



TRAVGUIER

Table of content

- ✓ Introduction
- ✓ Problem Statement & solution
- ✓ Methodology
- ✓ Pair programming
- ✓ Requirements
- ✓ Backlog
- ✓ Iterations
- ✓ Test-driven development
- ✓ Analysis
- ✓ Design

- ✓ Implementation
- ✓ Refactoring
- ✓ Organization and flow
- ✓ Lesson learn
- ✓ Challenges
- ✓ Tools
- ✓ Future work
- ✓ Conclusions



Introduction

The project name is **TRAVGUIER** we choose it according to **(Travil+Guide+Saver)**, we aim to build an application to help customers to make reservations for their trips in a simple way with a specific budget.

Problem Statement

Many sites and applications have started promoting tourist and recreational places to attract visitors, but the customer may find it difficult to choose due to the multiplicity of price changes and the difficulty of booking according to a specific budget in a specific space within a specific period.

Proposed Solution

After analysing the problem and gathering requirements, we found that the application can help the user to get trip plans according to a budget, region and time pre-set by the user, can also be booked, and can be modified later reservation.

Methodology

We use Extreme programming (XP) to develop our project. It's an agile envelopment method that doesn't need too much paperwork, documentation, or complicated meetings instead of that it's more focused on the programming of what needs to be done.

The benefits of using this approach are:

- Improve software quality
- Improve productivity

Pair programming

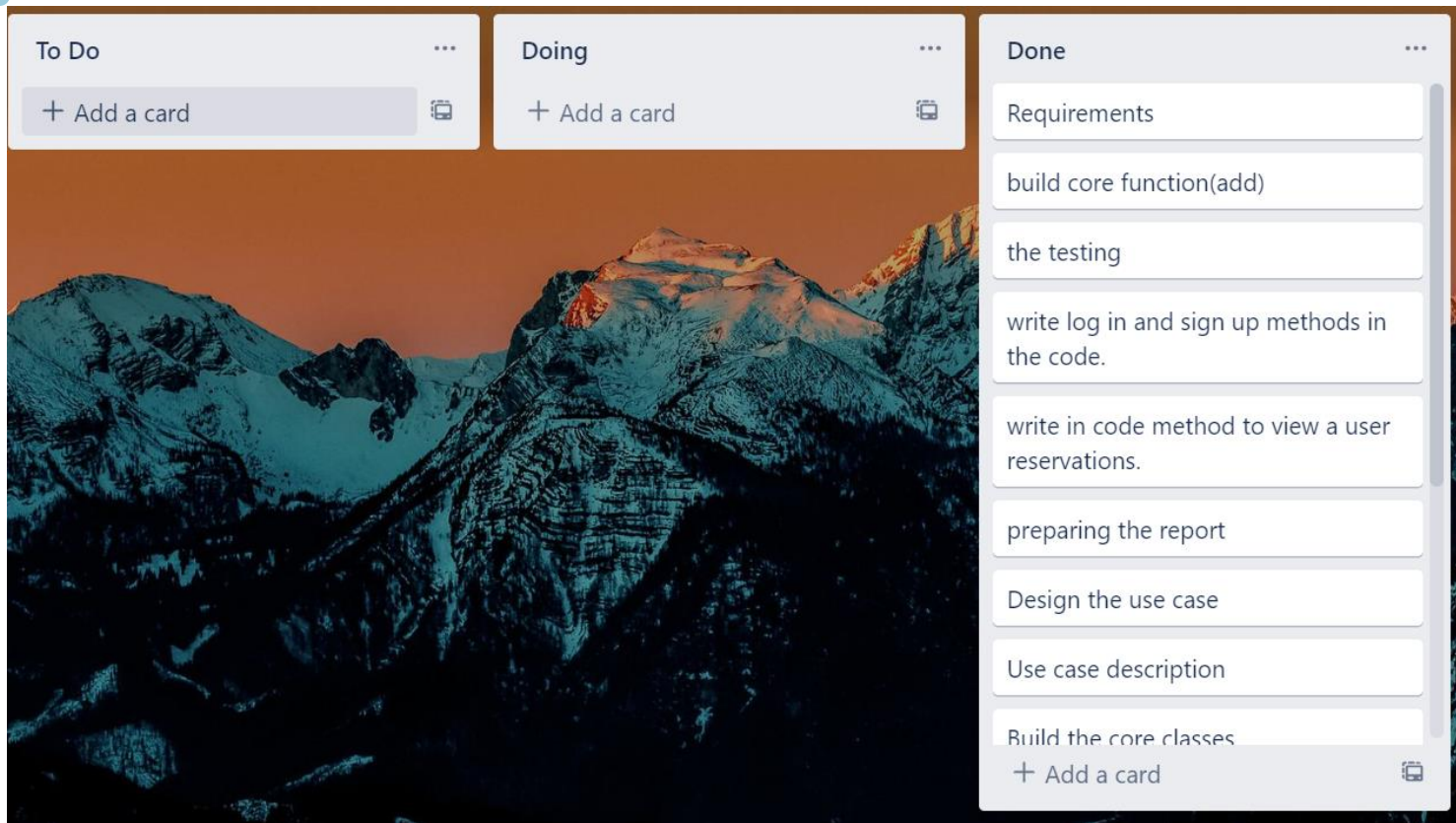
- Pair programming involves programmers working in pairs and developing code together. This helps develop common ownership of code and spreads knowledge across the team. It serves as an informal review process as each line of code is looked at by more than 1 person. It encourages refactoring as the whole team can benefit from improving the system code.

The use of pair programming was very useful in our project application to reduce errors and try to write the code in a better and simpler way. ●

Requirements

- A user can create a new account.
- A user can log in to my account.
- A user can add new reservations.
- A user can view their reservations.
- A user can delete existing reservations.
- A user can edit their reservations.

Trello



Iterations

Iteration 1: we start with the first core function (Add reservation) and do the required analysis and start coding in pair programming then test and refactor the code for this core function in this iteration.

Iteration 2: we start with the second core function (Edit reservation) and do the required analysis and start coding in pair programming then test and refactor code for this core function in this iteration.

Iteration 3: we start with the third core function (View reservation) and do the required analysis and start coding in pair programming then test and refactor code for this core function in this iteration.

Iteration 4: we start with the fourth core function (Delete reservation) and do the required analysis and start coding in pair programming then test and refactor code for this core function in this iteration.

Test-driven development

Test-driven development (TDD) is an approach to program development in which you inter-leave testing and code development.

Benefits:

- Code coverage.
- Regression testing.
- Simplified debugging.
- System documentation.

Test case discription

(Add reservation)

Test 1: Add reservation

Input:

1. A string representing the destination.
2. A number representing the budget.
3. A string representing the airline.

Tests:

1. Test whether the field in completed correctly

Output:

Reservation confirmation message or error massage.

Test case description (Edit reservations)

Test 2: Edit reservation

Input:

1. A number representing ID of the selected reservation
2. A number representing which part of the reservation change

Tests:

5. Test if the reservation exists
6. Test if the set-up of the information is done correctly

Output:

Confirmation of changes message

Test case description (View reservations)

Test 3: View reservation

Input:

1. A number representing the number of process

Tests:

1. Test if the user has reservations

Output:

Display the existing reservations

Test case discription (Delete reservations)

Test 4: Delete reservation

Input:

1. A number representing ID of the selected reservation

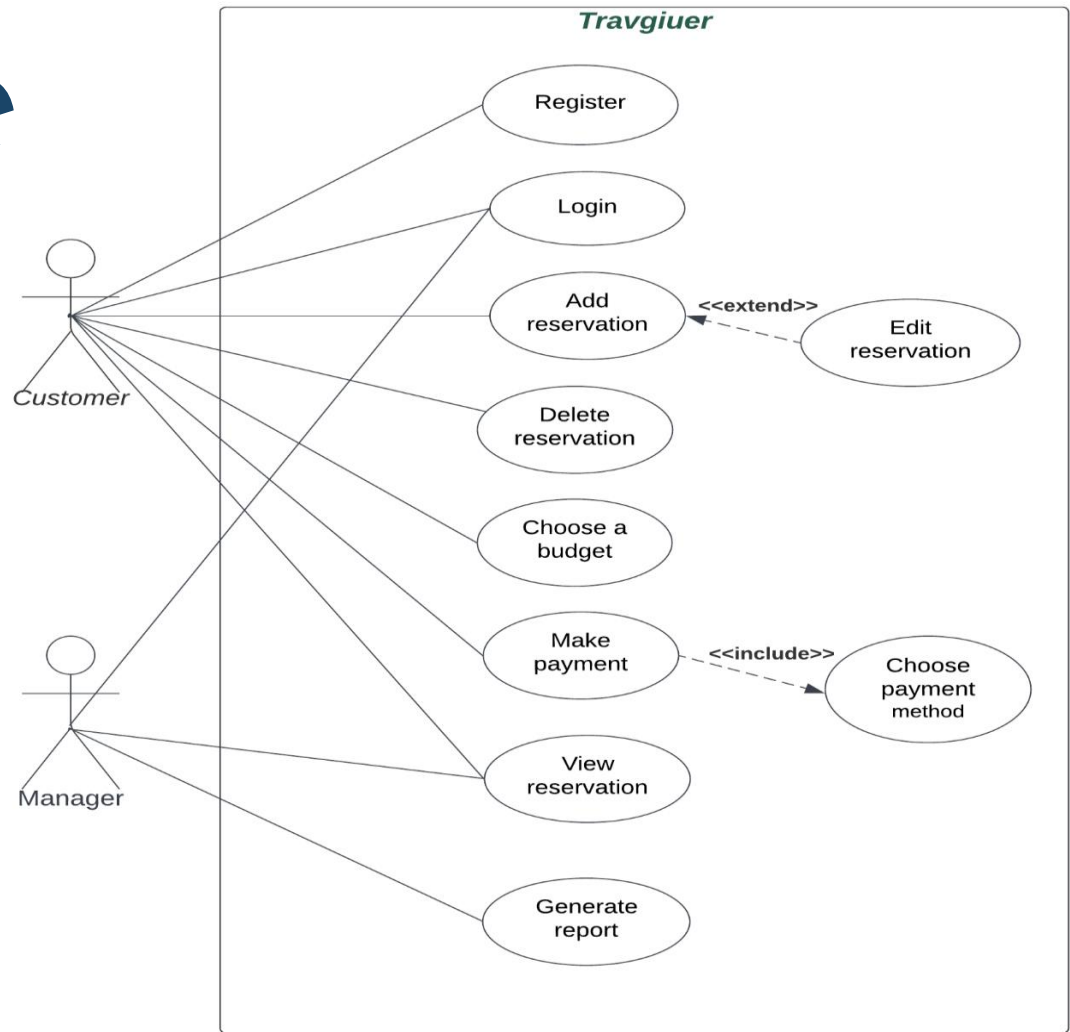
Tests:

7. Test if the reservation exists
8. Test if the setup of the information in done correctly

Output:

Confirmation of changes message

Use Case



• Use case description • (Add Reservation)

Use Case: Add reservation.
Primary Actor: Customer.
Level: Kite (Summary).
Stakeholders: Customer.
Precondition: login to the account.
Minimal Guarantee: show all available reservations.
Success Guarantee: add a reservation to the cart.
Trigger: The user will click add button.
Main Success Scenario: 1. System display all reservations. 2. User adds the desired reservation.
Extensions: 1a. System didn't add the reservation successfully. 1a1. System display error message. 1a2. The user reloads the page. 1a3. The user adds the desired reservation again.

• Use case description • (Edit Reservation)

Use Case: Edit reservation.
Primary Actor: Customer.
Level: Kite (Summary).
Stakeholders: Customer.
Precondition: have a reservation in the cart.
Minimal Guarantee: nothing changed.
Success Guarantee: edit the selected reservation.
Trigger: The user will open the cart.
Main Success Scenario: 1. User opens the cart. 2. System display all reservations in the cart. 3. User edit the desired reservation.
Extensions: 1a. System doesn't apply the edit successfully. 1a1. Changes do not appear to the user. 1a2. The user edits the desired reservation again.

• Use case description • (Delete Reservation)

Use Case: Delete reservation.

Primary Actor: Customer.

Level: Kite (Summary).

Stakeholders: Customer.

Precondition: have a reservation in the cart.

Minimal Guarantee: delete nothing.

Success Guarantee: delete a reservation from the cart.

Trigger: The user will open the cart.

Main Success Scenario:

1. User opens the cart.
2. System display all reservations in the cart.
3. User deletes the desired reservation.

Extensions:

- 1a. System didn't delete the reservation successfully.
 - 1a1. System display error message.
 - 1a2. The user reloads the page.
 - 1a3. The user deletes the desired reservation again.

• Use case description • (View Reservation)

Use Case: View reservation.

Primary Actor: Customer, Manager.

Level: Kite (Summary).

Stakeholders: Customer, Manager.

Precondition: make a reservation before.

Minimal Guarantee: display all the reservations.

Success Guarantee: display all reservation with details.

Trigger: choose the desired reservation to display details.

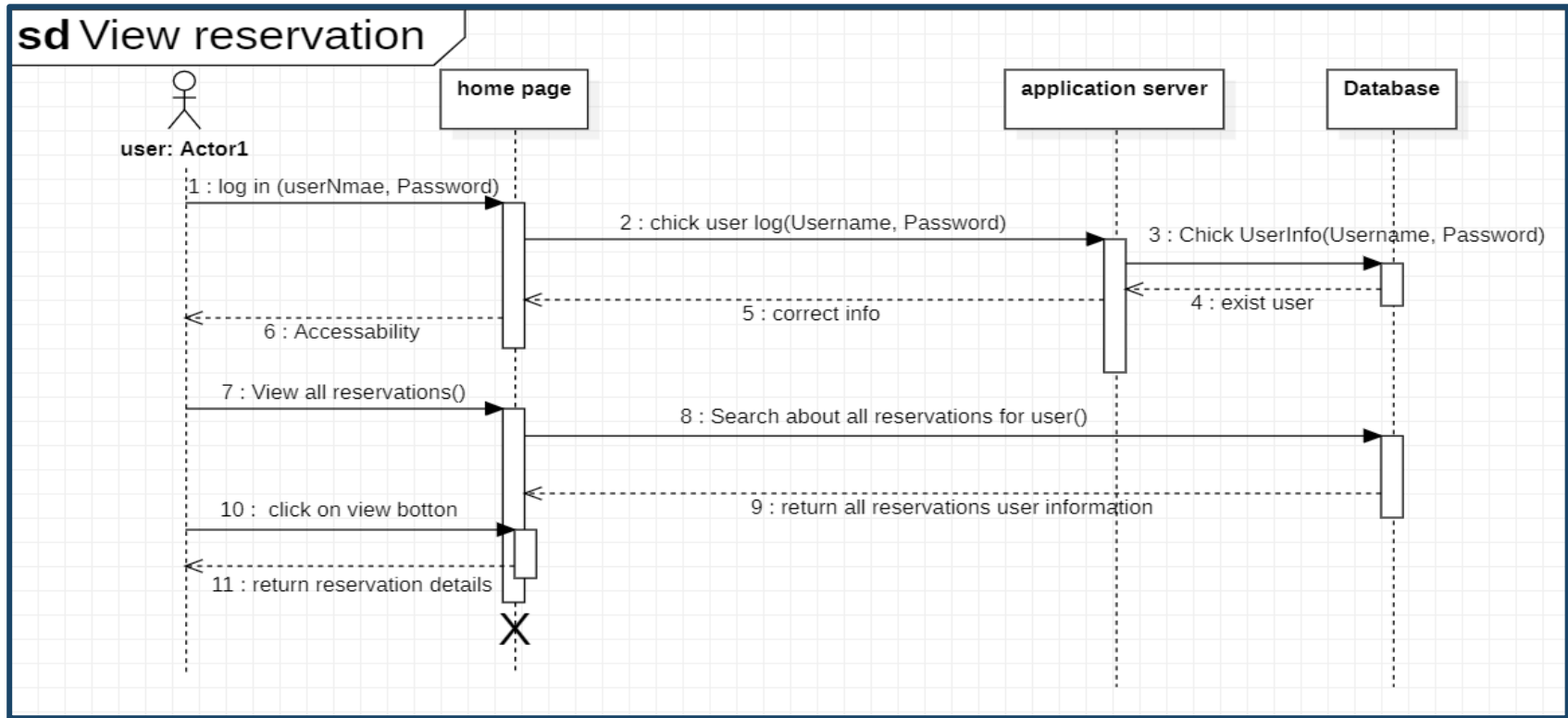
Main Success Scenario:

1. User opens the current reservations.
2. System display all current reservations.
3. User chooses the desired reservation.
4. System display reservation details.

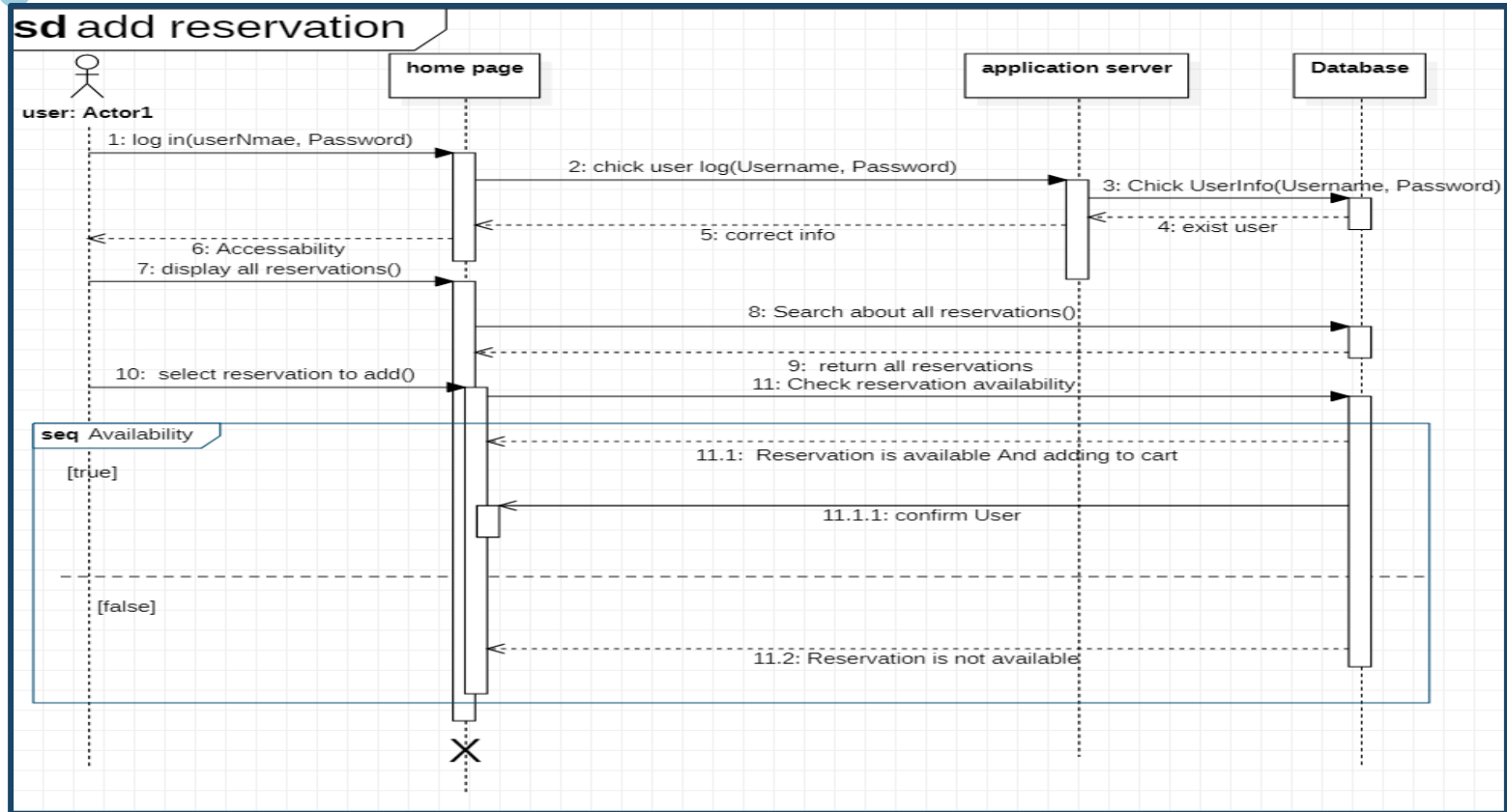
Extensions:

- 1a. System displays the reservation with missing details.
1a1. The user reloads the page.

Sequence Diagram (View reservation)

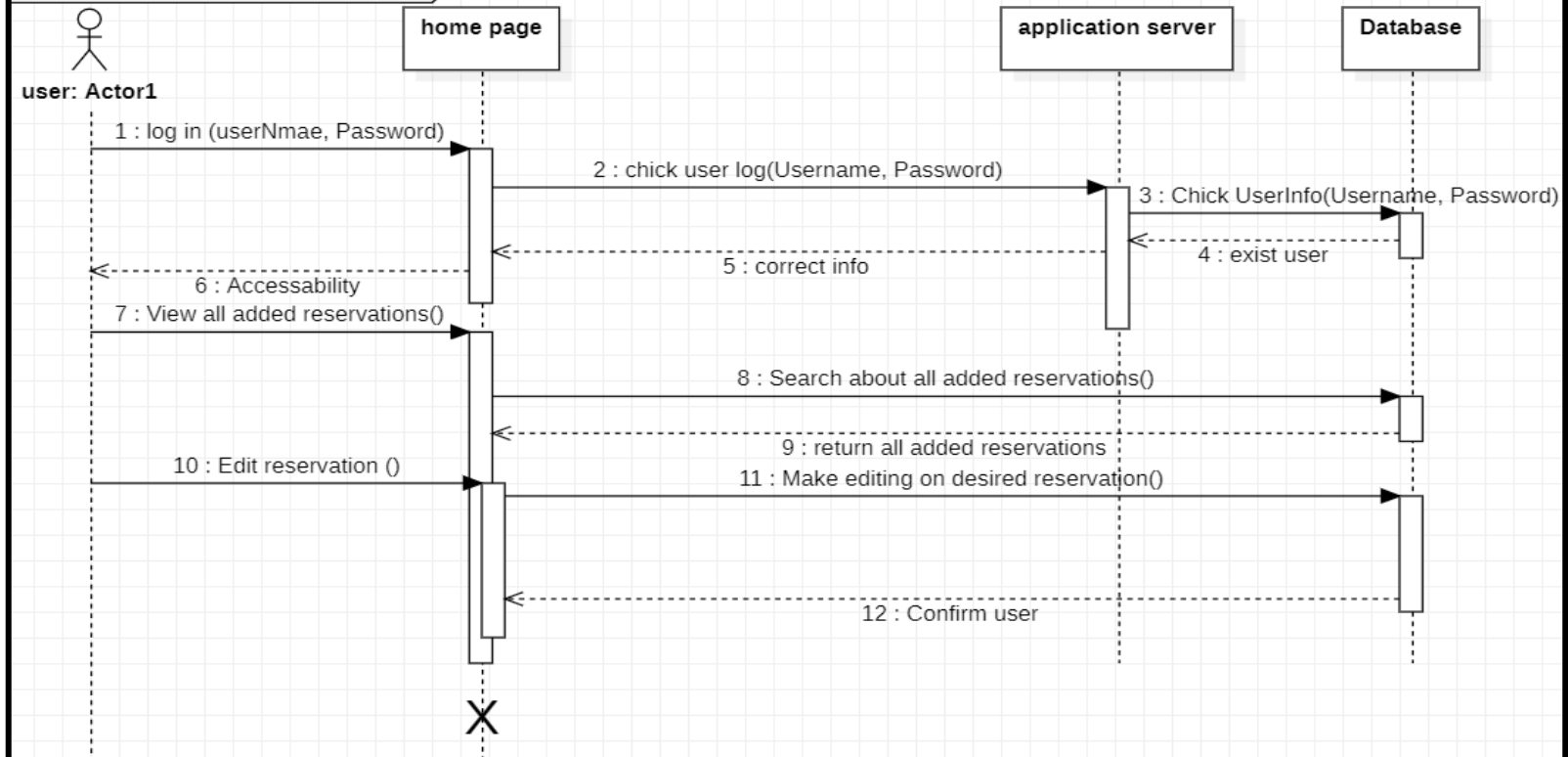


Sequence Diagram (Add reservation)

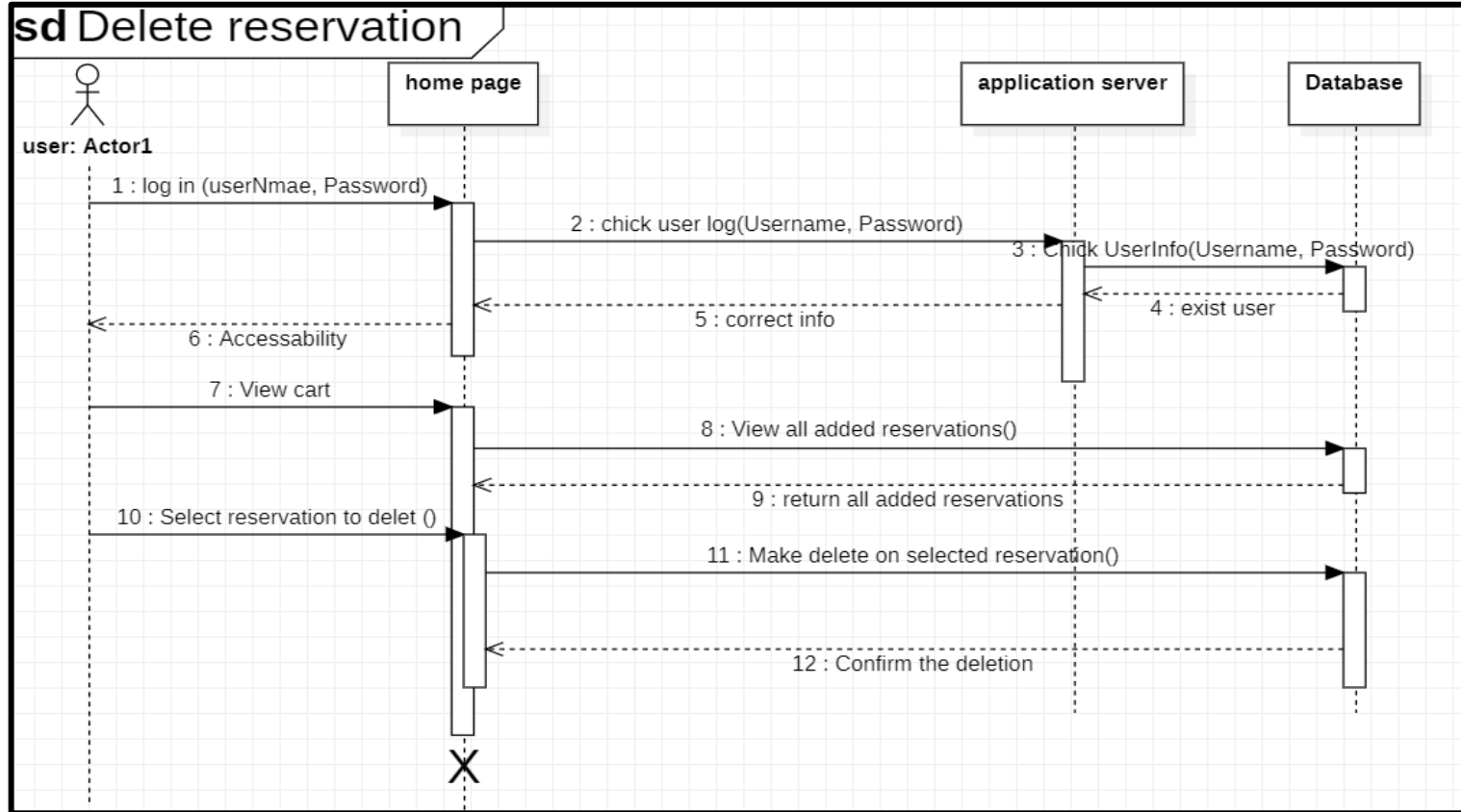


Sequence Diagram (Edit reservation)

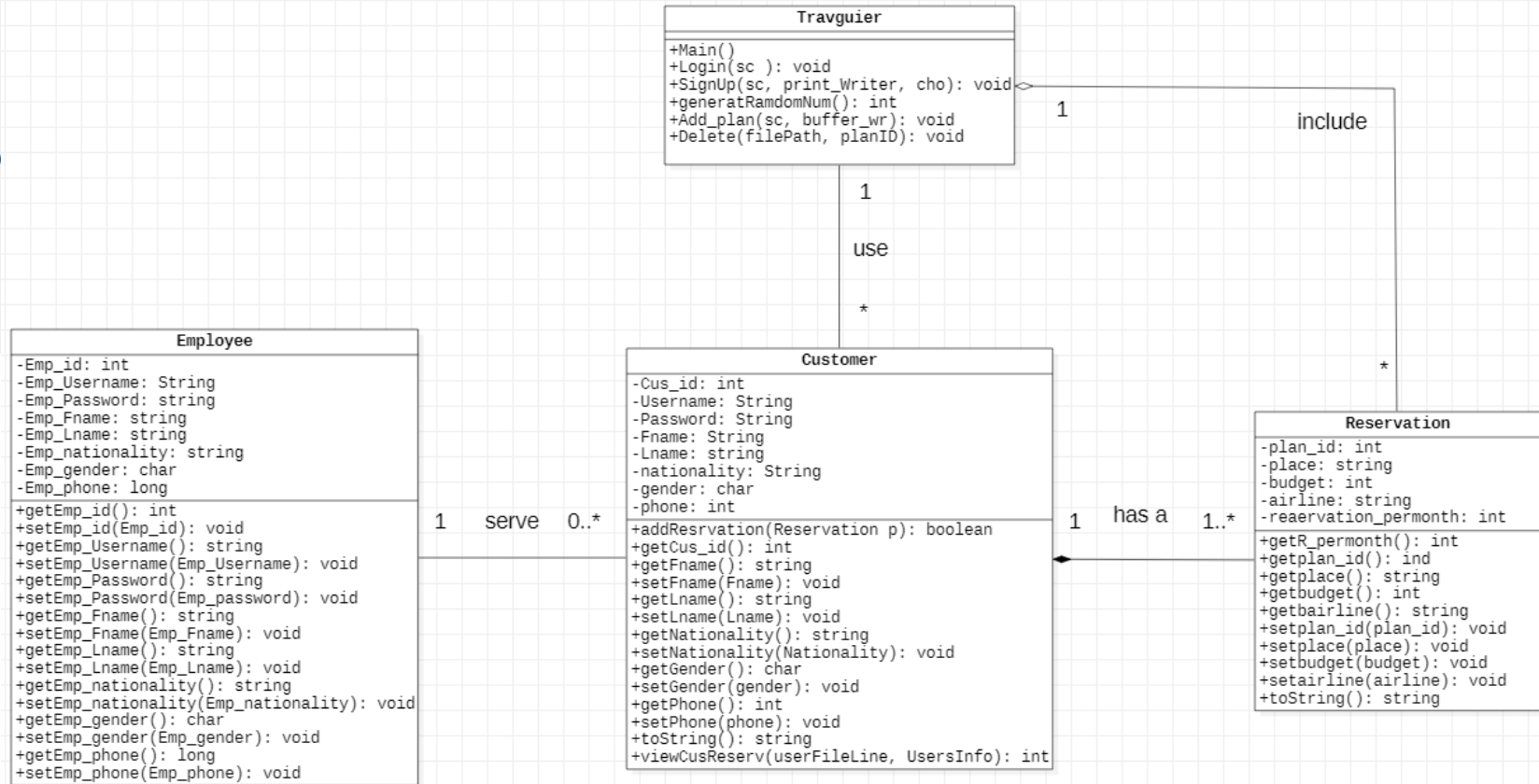
sd Edit reservation



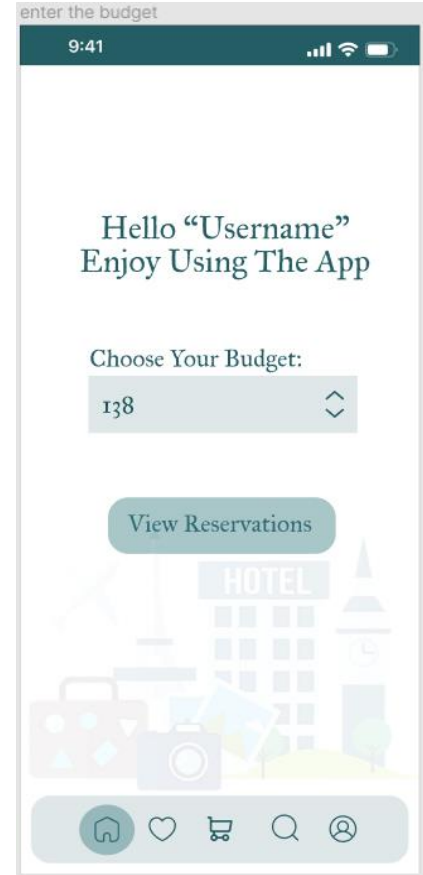
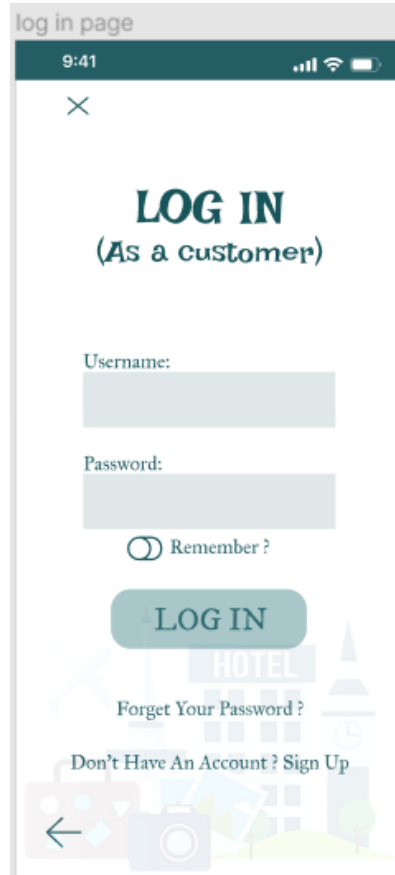
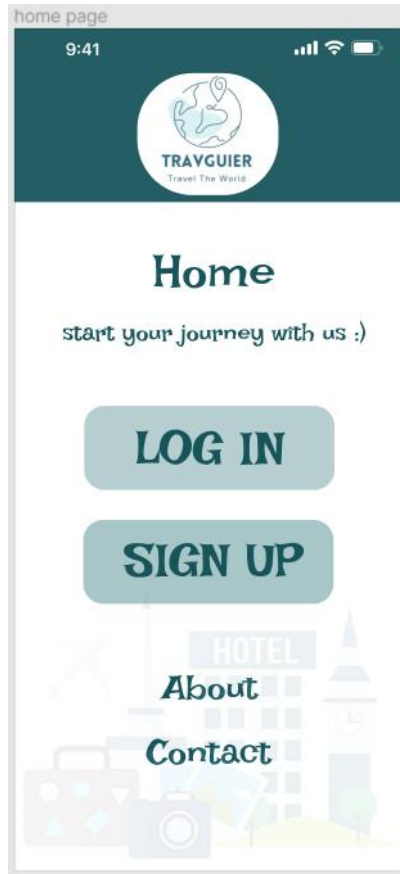
Sequence Diagram (Delete reservation)



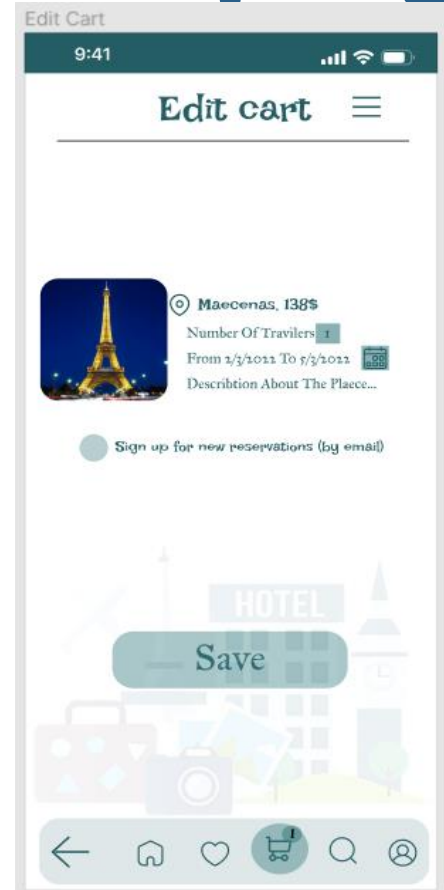
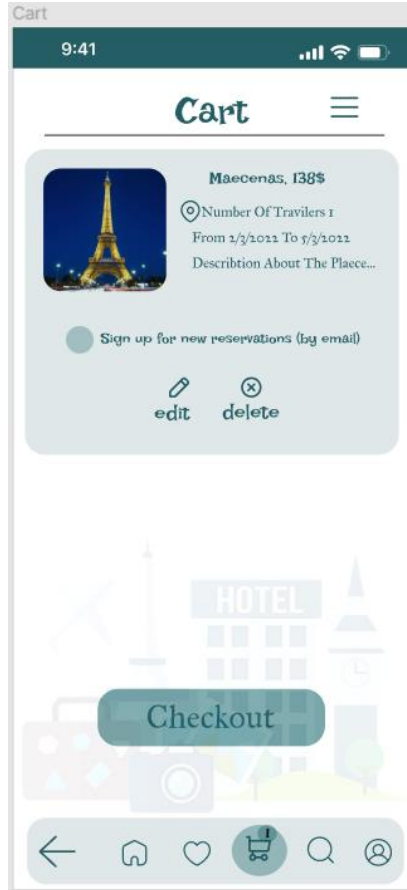
Class Diagram



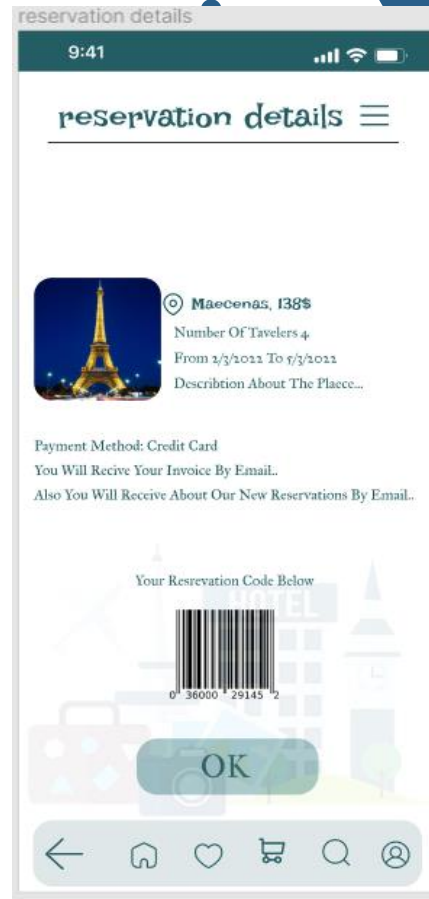
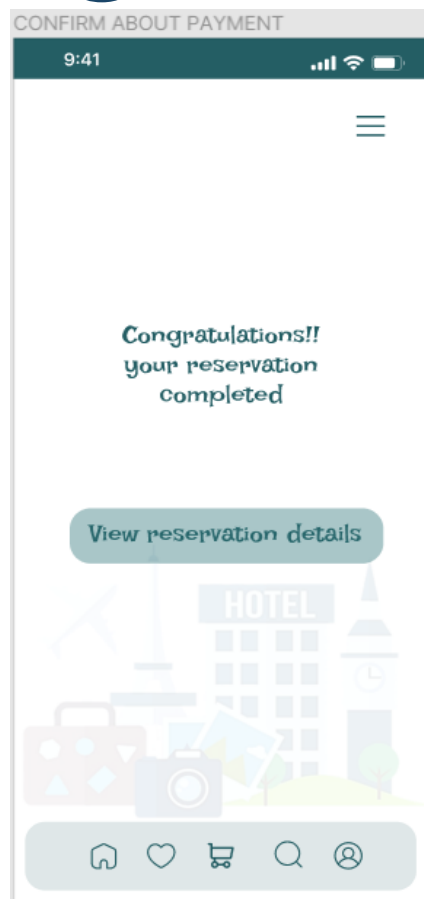
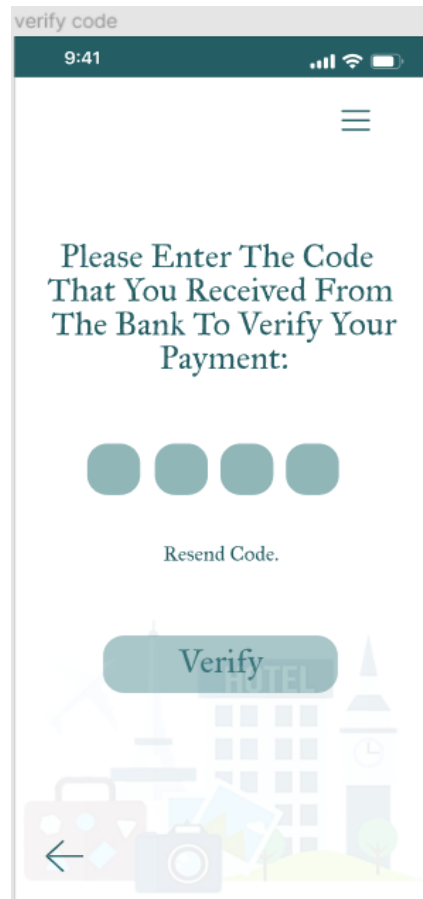
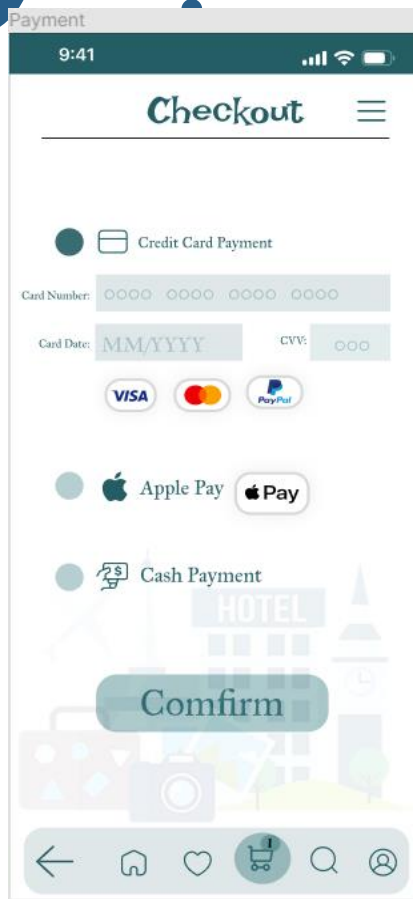
Design



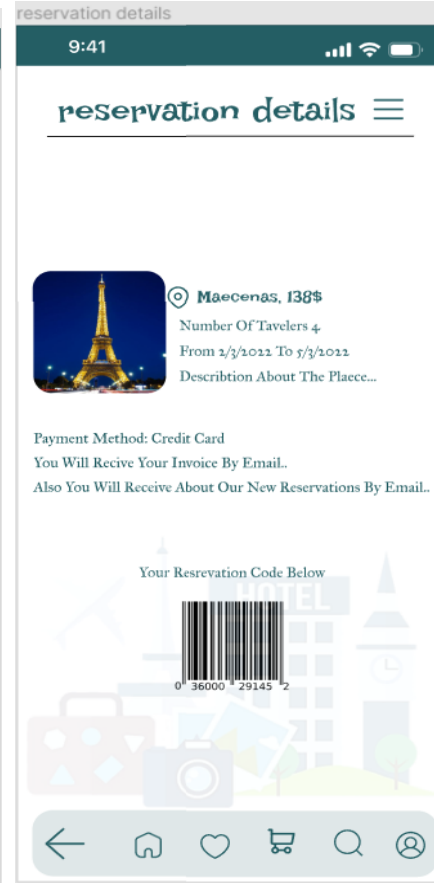
Design



Design



Design



Database

- Cus_info: customer info.
- Cust_info2: copy of customer info.
- Emp_info: employee info.
- New_Cus_info: new customer info.
- New_Emp_info: new employee info.
- Reservation_info: reservations info.

Implementation Code(Main)

```
File Cus_file_info = new File("New_Cus_info.txt");
File Emp_file_info = new File("New_Emp_info.txt");
File Log_cus = new File("Cus_info.txt");
File Log_cus2 = new File("Cus_info2.txt");
File Log_emp = new File("Emp_info.txt");
Scanner sc = new Scanner(System.in);
PrintWriter Writer_cus_info = new PrintWriter(Cus_file_info);
PrintWriter Writer_emp_info = new PrintWriter(Emp_file_info);
```

```
BufferedWriter wr = new BufferedWriter(new FileWriter("Reservation_info.txt", true));
```

```
//boolean m = true;
```

```
while (true) {
```

```
    initialMenu();
```

```
    int user_choise = sc.nextInt();
```

```
    switch (user_choise) {
```

```
        case 1: {
```

```
            System.out.print("Please enter username : ");
            String u_name = sc.next();
```

```
            System.out.print("Please enter Password : ");
            String Pword = sc.next();
```

```
            System.out.print("Please enter your First name : ");
            String FName = sc.next();
```

```
            System.out.print("Please enter your Last name : ");
            String Lname = sc.next();
```

```
            System.out.print("Please enter your Nationality Number: ");
            String nationality = sc.next();
```

```
            System.out.print("Please enter your Gender (M for Male, F for Female): ");
            char gen = sc.next().charAt(0);
```

```
            System.out.print("Please enter your Phone number : ");
            String phone = sc.next();
```

```
public static void initialMenu() {
```

```
    System.out.println("----- Welcome to TRAVGUIER System -----");
    System.out.println("-----");
    System.out.println("1. SignUp Create A New Account.");
    System.out.println("2. Login To Your Account.");
    System.out.println("9. Exit.");
    System.out.println("-----");
    System.out.print("Write The Number Of Process : ");
```

Implementation Code(Main)

```
String phone = sc.next();

int ran = generatRandomNum();

Scanner Read_Log_cus3 = new Scanner(Log_cus);
Scanner Read_Log_cus5 = new Scanner(Log_cus);

custm.SignUp(sc, Read_Log_cus5, Read_Log_cus3, Writer_cus_info, ran, u_name, Pword, Fname, Lname, nationality, gen
Read_Log_cus3.close();
Read_Log_cus5.close();

System.out.print(" adding is done !");
break;
}
case 2: {

System.out.print("Are you a customer(1) or an employee(2) ? ");
int userType = sc.nextInt();

System.out.print("Please enter your username : ");
String UserName = sc.next();
System.out.print("Please enter your Password : ");
String Pword = sc.next();
```


Implementation Code(Main)

```
public int CusSeachExist(Scanner numLines, Scanner raedInfo, String UserName, String Password) {  
  
    int lineNum = numOfLines(numLines);  
    int there = 0;  
    //int there2 = 0;  
    String[][] info = new String[lineNum][];  
  
    for (int i = 0; i < info.length; i++) {  
        String read = raedInfo.nextLine();  
        info[i] = read.split(",");  
        if ((info[i][0]).equalsIgnoreCase(UserName)) {  
            if (info[i][1].equalsIgnoreCase(Password)) {  
                there++;  
                break;  
            } else {  
                System.out.println("Wrong Password ! ");  
                //there2++;  
                break;  
            }  
        }  
  
        }else if (raedInfo.hasNext()){  
            continue;  
        }else{  
            System.out.println("There Are no account by : " + UserName + " username");  
        }  
    }  
  
    return there;  
}
```

Implementation Run(Main)

```
----- Welcome to TRAVGUIER System -----  
-----  
1. SignUp Create A New Account.  
2. Login To Your Account.  
9. Exit.  
-----  
Write The Number Of Process : 2  
Are you a customer(1) or an employee(2) ? 1  
Please enter your username : Almas.nogali  
Please enter your Password : Almas200  
  
----- Welcome to TRAVGUIER System -----  
-----  
3. Add reservation info.  
4. View reservation info.  
5. Edit reservation info.  
6. Delete reservation info.  
9. Exit.  
-----  
Write The Number Of Process :
```

Implementation Code(Add)

```
while (true) {  
    processMenu();  
    int user_choise2 = sc.nextInt();
```

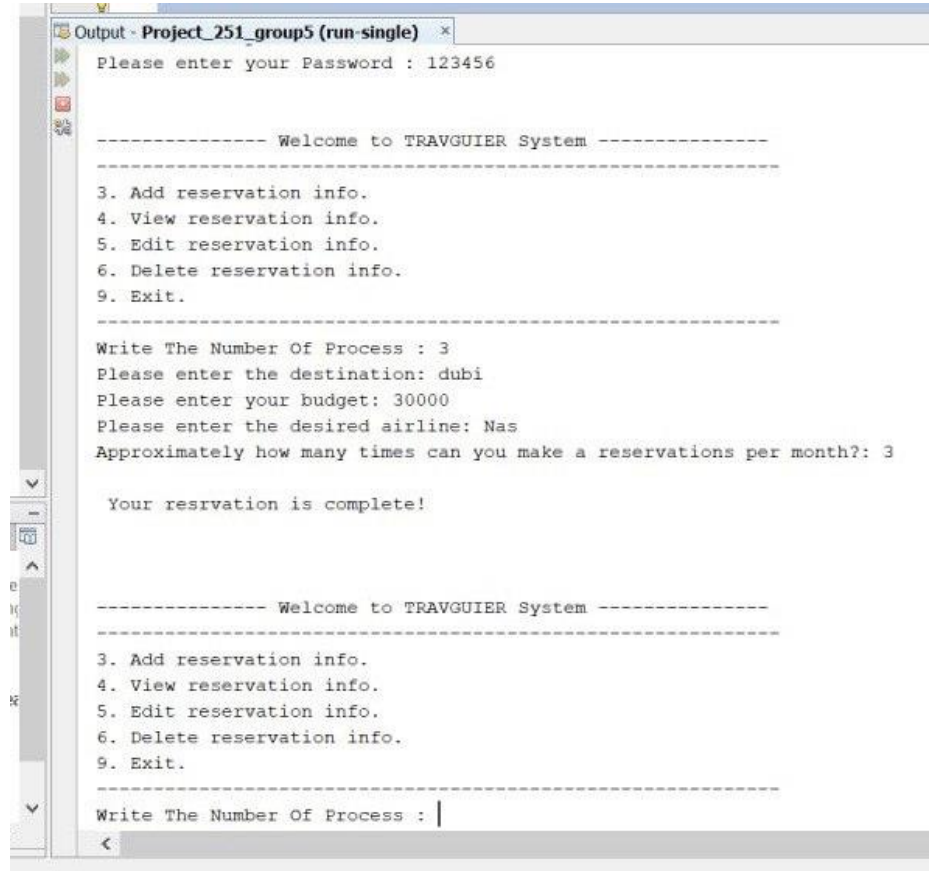
```
    switch (user_choise2) {
```

```
        case 3: {  
            //Add reservations  
            Add_plan(sc, wr, custm);  
  
            break;  
        }  
        case 4: {  
            //View reservations  
            Scanner Read_Log_cus7 = new Scanner(Log_cus);  
            Scanner Read_Log_cus8 = new Scanner(Log_cus);  
            custm.viewCusReserv(Read_Log_cus7, Read_Log_cus8);  
            Read_Log_cus7.close();  
            Read_Log_cus8.close();  
  
            break;  
        }  
    }
```

```
public static void processMenu() {  
  
    System.out.println("\n\n----- Welcome to TRAVGUIER System -----");  
    System.out.println("-----");  
    System.out.println("3. Add reservation info.");  
    System.out.println("4. View reservation info.");  
    System.out.println("5. Edit reservation info.");  
    System.out.println("6. Delete reservation info.");  
    System.out.println("9. Exit.");  
    System.out.println("-----");  
    System.out.print("Write The Number Of Process : ");  
  
}
```

```
public static void Add_plan(Scanner sc, BufferedWriter wr, Customer cu2) throws IOException {  
  
    System.out.print("Please enter the destination: ");  
    String place = sc.next();  
    System.out.print("Please enter your budget: ");  
    int budget = sc.nextInt();  
    System.out.print("Please enter the desired airline: ");  
    String airline = sc.next();  
  
    System.out.print("Approximately how many times can you make a reservations per month?: ");  
    int r_perMonth = sc.nextInt();  
  
    int plan_id = generatRandomNum();  
  
    Reservation plan = new Reservation(plan_id, place, budget, airline, r_perMonth);  
  
    cu2 = new Customer();  
  
    cu2.addReservation(plan);  
  
    if (cu2.addReservation(plan) == true) {  
        while (true) {  
            System.out.println("\n Your rsrvation is complete!\n");  
            wr.write(plan.toString());  
            break;  
        }  
    } else {  
        System.out.println("You did not enter the budget or how many reservation you want"  
            + " !,Please enter again: \n");  
        Add_plan(sc, wr, cu2);  
    }  
    wr.close();  
}
```

Implementation Run(Add)



```
Output - Project_251_group5 (run-single) x
Please enter your Password : 123456

----- Welcome to TRAVGUIER System -----
-----
3. Add reservation info.
4. View reservation info.
5. Edit reservation info.
6. Delete reservation info.
9. Exit.
-----

Write The Number Of Process : 3
Please enter the destination: dubi
Please enter your budget: 30000
Please enter the desired airline: Nas
Approximately how many times can you make a reservations per month?: 3

Your resrvation is complete!

----- Welcome to TRAVGUIER System -----
-----
3. Add reservation info.
4. View reservation info.
5. Edit reservation info.
6. Delete reservation info.
9. Exit.
-----

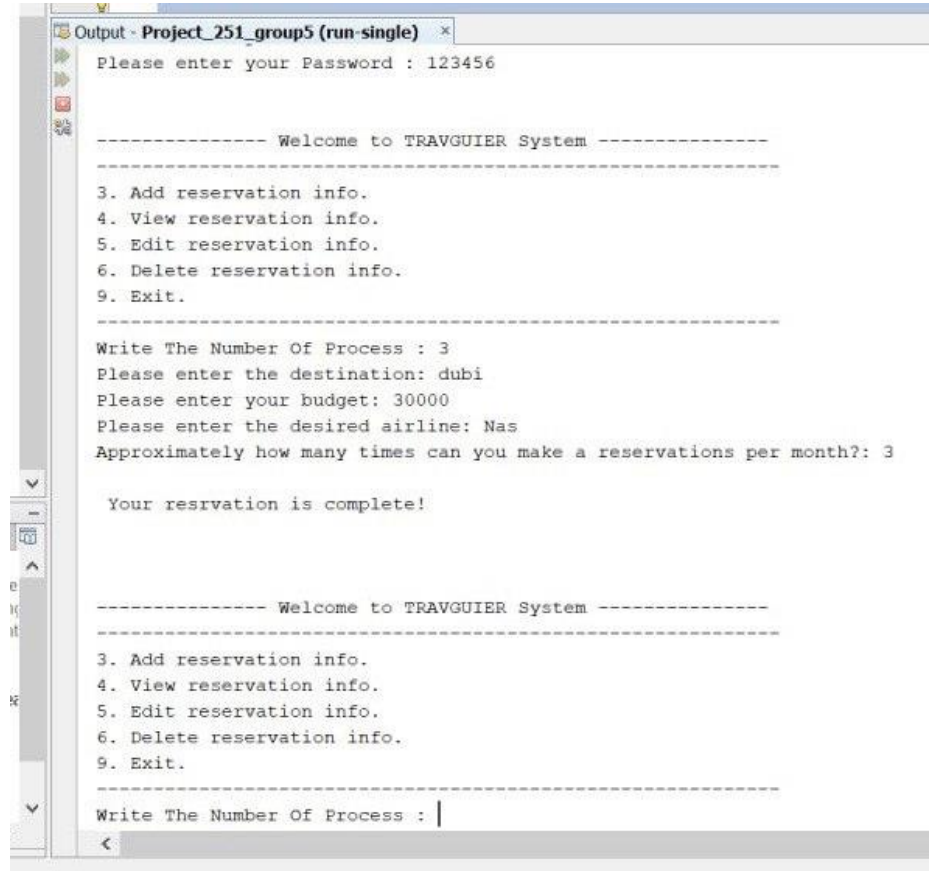
Write The Number Of Process : |
```

Implementation Code(View)

```
case 4: {  
    //View reservations  
    Scanner Read_Log_cus7 = new Scanner(Log_cus);  
    Scanner Read_Log_cus8 = new Scanner(Log_cus);  
    custm.viewCusReserv(Read_Log_cus7, Read_Log_cus8);  
    Read_Log_cus7.close();  
    Read_Log_cus8.close();  
  
    break;  
}
```

```
public int viewCusReserv(Scanner userFileLine, Scanner UsersInfo) {  
  
    int numOfReserv = 0;  
    int lineNum = numOfLines(userFileLine);  
    String[][] info = new String[lineNum][];  
    for (int i = 0; i < info.length; i++) {  
  
        String read = UsersInfo.nextLine();  
        info[i] = read.split(",");  
  
        if (((info[i][0]).equals(this.Username)) && ((info[i][1]).equals(this.Password))) {  
  
            System.out.println("*****");  
            System.out.printf("\n Username :%s \n", this.Username);  
            System.out.printf(" Password :%s \n", this.Password);  
            System.out.printf(" First name :%s \n", info[i][2].toString());  
            System.out.printf(" Last name :%s \n", info[i][3].toString());  
            System.out.printf(" National number :%s\n", info[i][4].toString());  
            System.out.printf(" Gender :%s \n", info[i][5].toString());  
            System.out.printf(" Phone number :%s \n", info[i][6].toString());  
            System.out.printf(" Customer ID :%s \n", info[i][7].toString());  
            System.out.printf(" Plan ID :%s \n", info[i][8].toString());  
            System.out.printf(" Destination :%s \n", info[i][9].toString());  
            System.out.printf(" Budget :%s \n", info[i][10].toString());  
            System.out.printf(" Airplane :%s \n\n", info[i][11].toString());  
            System.out.println("*****");  
  
            numOfReserv++;  
        }  
    }  
    return numOfReserv;  
}
```

Implementation Run(View)



```
Output - Project_251_group5 (run-single) x
Please enter your Password : 123456

----- Welcome to TRAVGUIER System -----
-----
3. Add reservation info.
4. View reservation info.
5. Edit reservation info.
6. Delete reservation info.
9. Exit.
-----

Write The Number Of Process : 3
Please enter the destination: dubi
Please enter your budget: 30000
Please enter the desired airline: Nas
Approximately how many times can you make a reservations per month?: 3

Your reservation is complete!

----- Welcome to TRAVGUIER System -----
-----
3. Add reservation info.
4. View reservation info.
5. Edit reservation info.
6. Delete reservation info.
9. Exit.
-----

Write The Number Of Process : |
```

Implementation Run(View)

```
Output - Project_251_group5 (run-single) x
Airlane :Airlines

*****

*****

Username :Ahmed
Password :123456
First name :Ahmed
Last name :ALAmodi
National number :1103452567
Gender :M
Phone number :966523485248
Customer ID :105438
Plan ID :3452
Destination :jeddah
Budget :40000
Airlane :Nesma Airlines

*****

----- Welcome to TRAVGUIER System -----
-----
3. Add reservation info.
4. View reservation info.
5. Edit reservation info.
6. Delete reservation info.
9. Exit.
-----
Write The Number Of Process :
<
```

Implementation Code(Edit)

```
case 5: {
    while (true) {
        //Edit reservations
        Scanner Read_Log_cus7 = new Scanner(Log_cus);
        Scanner Read_Log_cus8 = new Scanner(Log_cus);
        custm.viewCusReserv(Read_Log_cus7, Read_Log_cus8);
        Read_Log_cus7.close();
        Read_Log_cus8.close();

        System.out.println("Please choose Plan-ID that do you want to edit it : ");
        String chol = sc.next();

        System.out.println("_____");
        System.out.println("1. Edit Destination.");
        System.out.println("2. Edit Budget.");
        System.out.println("3. Edit Airline.");
        System.out.println("4. Exit.");
        System.out.println("_____");

        System.out.println("Please choose from 1 to 3: ");
        int cho = sc.nextInt();

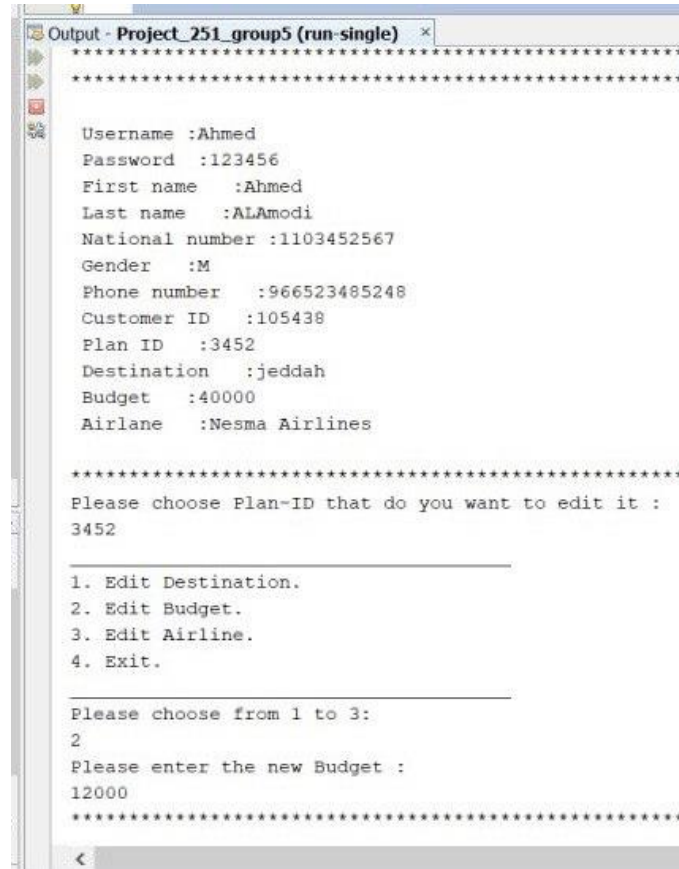
        switch (cho) {
            case 1: //Edit Destination
                System.out.println("Please enter the new Destination : ");
                String newDest = sc.next();
                String newInfoCus = "";

                Scanner Read_Log_cus5 = new Scanner(Log_cus);
                int lineNum = numOfLines(Read_Log_cus5);
                Read_Log_cus5.close();
```


Implementation Code(Edit)

```
switch (cho) {  
    case 1: { //Edit Destination  
        System.out.println("Please enter the new Destination : ");  
        String newDest = sc.next();  
        String newInfoCus = "";  
  
        Scanner Read_Log_cus5 = new Scanner(Log_cus);  
        int lineNum = numOfLines(Read_Log_cus5);  
        Read_Log_cus5.close();  
  
        Scanner UsersInfo = new Scanner(Log_cus);  
        String[][] info = new String[lineNum][];  
        for (int i = 0; i < info.length; i++) {  
            String read = UsersInfo.nextLine();  
  
            info[i] = read.split(",");  
  
            if (chol.equals(info[i][8]) && info[i][0].equals(custm.getUsername())  
                && info[i][1].equals(custm.getPassword())) {  
  
                newInfoCus = info[i][0] + "," + info[i][1]  
                    + "," + info[i][2] + "," + info[i][3]  
                    + "," + info[i][4] + "," + info[i][5]  
                    + "," + info[i][6] + "," + info[i][7]  
                    + "," + info[i][8] + "," + newDest  
                    + "," + info[i][10] + "," + info[i][11];  
  
            }  
        }  
        UsersInfo.close();  
  
        UsersInfol.close();  
  
        Log_cus.delete();  
        File trem = new File("Cus_info.txt");  
        output.renameTo(trem);  
  
        break;
```

Implementation Run(Edit)



```
Output - Project_251_group5 (run-single)
*****
Username :Ahmed
Password :123456
First name :Ahmed
Last name :ALAmodi
National number :1103452567
Gender :M
Phone number :966523485248
Customer ID :105438
Plan ID :3452
Destination :jeddah
Budget :40000
Airplane :Nesma Airlines

*****
Please choose Plan-ID that do you want to edit it :
3452

1. Edit Destination.
2. Edit Budget.
3. Edit Airline.
4. Exit.

Please choose from 1 to 3:
2
Please enter the new Budget :
12000
*****
```

Implementation Code(Delete)

```
case 6: {  
  
    //Delete reservations  
    Scanner ss = new Scanner(Log_cus2);  
    Scanner ss2 = new Scanner(Log_cus2);  
    int res_num = custm.viewCusReserv(ss, ss2);  
    ss.close();  
    ss2.close();  
  
    while (true) {  
  
        if (res_num > 0) {  
            System.out.print("Enter the reservation number that do you want to delete it: ");  
            String planID = sc.next();  
            Delete("Cus_info2.txt", planID, 9, ",");  
            res_num--;  
            System.out.println("                The deletion is done !");  
  
            System.out.println("*****");  
            System.out.println("\n\n1. Redisplay.");  
            System.out.println("2. Delete reservation.");  
            System.out.println("3. Exit.\n\n");  
            System.out.println("*****");  
            System.out.println("Please choose from 1 to 3: ");  
            int choose = sc.nextInt();  
        }  
    }  
}
```

Implementation Code(Delete)

```
public static void Delete(String filePath, String planID, int positionOfTerm, String delimiter) {
    int position = positionOfTerm - 1;
    File inputFile = new File(filePath);
    File NewFile = new File("newFile.txt");

    String currentline;
    String Data[];

    try {

        BufferedWriter writer = new BufferedWriter(new FileWriter(NewFile, true));
        PrintWriter pWriter = new PrintWriter(writer);
        BufferedReader reader = new BufferedReader(new FileReader(filePath));

        while ((currentline = reader.readLine()) != null) {
            Data = currentline.split(",");
            if (!(Data[position].equalsIgnoreCase(planID))) {
                pWriter.println(currentline);
            }
        }

        pWriter.close();
        pWriter.flush();
        writer.close();
        reader.close();

        inputFile.delete();
        File trem = new File(filePath);
        NewFile.renameTo(trem);

    } catch (IOException e) {
    }
}
```

Implementation Run(Delete)

```
Output - Project_251_group5 (run) x
----- Welcome to TRAVGUIER System -----
-----
3. Add reservation info.
4. View reservation info.
5. Edit reservation info.
6. Delete reservation info.
9. Exit.
-----
Write The Number Of Process : 6
*****

Username :Ahmed
Password :123456
First name :Ahmed
Last name :ALAmodi
National number :1103452567
Gender :M
Phone number :966523485248
Customer ID :105438
Plan ID :1245
Destination :Makkah
Budget :10000
Airplane :Flynas

*****
*****

Username :Ahmed
Password :123456
First name :Ahmed
```

Implementation Run(Delete)

```
Output - Project_251_group5 (run) x
*****
*****

Username :Ahmed
Password :123456
First name :Ahmed
Last name :ALAmodi
National number :1103452567
Gender :M
Phone number :966523485248
Customer ID :105438
Plan ID :3452
Destination :jeddah
Budget :40000
Airplane :Nesma Airlines

*****
Enter the reservation number that do you want to delete it: 3452
    The deletion is done !
*****

1. Redisplay.
2. Delete reservation.
3. Exit.

*****
Please choose from 1 to 3:
```

Refactoring

- Edit some variables, methods, and classes names.
- Simplify the code in the main class(Travguier) into a separated method and invoke them in the main (so the code becomes easy to maintainable).
- Remove all unused variables.

Test

- An automated test framework (e.g., Junit) is a system that makes it easy to write executable tests and submit a set of tests for execution.

Test Code

```
@Test
public void testAdd_plan() throws Exception {
    System.out.println("Add_plan");
    Scanner sc = null;
    BufferedWriter wr = null;
    Customer cu2 = null;
    boolean h = Travguier.Add_plan(sc, wr, cu2);
    // TODO review the generated test code and remove the default call to fail.
    assertTrue("True",h);
}
```

```
@Test
public void testDelete() {
    System.out.println("Delete");
    String filePath = "Cus_info2";

    String planID = "3452";
    int positionOfTerm = 9;
    String delimiter = ",";
    Travguier.Delete(filePath, planID, positionOfTerm, delimiter);
    assertEquals(filePath,planID,9,"");

    // TODO review the generated test code and remove the default call to fail.
}
```

Test Code

```
@Test
public void testNumOfLines() {
    System.out.println("numOfLines");
    Scanner Sc_read_lines = null;
    int expResult = 0;
    int result = Travguier.numOfLines(Sc_read_lines);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
}
```

Test Result

The screenshot displays the NetBeans IDE 8.2 interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The toolbar shows various icons for file operations and running tests. The left sidebar contains the 'Projects' and 'Services' tabs, with 'Projects' selected, showing a tree view of the project structure. The main editor area shows the 'TravguierTest.java' file with the following code:

```
115 Customer cu2 = null;
116 boolean h = Travguier.Add_plan(sc, wr, cu2);
117 // TODO review the generated test code and remove the default call to fail.
118 assertTrue("True",h);
119
120
121
122 /**
123  * Test of numOfLines method, of class Travguier.
124  */
125 @Test
126 public void testNumOfLines() {
127     System.out.println("numOfLines");
128 }
```

The bottom pane shows the 'Test Results' window for 'TravguierTest'. It displays the following information:

- Tests passed: 28/57/40
- 2 tests passed, 5 tests caused an error. (0.264 s)
- Test results list:
 - testGeneratRandomNum caused an ERROR: Not supported yet.
 - testMain caused an ERROR: java.util.NoSuchElementException
 - testNumOfLines caused an ERROR: java.lang.NullPointerException
 - testAdd_plan caused an ERROR: java.lang.NullPointerException
 - testProcessMenu passed (0.0 s)
 - testInitialMenu passed (0.001 s)
 - testDelete caused an ERROR: Not supported yet.

The right side of the test results window shows the output of the tests, including a list of menu items and a welcome message:

```
1. SignUp Create A New Account.
2. Login To Your Account.
9. Exit.

Write The Number Of Process : numOfLines
Add_plan
Please enter the destination: processMenu

----- Welcome to TRAVGUIER System -----

3. Add reservation info.
4. View reservation info.
5. Edit reservation info.
6. Delete reservation info.
9. Exit.
```

Lesson learned

- HOW to make our code in a simple way.
- How to use the extreme Programming approach.
- Work with a team.
- Improve our programming.

Challenges

- Time challenges.
- It was a little difficult to work with a new approach.
- Work with team.
- Improve our programming.

Tools

- NetBeans.
- Trello.
- GitHub.
- Junit.
- Google Drive.

Future Work

- Make the users able to rate and add reviews for visited places.
- Give the user a choice to apply filters (by placed by low rate, by high rate) when viewing reservations.
- User can make plan by himself.
- Take the budget from the user in range.



Conclusion

In conclusion, technology is found to facilitate our life's so we worked on this concept and work hardly to provide an application that will help users to enjoy and spend time with their families under their conditions and choices.

Organizational & flow

Somaya Nimatallah	Apply coding of one core function (with almas) Use case discretion (login, delete reservation) Sequence diagram (login, delete reservation)
Almas Nogali	Apply coding of one core function (with somaya) Use case discretion (add reservation, register) Sequence diagram (add reservation, register)
Jomana Almaghrabi	Apply coding of one core function (with dana) Use case discretion (delete reservation, edit reservation) Sequence diagram (delete reservation, edit reservation)
Mona Alshammari	Apply coding of one core function (with jomana) Use case discretion (choose budget, payment) Sequence diagram (choose budget, payment) Class diagram, Use case
Dana Alsaedi	Apply coding of one core function (with mona) Use case discretion (view reservation, generate report) Sequence diagram (view reservation, generate report) Use case



Thank You

Any questions?

