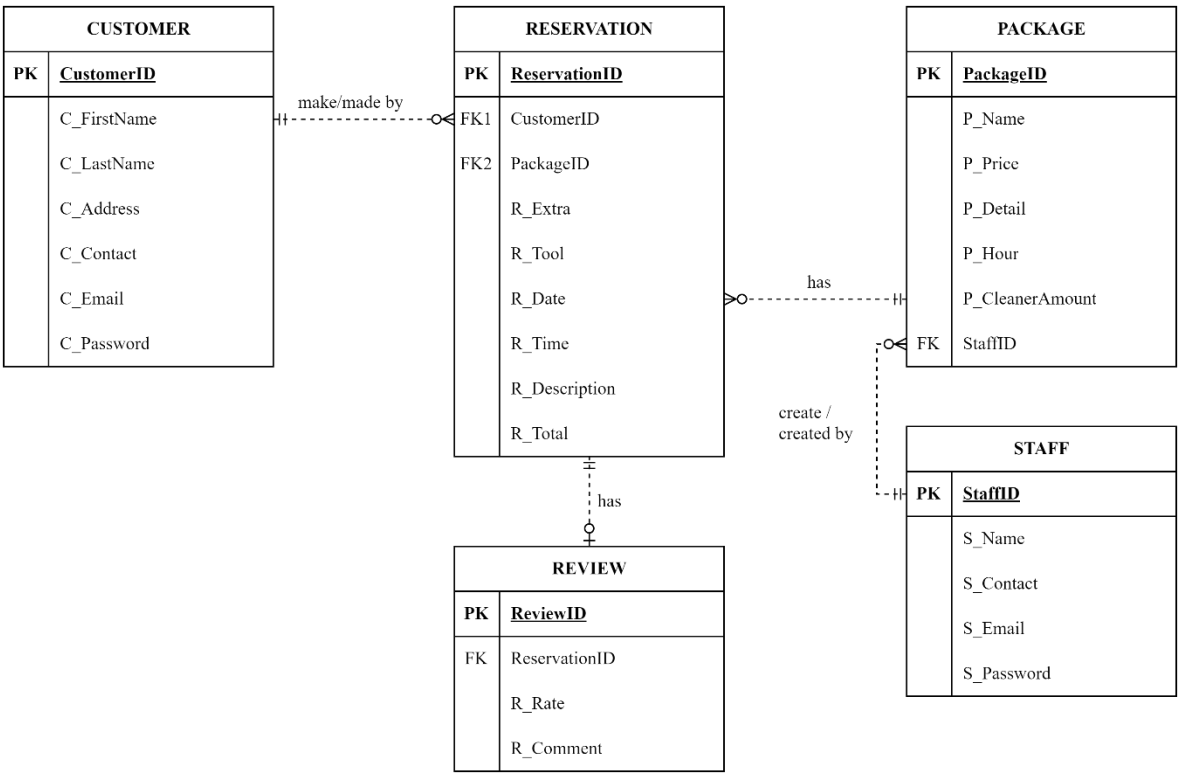
Staff – New Package



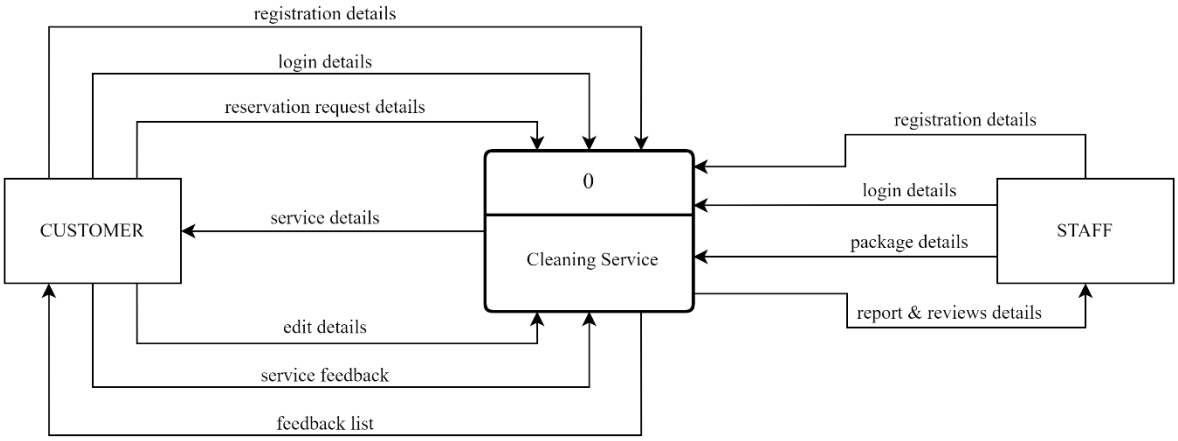
Staff – View Summary

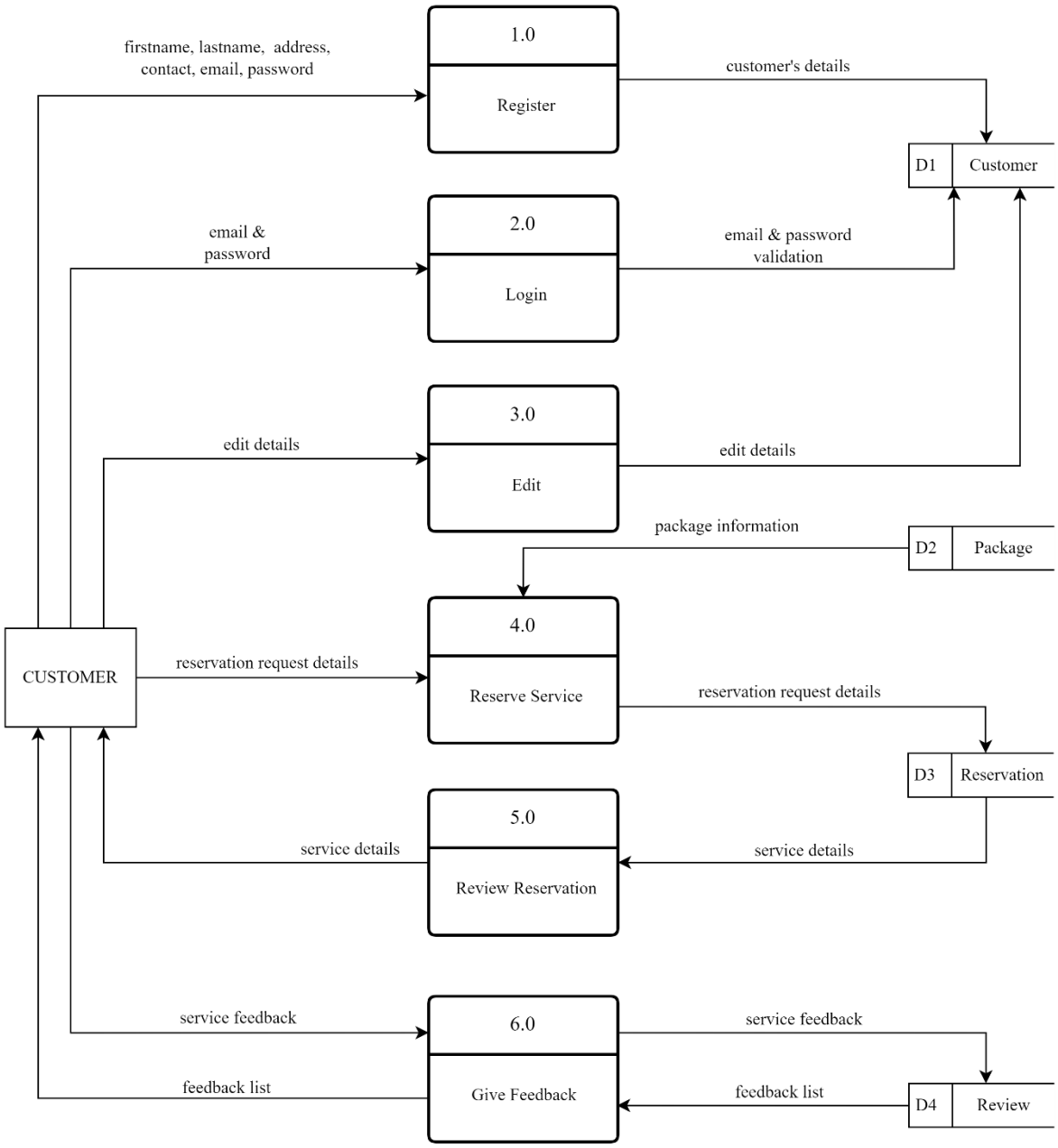


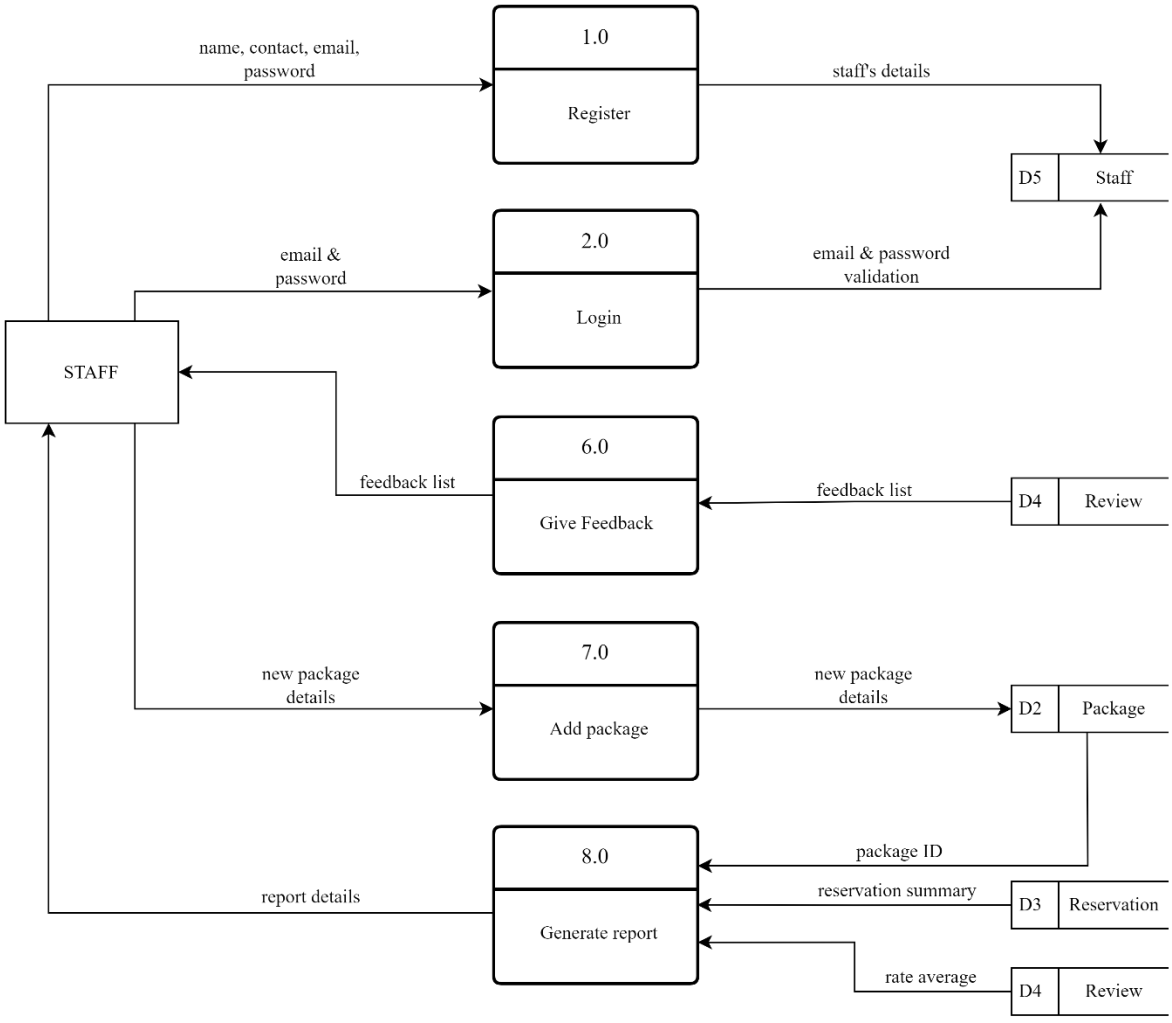
3.2 ERD

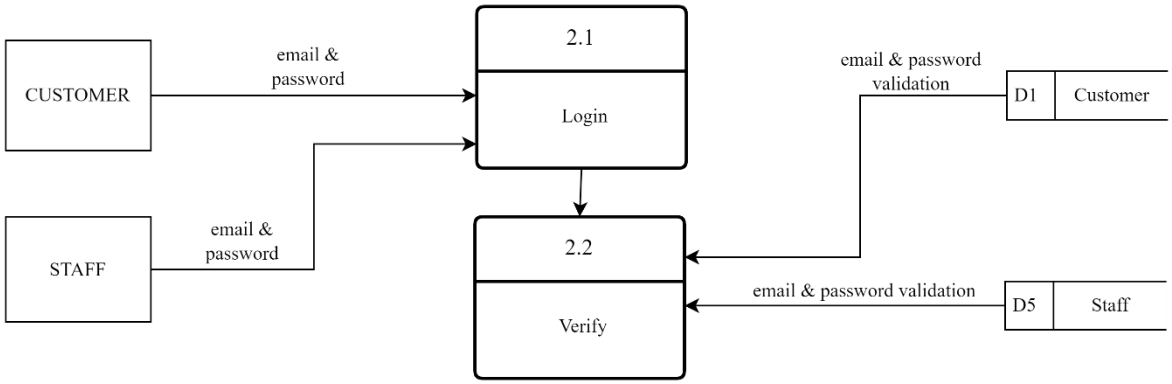

CUSTOMER
PK
CustomerID
C_FirstName
C_LastName
C_Address
C_Contact
C_Email
C_Password
make/made by
RESERVATION
PK
ReservationID
FK1
CustomerID
FK2
PackageID
R_Extra
R_Tool
R_Date
R_Time
R_Description
R_Total
has
PACKAGE
PK
PackageID
P_Name
P_Price
P_Detail
P_Hour
P_CleanerAmount
FK
StaffID
create /
created by
STAFF
PK
StaffID
S_Name
S_Contact
S_Email
S_Password
has
REVIEW
PK
ReviewID
FK
ReservationID
R_Rate
R_Comment

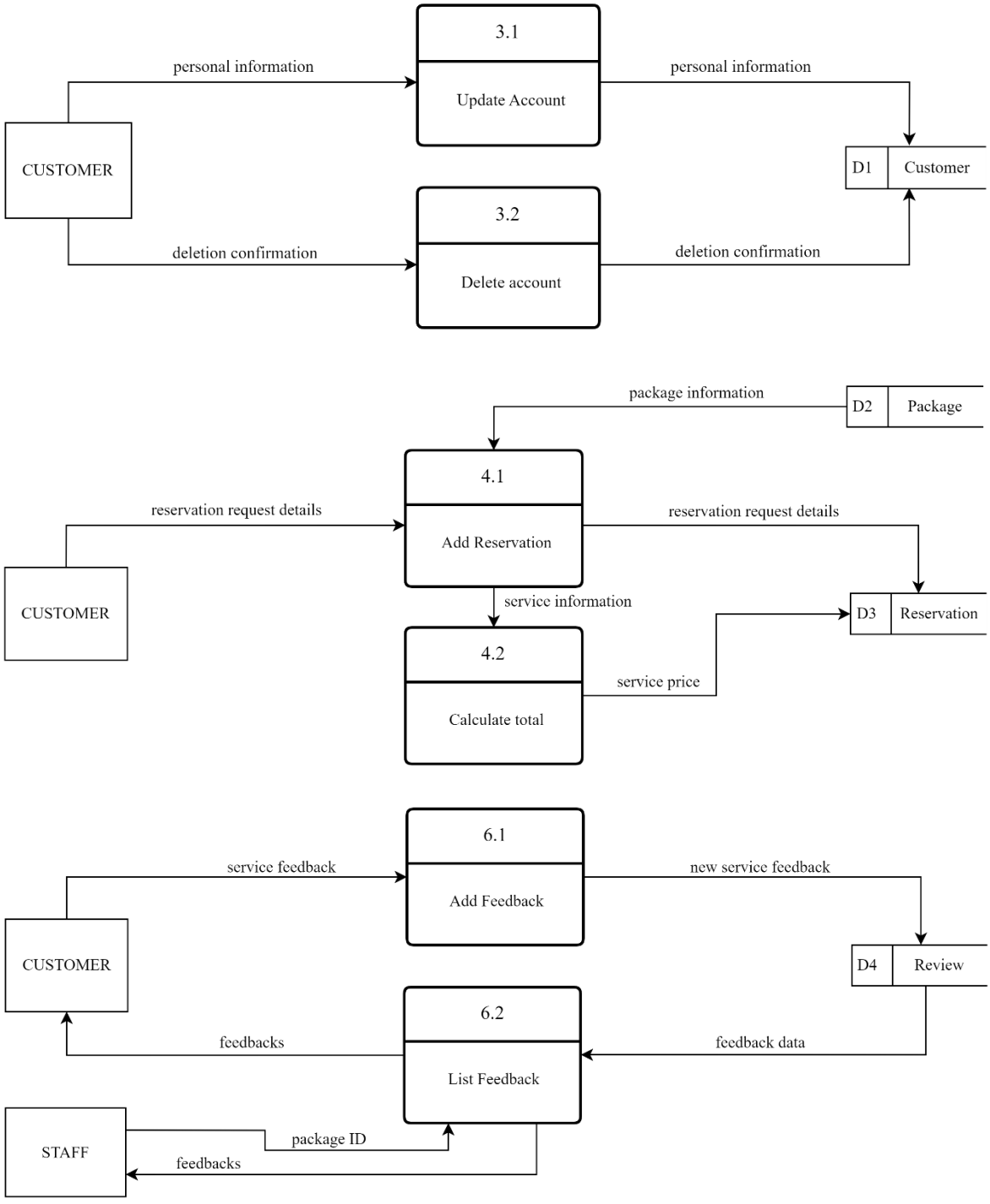
3.3 DFD


registration details
login details
reservation request details
CUSTOMER
service details
0
Cleaning Service
registration details
login details
package details
STAFF
report & reviews details
edit details
service feedback
feedback list


1.0
Register
firstname, lastname, address,
contact, email, password
customer's details
D1
Customer
2.0
Login
email &
password
email & password
validation
3.0
Edit
edit details
edit details
package information
D2
Package
4.0
Reserve Service
CUSTOMER
reservation request details
reservation request details
D3
Reservation
5.0
Review Reservation
service details
service details
service feedback
6.0
Give Feedback
service feedback
feedback list
feedback list
D4
Review


1.0
Register
name, contact, email, password
staff's details
D5
Staff
2.0
Login
email & password
email & password validation
STAFF
6.0
Give Feedback
feedback list
feedback list
D4
Review
7.0
Add package
new package details
new package details
D2
Package
8.0
Generate report
package ID
reservation summary
D3
Reservation
report details
rate average
D4
Review


CUSTOMER
email & password
2.1
Login
email & password validation
D1
Customer
STAFF
email & password
2.2
Verify
email & password validation
D5
Staff


3.1
Update Account
personal information
personal information
CUSTOMER
D1
Customer
3.2
Delete account
deletion confirmation
deletion confirmation
package information
D2
Package
4.1
Add Reservation
reservation request details
reservation request details
CUSTOMER
service information
D3
Reservation
4.2
Calculate total
service price
6.1
Add Feedback
service feedback
new service feedback
CUSTOMER
D4
Review
6.2
List Feedback
feedbacks
feedback data
STAFF
package ID
feedbacks

3.4 Data Dictionary

CUSTOMER							
ATTRIBUTE NAME	CONTENTS	DATA TYPE & SIZE	FORMAT	NOT NULL ?	DEFAULT VALUE	PK OR FK	FK REFERENCED TABLE
CustomerID	Customer's ID	INT				PK	
C_FirstName	Customer's first name	VARCHAR(25)		YES			
C_LastName	Customer's last name	VARCHAR(25)		YES			
C_Address	Customer's address	VARCHAR(200)		YES			
C_Contact	Customer's phone	VARCHAR(13)					
C_Email	Customer's email	VARCHAR(30)		YES			
C_Password	Customer's password	VARCHAR(12)		YES			
C_City	Customer's city	VARCHAR(50)		YES			
C_PostCode	Customer's Postcode	INT		YES			
C_State	Customer's State	VARCHAR(20)		YES			

STAFF							
ATTRIBUTE NAME	CONTENTS	DATA TYPE & SIZE	FORMAT	NOT NULL ?	DEFAULT VALUE	PK OR FK	FK REFERENCED TABLE
StaffID	Staff's ID	INT				PK	
S_Name	Staff's first name	VARCHAR(25)		YES			
S_Contact	Staff's phone	VARCHAR(13)		YES			
S_Email	Staff's email	VARCHAR(30)		YES			
S_Password	Staff's password	VARCHAR(12)		YES			

PACKAGE							
ATTRIBUTE NAME	CONTENT	DATA TYPE & SIZE	FORMAT	NOT NULL ?	DEFAULT VALUE	PK OR FK	FK REFERENCED TABLE
PackageID	Package's ID	INT				PK	
P_Name	Package's name	VARCHAR(25)		YES			
P_Price	Package's Price	DOUBLE		YES			
P_Detail	Packages' details	VARCHAR(4000)		YES			
P_Hour	Hour of package	INT		YES			
P_CleanerAmount	Amount of cleaner	INT		YES	2		
StaffID	Package creator's ID	INT		YES		FK	STAFF

RESERVATION							
ATTRIBUTE NAME	CONTENT	DATA TYPE & SIZE	FORMAT	NOT NULL ?	DEFAULT VALUE	PK OR FK	FK REFERENCED TABLE
ReservationID	Reservation's ID	INT				PK	
CustomerID	Customer's ID	INT		YES		FK	CUSTOMER
PackageID	Package's ID	INT		YES		FK	PACKAGE
R_Extra	Extra cleaner amount	INT					
R_Tool	Cleaning tools	VARCHAR(4)		YES			
R_Date	Reservation date	VARCHAR(12)	DD-MM-YYYY	YES			
R_Time	Reservation time	TIME	HH:MM:SS	YES			
R_Description	Reservation descriptio	VARCHAR(4000)					
R_Total	Reservation total cost	DOUBLE		YES			

REVIEW							
ATTRIBUTE NAME	CONTENT	DATA TYPE & SIZE	FORMAT	NOT NULL ?	DEFAULT VALUE	PK OR FK	FK REFERENCED TABLE
ReviewID	Review's ID	INT				PK	
ReservationID	Reservation's ID	INT		YES		FK	RESERVATION
R_Rate	Service rate	DOUBLE		YES			
R_Comment	Comment on service	VARCHAR(1000)					

3.5 Interface Design

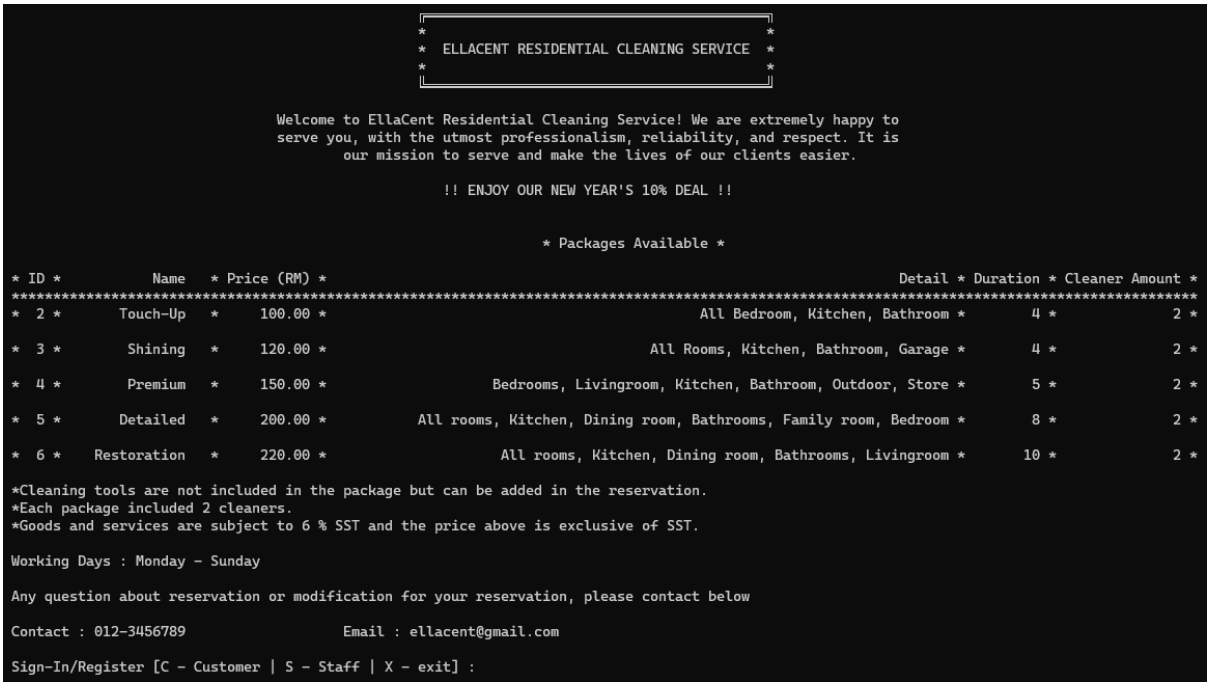


Figure 1 : Main Page

Figure 2 : Menu Selection Page

Figure 3 : Customer's Registration Page

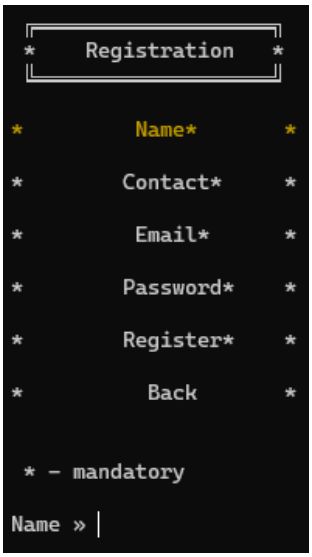


Figure 4 : Staff's Registration Page

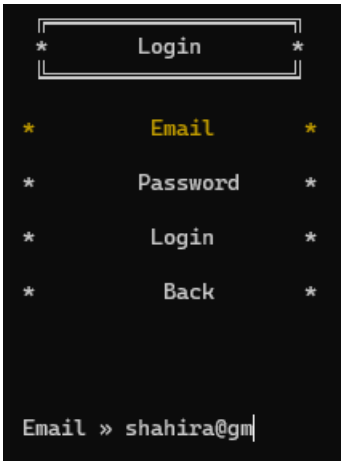


Figure 5 : Customer's & Staff's Login Page

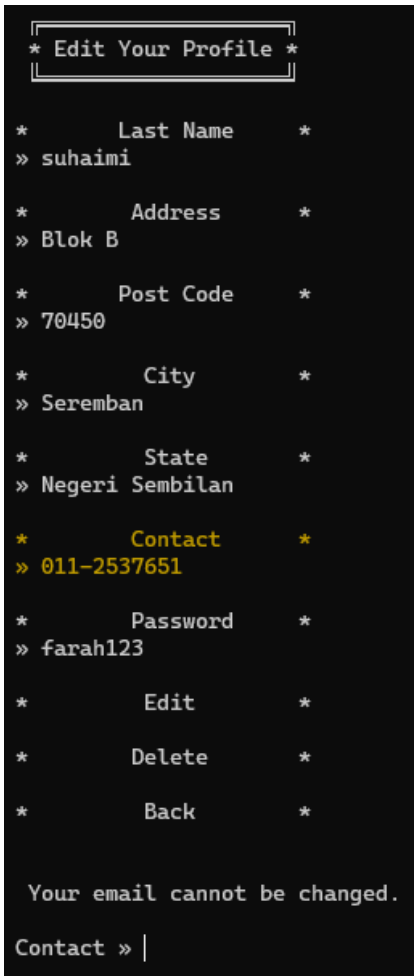


Figure 6 : Customer’s Details Edit Page

Figure 7 : Customer's Reservation Page

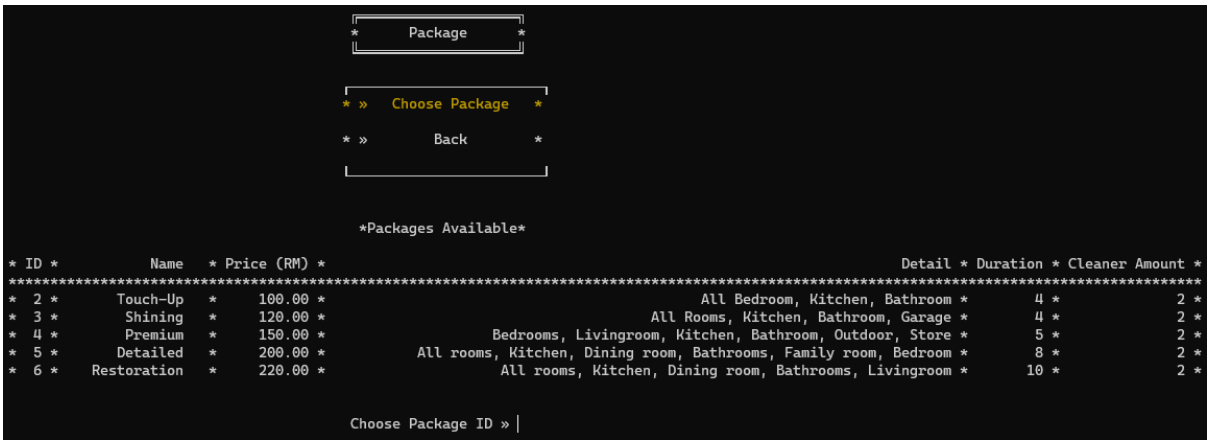


Figure 8 : Customer's Package Selection Page

Figure 9 : Customer’s Reservations View Page

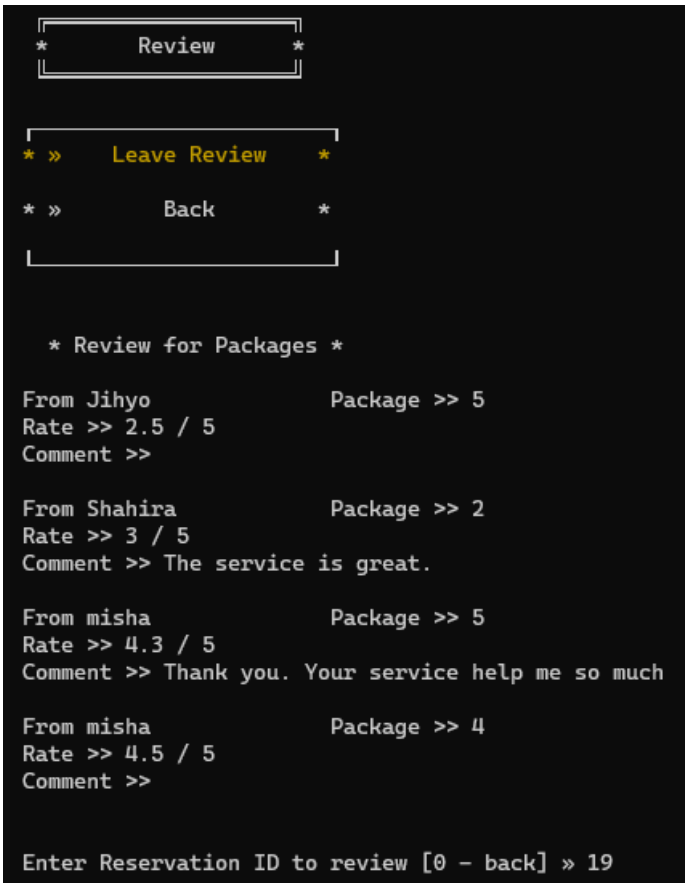


Figure 10 : Customer's Review List Page

Figure 11 : Customer's Leave Review Page

Figure 12 : Staff's New Package Page

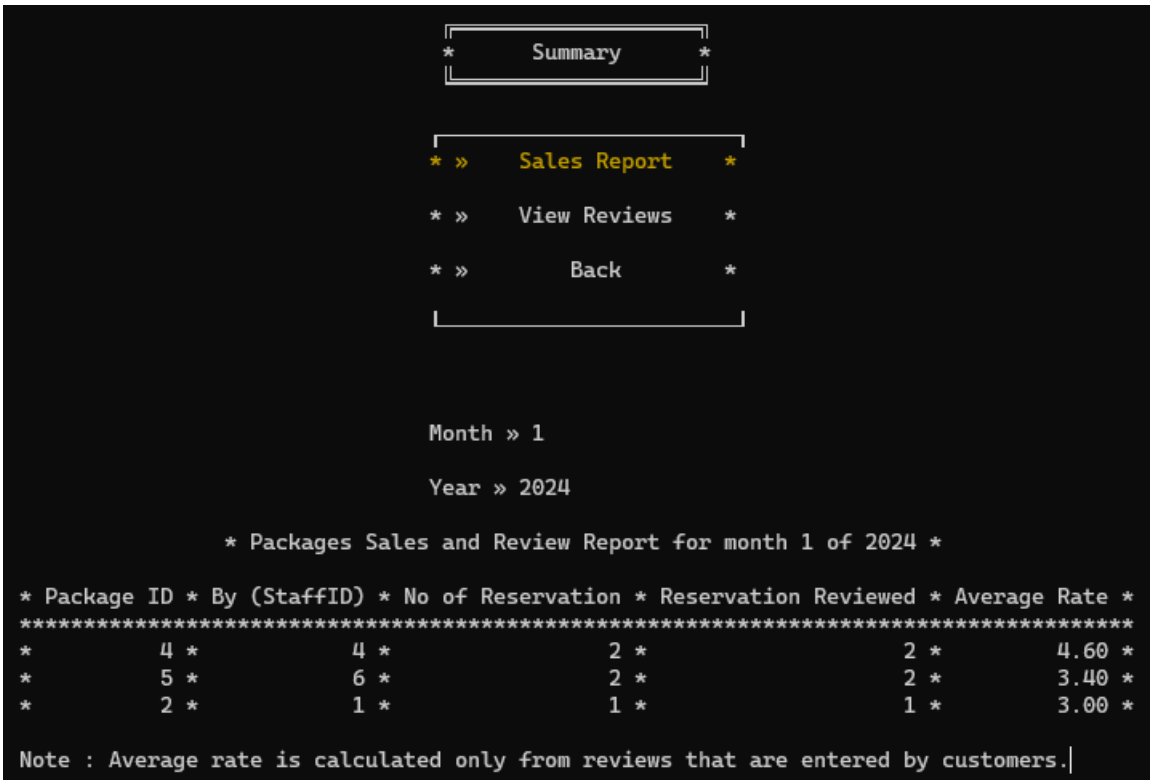


Figure 13 : Staff's Sales Report Page

Figure 14 : Staff's View Reviews for Specific Package Page

CHAPTER 4: IMPLEMENTATION

4.1 Naming Convention

Naming convention refers to rules in deciding the identifiers name for variables, functions, and other entities. For this system, each function declaration used PascalCase convention which each word started with uppercase letter and doesn't include character that represent space like underscore for example StaffLogin and Reservation.

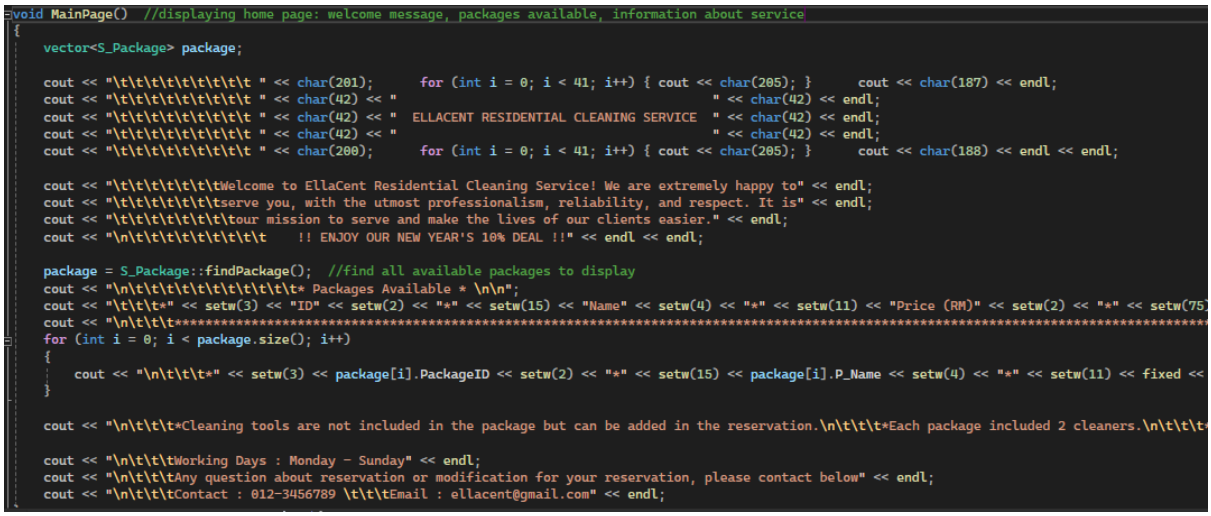
```
void MainPage();

//function prototype

//staff
void StaffRegistration();
void StaffLogin();
void StaffHome(S_Staff SS);
void NewPackage(S_Staff SS);
void ViewSummary();

//customer
void CustomerRegistration();
void CustomerLogin();
void CustomerHome(S_Account SA);
S_Account CustomerEdit(S_Account SA);
void Feedback(S_Account SA);

//customer's reservation
void Action(S_Account SA);
void Reservation(S_Account SA);
void ViewReservations(S_Account SA);
S_Reservation Package(S_Account SA, S_Reservation SR);
S_Reservation FindPackage(S_Account SA, int id, S_Reservation SR);

//external
string HidePassword(string pwd);
bool CheckDate(const string& dateinput);

//global variable
int id, day, month, year;
```

Figure 15 : Name Convention for Functions

4.2 Function

Functions are block of codes which should be declared in the global area and can be called multiple times from any part of the source code to perform action that is determined by the programmer.

There would be cases where the function may need parameters in order to perform its task. Any functions may return multiple, one, or no values at all. For this system, the functions mostly return no value so it use void return type for example MainPage() function which is called to only display the details of services.



```
void MainPage()  //displaying home page: welcome message, packages available, information about service
{
    vector<S_Package> package;

    cout << "\t\t\t\t\t\t\t\t\t " << char(201);      for (int i = 0; i < 41; i++) { cout << char(205); }      cout << char(187) << endl;
    cout << "\t\t\t\t\t\t\t\t\t " << char(42) << "                                        " << char(42) << endl;
    cout << "\t\t\t\t\t\t\t\t\t " << char(42) << "  ELLACENT RESIDENTIAL CLEANING SERVICE  " << char(42) << endl;
    cout << "\t\t\t\t\t\t\t\t\t " << char(42) << "                                        " << char(42) << endl;
    cout << "\t\t\t\t\t\t\t\t\t " << char(200);      for (int i = 0; i < 41; i++) { cout << char(205); }      cout << char(188) << endl << endl;

    cout << "\t\t\t\t\t\t\tWelcome to EllaCent Residential Cleaning Service! We are extremely happy to" << endl;
    cout << "\t\t\t\t\t\t\tserve you, with the utmost professionalism, reliability, and respect. It is" << endl;
    cout << "\t\t\t\t\t\t\tour mission to serve and make the lives of our clients easier." << endl;
    cout << "\n\t\t\t\t\t\t\t\t\t    !! ENJOY OUR NEW YEAR'S 10% DEAL !!" << endl << endl;

    package = S_Package::findPackage();  //find all available packages to display
    cout << "\n\t\t\t\t\t\t\t\t\t* Packages Available * \n\n";
    cout << "\t\t\t*" << setw(3) << "ID" << setw(2) << "*" << setw(15) << "Name" << setw(4) << "*" << setw(11) << "Price (RM)" << setw(2) << "*" << setw(75)
    cout << "\n\t\t\t**********************************************************************************************************************************
    for (int i = 0; i < package.size(); i++)
    {
        cout << "\n\t\t\t*" << setw(3) << package[i].PackageID << setw(2) << "*" << setw(15) << package[i].P_Name << setw(4) << "*" << setw(11) << fixed <<
    }

    cout << "\n\t\t\t*Cleaning tools are not included in the package but can be added in the reservation.\n\t\t\t*Each package included 2 cleaners.\n\t\t\t

    cout << "\n\t\t\tWorking Days : Monday - Sunday" << endl;
    cout << "\n\t\t\tAny question about reservation or modification for your reservation, please contact below" << endl;
    cout << "\n\t\t\tContact : 012-3456789 \t\t\tEmail : ellacent@gmail.com" << endl;
```

Figure 16 : Function MainPage() with void return type

There are cases where void return type is used but it still needs to use parameter passed by the user in order for it proceed with the job. For example, NewPackage() function needs the set of values which is SS to use one of its attribute, StaffID, to be set as another variable that will be used in task that need the exact same value passed.

```
void NewPackage(S_Staff SS)  //add new package
{
    S_Menu menu;
    menu.header = "    New Package    ";
    menu.addOption("       Name*       ");
    menu.addOption("       Price*      ");
    menu.addOption("      Detail*      ");
    menu.addOption("       Hour*       ");
    menu.addOption("        Add        ");
    menu.addOption("        Back       ");
    menu.footer = "* - mandatory";

    S_Package SP;
    SP.StaffID = SS.StaffID;
    int option = 0;
    string tmp;

    while (1)
    {
        option = menu.prompt1(option);
        switch (option)
        {
        case -1:
            return;
            break;
        case 0:
            cout << "\n\t\t\t\t\t\t\t\tName " << char(175) << " ";
            getline(cin >> ws, SP.P_Name);
            menu.setValue(0, SP.P_Name);  //to display user's input
            break;
```

Figure 17 : Function NewPackage() with parameter S_Staff SS

On the other hand, there are functions that will return set of attributes. In the system, for example, function with return type S_Reservation will return attributes, like packageID and P_Price, of a package that it found.

```
S_Reservation FindPackage(S_Account SA, int id, S_Reservation SR)  //to find if PackageID requested exist
{
    S_Package package = S_Package::findPackage(id);  //using the id to find matching PackageID in package database

    if (package.PackageID == 0)
    {
        cout << "\n\t\t\tPackage not found";
        _getch();
        return SR;
    }

    SR.addpackage(package);  //if found, add the details about the PackageID into reservation database
    cout << "\n\n\t\t\t\t\t\t\t\tPackage added."; _getch();
    return SR;
}
```

Figure 18 : Function FindPackage() with return type S_Reservation

4.3 Selection

Selection is where the system will prompt user to make a decision and select one of the options available in order to proceed with the further process intended. One of the selection statements used in this system is else if where user has to decide if he or she want to enter as a staff or a customer. If user choose ‘c’, they will enter the system as customer. Same goes for ‘s’ for staff but if user entered other than ‘c’ or ‘s’, the system will automatically end.

```
cout << "\n\t\t\tSign-In/Register [C - Customer | S - Staff | X - exit] : ";
cin >> signin;

S_Menu menu;
menu.addOption("    Registration    ");
menu.addOption("       Login        ");
menu.addOption("        Back        ");

if (signin == 'C' || signin == 'c')
{
    menu.header = "      Customer     ";

    while (loop)
    {
        switch (menu.prompt2())
        {
        case -1:
            return 0;
            break;
        case 0:
            CustomerRegistration();  //to register new customer
            break;
        case 1:
            CustomerLogin();  //existing customer login
            break;
        case 2:
            loop = false; break;
        default: break;
        }
    }
}
else if (signin == 'S' || signin == 's')
{
    menu.header = "       Staff      ";

    while (loop)
```

Figure 19 : Selection of signin role (customer or staff) using else if

Switch statement is also used where each case of number has its own task. For example,

in this system, number of cases will be determined by the number of menu options added starting with case number 0. Lets say that the first option added is ‘Email’, then case number 0 would be for email. If user choose case number 0, the system will prompt user to enter their email.

```
menu.addOption("        Email       ");
menu.addOption("       Password     ");
menu.addOption("        Login       ");
menu.addOption("         Back       ");

S_Staff SS;
int option = 0;

while (1)
{
    option = menu.prompt1(option);
    switch (option)
    {
    case -1:
        return;
        break;
    case 0:
        cout << "\n\t\t\t\t\t\t\t\tEmail " << char(175) << " ";
        cin >> SS.S_Email;
        menu.setValue(0, SS.S_Email);
        break;
    case 1:
        cout << "\n\t\t\t\t\t\t\t\tPassword " << char(175) << " ";
        cin >> SS.S_Password;
        menu.setValue(1, HidePassword(SS.S_Password));
        break;
    case 2:
        if (SS.login())  //if user's inputs match with any data in database, it will return true and proceed
        {
            StaffHome(SS);
        }
        else
        {
```

Figure 20 : Selection of log in menu using switch case

4.4 Control

Control statement is where the system will redirect the flows of the tasks which if certain values satisfy its conditions, the system will execute statement contained in the control statement body or otherwise.

The most used control statement used in this system is checking the length or range of the value. For example, in Feedback() function, if user entered a rate and want to proceed entering the review, the system will first check if the rate is not less than zero and also not more than five (R_Rate >= 0 && R_Rate <= 5). If the value satisfy, it will proceed to insert the review.

```
cout << "\n\t\t\t\t\t\t\t\tAre you sure to enter this review ? [Y - yes | X - no/back] " << char(175) << " ";
char leavereview;
cin >> leavereview;
if (leavereview == 'Y' || leavereview == 'y')
{
    if (SS.R_Rate >= 0 && SS.R_Rate <= 5)
    {
        SS.addreview();
        cout << "\n\t\t\t\t\t\t\t\tDone"; _getch();
        loop = false;
    }
    else
    {
        cout << "\n\t\t\t\t\t\t\t\tPlease enter the rate"; _getch();
    }
}
```

Figure 21 : Control check of R_Rate value using else if and operator

In other ways, a do-while loop is used to make sure that is the user want to exit the loop, they have to make sure that the password length obey the rule where it must be more than five and also less than 13 (tmp.length() <= 5 || tmp.length() >= 13).

```
do
{
    cout << "\n\t\t\t\t\t\t\t\tPassword " << char(175) << " ";
    cin >> tmp;

    if (tmp.length() > 5 && tmp.length() < 13)  //make sure that password length is more than 5 digits and lest than 13 digits
    {
        SS.S_Password = tmp;
        menu.setValue(3, SS.S_Password);
    }
    else
        cout << "\n\t\t!!Password must be 6 to 12 character!!" << endl;
} while (tmp.length() <= 5 || tmp.length() >= 13);
```

Figure 22 : Control check of S_Password length using do-while and operator

4.5 Error Handling

Error handling is where programmer has to apply detection and resolution of programming. Its aim is to prevent any error from occurring including human error that will affect the whole system.

One of the error handling applied in this system is when both customers and staffs insert invalid password for registration. If user attempts to insert password that is less than 6 digits or more than 12 digits, an error message will appear and prompt user until user enter a valid password.

```
do
{
    cout << "\n\t\t\t\t\t\t\t\tPassword " << char(175) << " ";
    cin >> tmp;

    if (tmp.length() > 5 && tmp.length() < 13)  //make sure that password length is more than 5 digits and lest than 13 digits
    {
        SS.S_Password = tmp;
        menu.setValue(3, SS.S_Password);
    }
    else
        cout << "\n\t\t!!Password must be 6 to 12 character!!" << endl;

} while (tmp.length() <= 5 || tmp.length() >= 13);
```

Figure 23 : Invalid S_Password error handling

Another error handling applied is when customer enter rate exceeding five for review purpose. If this occur, the system will display a message and not insert the rate entered by customer into database.

```
if (SS.R_Rate >= 0 && SS.R_Rate <= 5)
{
    SS.addreview();
    cout << "\n\t\t\t\t\t\t\t\tDone"; _getch();
    loop = false;
}
else
{
    cout << "\n\t\t\t\t\t\t\t\tPlease enter the rate"; _getch();
}
```

Figure 24 : Invalid R_Rate error handling

In a case where the date entered by user is either in the past or doesn't even exist, error handling is applied. The date entered by user will have to go through a function that will check the validity of the date. If date is not valid, the date entered will not be inserted into database.

```
cout << "\n\t\t\t\t\t\t\t\tDay " << char(175) << " ";
cin >> day;

cout << "\n\t\t\t\t\t\t\t\tMonth " << char(175) << " ";
cin >> month;

cout << "\n\t\t\t\t\t\t\t\tYear " << char(175) << " ";
cin >> year;

tempdate = to_string(day) + "-" + to_string(month) + "-" + to_string(year);

if (checkdate(tempdate))  // to see if date entered surpass current date and exist
{
    SR.R_Date = tempdate;  //rearrange the date inputs
}
else
    cout << "\n\t\t\t\t\t\t\t\tDate entered is invalid."; _getch();
```

Figure 25 : Invalid date error handling

This function will check the day, month and year of the input which makes sure that the date is not in the past and actually exist. For example, 30-2-2024 is not valid as it doesn't exist while 27-7-2023 is also not valid because it is in the past. If the date is valid, the function will return true.

```
bool CheckDate(const string& dateinput)
{
    std::tm inputdate = {};
    std::istringstream ss(dateinput);
    char dash;  // to store the dash between day, month, and year
    ss >> inputdate.tm_mday >> dash >> inputdate.tm_mon >> dash >> inputdate.tm_year;

    // Adjust the year and month to match the structure representation
    inputdate.tm_year -= 1900;
    inputdate.tm_mon -= 1;

    // Get the current date
    std::tm currentDate;
    auto currentTime = std::chrono::system_clock::to_time_t(std::chrono::system_clock::now());
    localtime_s(&currentDate, &currentTime);

    // Convert both dates to time points for easy comparison
    auto currentTimePoint = std::chrono::system_clock::from_time_t(currentTime);
    auto userInputTimePoint = std::chrono::system_clock::from_time_t(std::mktime(&inputdate));

    //check if user's entered date is exceeding current date & time, and is actually exist. If both satisfied, return true.
    return (userInputTimePoint > currentTimePoint) && (inputdate.tm_year == year - 1900) && (inputdate.tm_mon == month - 1) && (inputdate.tm_mday ==
}
```

Figure 26 : Function included in invalid date error handling

CHAPTER 5: CONCLUSION

5.1 Constraints

The system is not really user-friendly since the system use CLI (command line interface) which we shouldn't assume that everyone knows how to understand it. Although the system have multiple functionality, it is still not complete if compared with actual business systems out there. For example, one of the functionalities that should be included in the system is cancelling or modifying the reservation, but these jobs cannot be applied in the system since there are policies that should be obeyed, for instance, user cannot edit or delete a reservation if current time is less than 24 hours from service delivery time.

5.2 Future Improvements

To improve this system, it would be better to design the interface that is easier for user to understand so user will not be confused and misunderstand the flow of the system. It would also be nice if user especially customer could perform every task in the system without having to do additional task outside of system for example the modification of reservation that follow the company's policies. Furthermore, it would be better if everyone who are actually involved in this service could also access the system, for example cleaners as they also have important roles for this business. Lastly, it would be preferable if customer can perform payment before or after the service is done to ease the customer with the process.