Final Report

PROJECT NAME: Accommodation Web Portal

GROUP NAME: superNB

GROUP MEMBERS: Wenfei Guo(z5135080)

Tao Bai (z5111218)

Shengchen Xue (z5111075)

Zhichao He (z5282955)

Outline

Abstract

With the rapid development of the Internet and Information Technology, there is a trend that more and more services are transplanted to the Internet. This kind of evolution has led to significant success to countless enterprises and industry.

Similar to many other countries and regions, Australia especially New South Wales is a famous tourism destination. Accommodation industry is raised by the development of tourism. It's fairly reasonable for developing accommodation web portal to extend the business.

Our COMP3900 group (superNB) aims to develop a most convenient and helpful website which can provide the renters with the latest accommodation information and most effective searching strategies and many other advantages among the current websites.

This report aims to introduce the background of the various accommodation websites as well as the current drawbacks and advantages. Also we will give an introduction of the system structure, functionality design and details, third party utilities, some highlights and challenges of the project and a detailed user documentation which give the user a guidance of our web portal.

Background

The fact is that the explosion of Internet leads to many significant changes to people's daily life. Many traditional industries such as foodservice and cinemas have already developed relatively complete system of online service, and more and more customers are attracted by this modern way.

Online services provide the customers with better experience than the traditional business pattern. And the advantages also fit in the accommodation renting field which our group aims to implement. To be specific, the process of searching, viewing, comparison and paying for the accommodation will all be more convenient and fast. Customers never have to come to the estate agencies and some other places to find their dream house. They don't need to go to the real place to check the house and can just view the pictures on the website instead. Everything can be done via the website, it saves a lot of time and attracts more and more customers.

Feedback is also more convenient to be done through the website, customers can just leave their rating and suggestions by simply typing. These feedback is not only a kind of evaluation to the accommodations but also useful guidance to customers who are interested in the house or hotel. However, the traditional accommodation service cannot provide such comprehensive feedback. Although some big brand hotel may be able to provide such feedback through their corporate image, it cannot be more accurate and objective than the feedback from real customers.

The current accommodation websites like Airbnb are one of the most famous booking platform over the world. However there are still some drawbacks due to the various requirements from the customers. For example, Airbnb mainly provide accommodation host by individual providers. There should be more types of properties provided for those who want to live in a branded hotel or even Motel.

Another disadvantage among these websites is that the searching part provides too few filters or searching attributes. This may cause user's inconvenience as many

people would like to search for accommodations in more details such as whether there is a gym and whether pets are permitted to the property etc. Airbnb and other websites only provide the check-in and check-out date, guest number and locations but there are no such detailed characteristics. Although such characteristics can be viewed in the details of the properties but it's time-wasting for customers who has many requirements.

Considering extreme convenience of online service and the problems of the current websites, a better performed accommodation portal can be established by adding these features and will attract more customers and accommodation providers in New South Wales.

Overview

In this overview, the aim and overall architecture of this web application website will be demonstrated.

The main goal of this project is to build a convenient website to help customers to reserve or rent a suitable place and allow providers to publish their accommodations. Each account can not only publish accommodations but also make booking.

Backend overview:

1. 22 servlets in total. Each servlet has its own function to store data and help frontend reveal data.
2. Four tables and six corresponding objects created in total to store accommodation data, order data, basic user information and detailed user information.

This application should follow these design rules:

1. user-friendly, easy to understand and use. i.e. All the information on each website should be clear and accurate.
2. UI design style in each interface or website should be consistent.
3. Keeping user data securely. i.e. All the user key information should be encrypted in the database.
4. Input Information error reminder. i.e. All the information provided by user should be checked valid or not immediately.

The architecture of the system can be divided into five parts:

1. Backend Programming Languages Selection.
2. IDE Selection.
3. Web Framework Selection.
4. Front-end Framework Selection.
5. Database Selection.
6. Web Server Selection.

The overall architecture of this project:

1. Backend Programming Languages Selection: Java
   As explained in proposal, Python and Java are both popular languages in developing web projects. However, the two backend developers in this group only have little prior knowledge of Python and both of them have no experience of developing backend part in a web application, so they have to learn some knowledge about web development at first. After considering those aspects, Java is the first choice for backend development.
2. IDE Selection: IDEA
   In the beginning, our team selected Eclipse as IDE. However, after noticing that Eclipse does not have a good support for Java EE, we decided to use IDEA as our IDE. As it is the first time to use this IDE, building environment for each group member spent a lot of time.
3. Web Framework Selection: Java EE
   Java EE is the basic Framework for Java Web Development, it is relatively easy to develop and debug. The backend could process and deliver data independently. Also, using Java EE can keep this project MVC and ensure this project will finish ontime.
4. Front-end Framework Selection: HTML, JavaScript, CSS
   HTML, CSS, and JavaScript are three district coding languages that together are used to build Website and Web Applications.
5. Database Selection: MySQL and JDBC
   MySQL is a free and compact database software. More importantly, it is open-source, so using MySQL could let our team maintain data more efficiently. The two backend developer have learnt some MySQL tutorial before, so they do not need to spend extra time to learn that. Our team using Java as backend programming language so JDBC(Java Database Connectivity) could let backend have more efficient and effective control on database.
6. Web Server Selection: Tomcat
   Tomcat is a stable open-source web server software with long history. It is not hard to deploy so it can be deployed on all members' computers easily. Also, Java EE have a good support for Tomcat so it is friendly to backend developer to maintain data transmission.

The four main functionality design are:

1. User information management module.

   After setting Tomcat server, the localhost IP address will be open to the public. Then public user could access this website by typing localhost IP address with /Accommodation. There are only one account type available for users. Customers can be both tenant and accommodation provider by using this account.

   This module contains login system, detail information manage system and rating system. Users can make new order or upload new house only after login, so new user must register a new account first. This module also provide login and modify further user information function. After finishing order, user could also rate the owner of that accommodation.
2. Accommodation search module and visitor request module.

   (1) In the homepage, user can type in the locations, booking dates and guest numbers to search the accommodations that they want.
   (2) In the main search page, user can modify guest numbers and research again.
   (3) In the main search page, there are more filters for users to help them find accurate accommodations that they want. User can search personal accommodations and company accommodations separately. Details of this part will be shown in functionality design part.
3. Accommodation advertising module and accommodation review module.

   User can upload both company and personal accommodations. Uploading different types of accommodation needs to upload different information, like company accommodation needs to provide stars and hotel type. User can upload at most nine pictures for each accommodation. Also, in the post accommodation lists, user could delete the accommodation that post before.
4. Accommodation Booking module and Order management module.

   In the accommodation information page, user could make order if that accommodation available. After made the order, user could review the order and give rate to the house owner, each order could only be rated once. Also, in the order list, user could review the order at any time.

Functionality Design

Accommodation Advertising Module

We need to build a tidy and plentiful interface to let provider can show information that they wanted, so we will as totally as possible to cover 。We separate two parts:

personal and company, for users to fill in accommodation information more conveniently and correctly. Under these two choices, there are some different questions to aim at two different situations. It can avoid confusion usefully and help users understand what they need to do.

In each page on the top right corner exist a button called “personal central”. After click it users will see a button ““Become a host”. If users want to rent out their house, they can click this button and it will jump to a new website name “provider.html” to fill information of their places.

➢ **Step 1: Add Place Information:**

- In provider.html, the first line is “Title”, providers can write a synopsis about their place, it will show on search pages to help customer convenient know introduction of your place.
- Then it has a textbox: “Address”,providers need to fill in address of their place, it will store in database and show in accommodation information page to help customer know.
- “City” text box: providers need to fill in City, after it store in database, it will be a search criteria because when customer search suitable location, they may search by aim city.
- “Suburb” text box: providers need to fill in Suburb, after it store in database, it will be a search criteria because when customer search suitable location, they may search by aim suburb for more accuracy.
- “Zip code” text box: providers need to fill in Zip code, for more correctly to show address of their place, avoid misunderstand like duplication of suburb or city name.
- “Price per Day ($) text box: providers write price of this accommodation per day, after store in database, it can be a criteria for customer sort(Low price sort) and calculate total price.
- Available Date: provider need to select check in date and check out date of this accommodation place, for avoid duplication rent or be a search criteria when customer search, it also necessary when calculating total price.
- Selected boxes for choose type of provider:
    - personal
    - company

  After Choose type of provider, it will display some questions which aim to different situation. it will store in database and become search criterias to help customers find suitable place.
    - If provider click company before:
        - show select boxes for choose kind of hotel:
            A. Capsule hotel
            B. Boutique hotel
            C. Lovehotel

D. Ryokan

- show select boxes for choose level of hotel:
  A. no star
  B. 1 star
  C. 2 star
  D. 3 star
  E. 4 star
  F. 5 star
  G. 6 star
  H. 7 star
- offical web page text box:
  Providers fill in their hotel official page here. when mouse move on this text box, it will to display a text to destripe why providers need to
  provider official page here.

- If provider click personal before:
  - display “Choose type of rent” and 3 new choices under kind of place:
    A. Entire place
    B. Private room
    C. Shared room
  - display select boxes for choose kind of place which providers want to list in the first line:
    A. 3 Choice:
       a. Apartment,
       b. Unit
       c. House

    just can choose one type
  - Display “ Number of bedroom”, “Number of bathroom”, “Numbers of kitchen” for providers to provide these base information of their place.
  - Display “Amenities” check box: for provider convenience to provider amenities which they exist. it can also become search criteria when customer search.
    A. Free Park
    B. Gym
    C. Wifi
    D. Lift
    E. TV

- “Max Number of guest”: provide the max number guests. it can also become search criteria when customer search.
- “Able to fee pet?” check box: for provider convenience express.

- A big text box named “Description” for provider to add additional detail about this accommodation.

➢ **Step 2: Update some photos for place:**

Able to upload pictures about this accommodation.It can let provider show their house more intuitively and can also help consumer know this house more directly.

There are two buttons at the end of the webpage:

- Reset button: If providers upload too much wrong information in this page, they can click this button to clean all answers on this page.
- submit button: After providers upload their information, they can click this button and the system will transfer these information in backend and this place will enter into database.

Accommodation Booking Module

A list of accommodations that satisfied customer request will be displayed after user press the ‘search’ button. Customers can click any result accommodation for further detail. Accommodation detailed information pages will contain different content different by provider type.

If the accommodation is provided by company, the features would be:

- Related accommodation information: Accommodation Advertisement information will be shown.
    - Accommodation related pictures (0~9)
    - Accommodation Title
    - Accommodation star level
    - Pet allowance
    - Hotel Type
    - Accommodation City
    - Accommodation Suburb
    - Accommodation PostCode
    - Accommodation Address
    - Accommodation Description
    - Price Per Day
    - **URL to offocial hotel website**: users can make orders in offical hotel page.
- Google Map: we apply Google Map as Third Party Functionality in this page, users can find the location in map by input detailed location information.

If the accommodation is provided by individual, the features would be:

- Related accommodation information: Accommodation Advertisement information will be shown.
    - Accommodation title
    - Accommodation related pictures (0~9)
    - Accommodation Structure Type
    - Accommodation Rent Type
    - Accommodation Type
    - Accommodation City
    - Accommodation Suburb
    - Accommodation PostCode
    - Accommodation Address
    - Accommodation Description
    - Price Per Day
    - Bedroom Number
    - Bathroom Number
    - Kitchen Number
    - Pet Allowance
    - Park
    - Gym
    - Wifi
    - Lift
    - Television
- Provider's personal detail
- Google Map: we apply Google Map as Third Party Functionality in this page, users can find the location in map by input detailed location information.
- **Order Section**(users have to login before applying this function, they will receive alert if not):
    - check-in and check-out date: Users need to confirm valid the check-in and check-out date before they book the accommodation.
    - Guest Number: user need to provide guest number for booking, if it is over the maximum affordable number of the accommodation, request will be alert and reject.
    - booking button: After users make sure they want to book this accommodation, they can press this button to start booking, it will jump to Customer Detail Confirm page.
- Customer Detail Confirm: Customers need to fill in and confirm their personal detail before they finish booking. Once they finish confirmation, booking is made and will jump to complete order page.

- Complete Order Page: This page will include related order information and accommodation. Customers can also rating in this page once since they have done the payment.

Accommodation Search Module

This part contains one of the most key functions of the website. This module implements the searching system which helps the user to find target accommodation information according to the request. As there are thousands of different information in the database, using a search module is much better than looking for them aimlessly. Users' experience will be highly improved. The implementation of this part is mostly backend's work, so there is no much user interface of search module. This module is applied in homepage.We will apply javascript technology to achieve request proposal of Date-Related Fields that can show calendar and the user-defined search bar. After customers uploaded their acquirement, they can simply click 'search' button which will deliver those data to backend for result. HTML and CSS technique can be helpful in User Interface design.

Features:

- Search required information: Users have to input listed accommodation information before searching, this module can be achieved before login. Alerts will be shown if users press search button before fill in the blank:
    - Date-Related Fields: Users can upload their preferred check-in and check-out date here. After they click the blank field, a calendar will pop up for them to specify check-in and check-out date.
    - Accommodation Location: Users can apply their location requirement here. Format of postcode, city, suburb, specific street can all be accept no matter in individual or combination.
- Search button: this is the only visual part of the search module. After the button is clicked, the webpage will then process the input request and send a request to the backend to do the searching.
- MySQL database: all the operations such as adding, updating and searching for information are based on the database. The database must be designed according to the requirement of the website. We should create essential tables to store the information and control the amount of the tables. Otherwise, the relations between each table will be difficult to establish.
- Search request processing: the input request should be processed and analyzed for further use. First, we must check whether the submit button was clicked and whether there is any request is input or selected. Only after we make sure that the requests are valid can we continue to search for the information.
- Information searching: the backend does accordingly query operations in the database to find whether there is any appropriate item which matches the user's request.
- Result display: After searching is done, the webpage should show the user the searching result. If there are no matching results, the webpage should tell the user

about the situation. Otherwise, the result of searching should be displayed in a table by doing some query in the database. And also, the results should be sorted by released date or the rating. The result sorting is also implemented by MySQL query in the backend.

Visitor Request Module

In this project, we focus on providing users the most accurate and specific information based on their requests. This module can be helpful in finding the acquired accommodation properties and filtering out inappropriate results. The “Request Screen” will contain a combination of button filters and checkbox filters.

Features:

Button Filters:

- Show all Button: click to see all satisified result.
- Low Price Button: sort result list in price order
- Guest Button: will provide a dropdown to allow user change guest number
- Provider Type button: have two options (personal and company), new detailed button will pop up by choosing any one of the selection.
- Personal Accommodation detail Button: will be shown by choosing personal in provider type dropdown. It include different checkboxes that will be shown below.
- Company Accommodation detail Button: will be shown by choosing company in provider type dropdown. It include different checkboxes that will be shown below.

Checkout Filters:

- In Personal Accommodation detail Button Dropdown
    - Amenities
        - park
        - gym
        - wifi
        - lift
        - television
    - Bedroom Number
    - Bathroom Number
    - Kitchen Number
    - Place Type:
        - House
        - Unit
        - Apartment
    - Structure Type:
        - EntirePlace
        - PrivateRoom,
        - ShareRoom
- In Company Accommodation detail Button Dropdown
    - Hotel Star Level (1-7 star)
    - Hotel Type

- CapsuleHotel
- BoutiqueHotel
- LoveHotel
- Ryokan

Accomodation Review Module

User can review post accommodation list, the bought order list and sold order list.
In the post accommodation list, user can browse all the accommodations that he posted. He can also access the information page of each accommodation by clicking “View Details” button. Furthermore, user can delete the accommodations that posted before by clicking “Delete Accommodation”.
In the bought order list, user can see all the orders which he is the renter. After clicking each order, he can access order information page and give rate to the owner of that accommodation, each order could be rated only once.
In the sold order list, user can see all the orders which he is the seller.

User Information Management Module

This part contains all user informations like sign up, log in, personal information, item list, and order list.
Sign up: There are four steps in sign up:

1. Account Information:
   Enter first time used username and password, verify password need same as password.
   If user used a used username, it need to show:This username has been used.
   If user didn’t enter a username, it will show: username cannot be empty.
   If user didn’t write a password, it need to show: Password cannot be empty.
   If user verify Password is not match with password, it will show: Password not match.
   When all finish, click next step, it will show a information: Congratulation, you have registered successfully!
   Then jump to next step: personal infromation.
2. Personal Information:
   Enter personal information. User can fill in Salutation, Gender, First Name, Last Name, Email, Phone, Zip Code, Suburb and Home Address. It is not coerciveness, if user don’t want to enter, they can click “Pre Step|Back To HomePage” to go back homepage, they can also click home button  on the top left corner to go back homepage.
3. Payment Information:
   Enter Payment information for fasr payment. If user fill in their payment information(card Type, Card Number, CVC, Card Holder Name, Expiry Month and Expiry Year) here, it will store in their personal information, when user pay in our system, they can use this card directly(InOrder/ Payment Part)

This step is optional, if user don't want to fill in, it just need to click home button on the top left corner to fo back homepage or click "Next Step|Confirm Your Details" to check information they fill in before.

4. Confirm Your details:
   Check all information correctly. If something wrong, just click "Prev Button" to go back last page to check and change.Information filled in before page will show in each text boxes, to help user check whether wrong and find which information need to be modified.

Log in: When customer login, they need to enter username and password. There are two button in this page: LOGIN and RESET button.
After enter username and password, if user click reset, it will empty these two text boxes. Click LOGIN: if username or password is wrong, it will display information: "Username or password is wrong"
Log out:
Change Password:
Personal information: After login, user will see a "Personal Central" button on top right corner in each page. Click "Personal Central", user will see "Personal Information" button, in this page, user can able to see all information which they edited. In here all information cannot be modified for avoid careless touching. If some information is wrong or empty, user can click button at the end of page: Edit information, then they can modify their information. "Edit information" Button changes to submit for user click after modify.
Item List: After login, user will see a "Personal Central" button on top right corner in each page. Click "Personal Central", user will see "Item List" button. In this page user can see All item which they posted. Click button "Rented Item List" can view all accommodations which be rented.Click button "All item List" will display all posted item.
Order List: After login, user will see a "Personal Central" button on top right corner in each page. Click "Personal Central", user will see "Order List" button. In this page user can see All item which they ordered. It will be displayed by Order created Date

Google Map Module

Use Google Map to give user a convenience window, click "+" to enlarge or click "-" to shrunk map to find address. User can also click it and jump to google map page to search address accurately.

Functionality Details

Accommodation Advertising Module

In provider page, the biggest problem is: user not only need to upload string or integer details, but also need to upload images.
Firstly we need to catch username, because it is necessary to find corresponding user in database, for store these information into right place. Username can be catched at window entry function:

```
$(function(){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
})
```

Because in different type of provider need to display display questions, firstly we build some fieldsets(corresponding each question), these fieldsets initiate “display none” :

1. id = “hotel”, for question : Type of your hotel
2. id = “hotel1”, for question: Level of your hotel
3. id = “personal”, for question: Choose type of rent
4. id = “personal1”, for question: Type of your place
5. id = “personal2”, for question: Numbers of bedroom, Numebr of bathroom
6. id = “personal3”, for quesiorn: Numbers of kitchen
7. id = “personal4”, for question: Amenitites
8. id = “urlset”, for: Official WebPage

Use javascript to build “tpe” function and used by personal radio and company radio to control which fieldset need to be displayed:

```
function tpe(obj){
    if(obj =="personal"){
        document.getElementById("personal2").style.display="";
        document.getElementById("personal").style.display="";
        document.getElementById("personal1").style.display="";
        document.getElementById("personal3").style.display="";
        document.getElementById("personal4").style.display="";
        document.getElementById("hotel").style.display="none";
        document.getElementById("hotel1").style.display="none";
        document.getElementById("urlset").style.display="none";
    } else if(obj =="company") {
        document.getElementById("hotel").style.display = "";
        document.getElementById("hotel1").style.display="";
        document.getElementById("urlset").style.display="";
        document.getElementById("personal").style.display = "none";
        document.getElementById("personal1").style.display="none";
        document.getElementById("personal2").style.display="none";
        document.getElementById("personal3").style.display="none";
        document.getElementById("personal4").style.display="none";
    }else {
        document.getElementById("hotel").style.display = "none";
        document.getElementById("hotel1").style.display="none";
        document.getElementById("personal").style.display = "none";
        document.getElementById("personal1").style.display="none";
        document.getElementById("personal1").style.display="none";
        document.getElementById("personal3").style.display="none";
        document.getElementById("personal4").style.display="none";
        document.getElementById("urlset").style.display="none";
    }
}
```

At the step of catch images, because there are 3 elements inside div “imgs”, so the initial value of i is 3 and every time i + 3. Because user may need to add multiple images, so we use function “addimg()” to add div:

```
function addimg() {
    i++;
    if(i <= 9){
        var index = $("#imgs").children().length;
        $("#imgs").append("<input type=\"file\" id="+index.toString()+" multiple=\"multiple\" name=\"image0\"  accept=\"image/*\" />\n" +
            "                    <input onclick=\"addimg()\" type=\"button\" value=\"Add New Photo\"><br>")

    }else{
        alert("At most 9 photos can be uploaded!")
    }

}
```

We set a limit number: user at most can add 9 images. If user wanted add more than 9 pictures, it will alert information: At most 9 photos can be upload”
we used javascript to deal with all values which user inputed. Every input have their own id, for collect data and send them into database.
Because images need to use “file”, for collect all information together, we need to use formData, and use append to add all informations which servlet needed into formData.

```
var formData = new FormData();
formData.append('file', $('#file')[0].files[0]);
if(imgs != 3){
    for(var i=3;i<imgs;i=i+3){

        formData.append('file', $("#"+i.toString())[0].files[0]);
    }
}

formData.append("userName", uname);
formData.append("guestNum",guest);
formData.append("datefilter",date);
formData.append("description",decription);
formData.append("accommodationType",typePlace);
formData.append("hotalType",typehotel);
formData.append("star",levelhotel);
formData.append("structureType",typePrivate);
formData.append("bedroom",bedroom);
formData.append("bathroom",barhroom);
formData.append("kitchen",kitchen);
formData.append("park",park);
formData.append("gym",gym);
formData.append("wifi",wifi);
formData.append("lift",lift);
formData.append("television",tv);
formData.append("address",address);
formData.append("postCode",code);
formData.append("suburb",suburb);
formData.append("city",city);
formData.append("pricePerDay",price);
formData.append("url",url);
formData.append("pet",pet);
formData.append("title",titles);
formData.append("rentType",typeRent);
```

After use formData stored all information, use “ajax” to upload information.Use “ POST” type and make sure we use the same route with corresponding servlet setted:

```
$.ajax({
    type: "POST",
    url: "/Accommodation/UploadBasicServlet",
    dataType: "json",
    data: formData,
    processData: false,
    contentType: false,

    success: function (msg) {
        console.log(msg)
        window.location.href="itemList.html?variable_name="+uname
    },
    error: function(error){
        console.log(error)
    }
})
```

For realized Notices on “Official Webpage” textbox, we use css to build a tooltip, use class “tooltip in input which id = “url” and build a span used class “tooltip_content”

```
<fieldset id="urlset" style="...">
    <label for="url" class="left_element">Official WebPage </label>
    <div class="tooltip">
        <input type="text" name="text_field" value="" size="30" id="url" placeholder="Please fill your offical webpage" class="tooltip">
        <span class="tooltip_content">
        <strong>Why we need this?</strong><br />It can easy for customers to find your hotel's offical page and help them order!
    </span>
    </div>

</fieldset>
```

Backend:

In the “UploadBasicServlet”, backend will check whether the data from frontend is formfield. If it is formfield, then store it as text data. Otherwise, check whether it is image files, if this input data is image file, then store it in the localhost. All other text data will store in the database. Also, the photopath and random image filnames will be stored in the databse.

```
FileItemFactory factory = new DiskFileItemFactory();
ServletFileUpload upload = new ServletFileUpload(factory);

List items = null;
try {
    items = upload.parseRequest(request);
} catch (FileUploadException e) {
    e.printStackTrace();
}
```

```
Iterator iter = items.iterator();
while(iter.hasNext()){
    FileItem item = (FileItem) iter.next();

    if(item.isFormField()){
```

Accommodation Booking Module

Frontend:

1. In both personal and company accommodation information pages, related accommodation information will be displayed. In JQUERY window.onload function(i.e. it is onload function since we want the data to be displayed as soon as user went into the page), we will apply AJAX method and use “GET” method to get required accommodation information after sending back the accommodation ID.

```
$(function (){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var accid = pair[1];
    var uuu = pair[2];
    accn = accid;
    uu = uuu;
    $.ajax({
        type: "GET",
        url: "/Accommodation/ConcreteAccommodationServlet",
        dataType: "json",
        data: {accommodationID:accid},
        success: function (data) {
            <!-- ammommodation key -->
            var accoinfo = data.accommodation;
            providern = accoinfo.userName;
            var accommodationID = accoinfo.accommodationID;
            var prov = accoinfo.userName;
```

To display the data from servlet, we apply following method in AJAX function in javascript.

```
var blocka = "<span>Accommodation: "+title+"</span>";
var blocka0 = "<span>Accommodation Provider: "+prov+"</span>";
var blocka1 = "<span>Structure Type: "+structureType+"</span>";
var blocka2 =  "<span>Rent Type: "+rentType+"</span>";
var blocka3 = "<span>Accommodation Type: "+accommodationType+"</span>";

$("#testblocka").html(blocka);
$("#testblocka0").html(blocka0);
$("#testblocka1").html(blocka1);
$("#testblocka2").html(blocka2);
$("#testblocka3").html(blocka3);
```

By applying this method, we can simply call it in body part to display information

```
<div id="testblocka"></div>
```

2. If the accommodation is provided by company, Users can make order in this make if accommodations are provided by individual. The order operation will be called if the order button is clicked.

```
<button class="orderbtn" style="color: #4f4f4f;font-family: 'Gill Sans';margin-left: 20%" onclick="return fororder()"><span>Order</span></butto
```

The fororder function will connect related servlet and provide required information to make a new order.

```
$.ajax({
    type: "POST",
    url: "/Accommodation/MakeOrderServlet",
    data: {accommodationID: akkk, inputDate:datefilter,guestNumber:guest,amount:amount, buyerName:lala, sellerName:lolo},
    dataType: "json",
    success: function(data){
        console.log(data);
        window.location.href="checkout.html?variable_name="+lala;

    },
    error: function(error){
        alert(error);
    }
});
return false;
```

3. After user make order, it will jump into payment page. Use window entry function to get information of users

```
$(function(){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    document.getElementById("person").value = "Hi, " + uname;
    $.ajax({
        type: "POST",
        dataType: "json",
        url: "/Accommodation/DisplayUserServlet",
        data: {
            userName: uname,
        },
        success: function (msg) {
            console.log(msg);
            var salutation = msg.user.salutation
            document.getElementById("salut").value=salutation;
            var email = msg.user.email
            document.getElementById("email").value=email;
            var firstname = msg.user.firstName
            document.getElementById("firstname").value=firstname;
            var lastname = msg.user.lastName
            document.getElementById("lastname").value=lastname;
            var gender = msg.user.gender
            document.getElementById("gender").value=gender;
            var phone = msg.user.phoneContact
            document.getElementById("phone").value=phone;
            var cardnamee = msg.user.cardName
            document.getElementById("nameoncard").value = cardnamee;
            var cardtype = msg.user.cardType
            if(cardtype == "Visa"){
                document.getElementById("visa").checked=true;
                document.getElementById("master").checked=false;
                document.getElementById("master").readonly=true;
                document.getElementById("visa").readonly=true;
            }else{
```

```
                document.getElementById("visa").checked=false;
                document.getElementById("master").checked=true;
                document.getElementById("master").readonly=true;
                document.getElementById("visa").readonly=true;
            }
            var cardno = msg.user.cardNum
            document.getElementById("cardno").value=cardno;

            var cvc = msg.user.cvc
            document.getElementById("cvc").value=cvc;

            var month = msg.user.expireMonth
            document.getElementById("month").value=month;

            var year = msg.user.expireYear
            document.getElementById("year").value=year;

        },
        error: function (error) {
            console.log(error)
        }

    })

})
```

Certainly user still can change informations.

There are 3 ways to pay for order: default card, new card or paypal.

Click default card will show the information which user inputed before.

User can also change informaitons if there exists some mistakes.

Click new card will show empty textboxed for user to fill in card information

After choose way to pay, click “Pay Now” button, it will show a modal.

For this modal,we use javascript and css

```
<section>
    <button id="modalBtn" class="button" style="...">Pay Now</button>
    <div id="simpleModal" class="modal" style="...">
        <div class="modal-content">
            <div>
                <header style="...">
                <span class="closeBtn">×</span>
                    <h2 style="...">Done Your Payment?</h2></header>
                <div style="...">
                    <button onclick="successPay()" style="font-family:'Arial Black';font-size:20px;
                    cursor: pointer;width: 200px;height: 100px;background-color: #2da5da;font-size: medium">
                        Yes, Done it!</button>
                    <button onclick="window.location.reload()" style="font-family:'Arial Black';
                    font-size:20px;cursor: pointer;width: 200px;height: 100px;background-color: #fb6b5b;
                    font-size: medium">No, Still has problem!</button>
                </div>

            </div>

        </div>
    </div>
    <script src="js/modal.js"></script>
</section>
```

we build a button which id = “modalBtn” and a div which id = “simpleModal” and a div which id = “modal-content.

In modal.js, these id can be get by “document.getElementById”, javascript will give them different actions to do:

```
var modal = document.getElementById( elementId: 'simpleModal');
var modalBtn = document.getElementById( elementId: 'modalBtn');
var closeBtn = document.getElementsByClassName( classNames: 'closeBtn')[0];
modalBtn.addEventListener( type: 'click', openModal);
closeBtn.addEventListener( type: 'click', closeModal);
window.addEventListener( type: 'click', outsideClick);
function openModal() {
    modal.style.display = "block";
}
function closeModal() {
    modal.style.display = 'none';
}
function outsideClick(e) {
    if (e.target == modal) {
        modal.style.display = 'none';
    }
}
```

in modal content, it will show two button, one is “Yes Done it!” and “ No, Still has problem!”

Click “Yes done it” button will call function “successPay()”, it will jump to complete order and send username and orderID into complete order page together:

```
function successPay(){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    var orderid = pair[2];
    window.location.href="CompleteOrder.html?variable_name="+uname+"="+orderid;
}
```

Click “No, still has problem” Button will close this modal and stay at this page. We use reload function in this button, reload page to stay at present page:

```
<button onclick="window.location.reload()"
        style="font-family:'Arial Black';font-size:20px;cursor: pointer;width: 200px;
        height: 100px;background-color: #fb6b5b;font-size: medium">No, Still has problem!</button>
```

4. Complete order will be displayed in a new page. This page will show related accommodation and order information by applying the method we mentioned above. In addition, rating system is also included in this page. There are five stars shown which represent five levels.

   After selecting the star level based on users’ experience, customers can click submit button to make a review.

```
<button id = "kkk"onclick="return forrate()">Apply</button>
<div id ="kkka" style="..."><h4> you have marked this accommodation already</h4></div>
```

The called function will connect related servlet to make rating recorded.

```
$.ajax({
    type: "POST",
    dataType: "json",
    url: "/Accommodation/RateUserServlet",
    data: {orderID:iddd, rate:i},
    success: function (data) {
        alert("Thank you");
        window.location.reload();
    },
    error: function(error){
        alert(error);
    }
});
return false;
```

In addition, add if statement to make sure customer can rate once in a order.

```
var reviewstate = data.order.whetherRate;
if(reviewstate == 0){
    document.getElementById("kkk").style.display = "block";
    document.getElementById("kkka").style.display = "none";
}else{
    document.getElementById("kkk").style.display = "none";
    document.getElementById("kkka").style.display = "block";
}
```

5. HTML and CSS technique are also applied in User Interface design.

Backend:

1. When making order, “MakeOrderServlet” will check whether this order valid by checking date, guest number and valid user. This servlet will return orderID to the front end if succeeded.

```
if(!dao.checkValidGuest(newOrder)){
    checkGuest = 0;
}
else{
    checkGuest = 1;
}

if(!dao.checkValidDate(newOrder)){
    checkDate = 0;
}
else{
    checkDate = 1;
}

if(!dao.checkValidOrder(newOrder)){
    checkOrder = 0;
}
else{
    checkOrder = 1;
}
```

2. Front end gives backend an orderID, then in the “DisplayOrderServlet”, backend will return an order with all information by doing a query for this input orderID.

Accommodation Search Module

Frontend:

1. Input information includes check in and check out date. Users can upload their preferred check-in and check-out date here. After they click the blank field, a calendar will pop up for them to specify check-in and check-out date.

```
<input id="datefilter" type="text" name="datefilter" placeholder="dd/mm/yy - dd/mm/yy" style="font-size:15px;
<script type="text/javascript">
    $(function() {
        $('input[name="datefilter"]').daterangepicker({
            autoUpdateInput: false,
            locale: {
                cancelLabel: 'Clear'
            }
        });
        $('input[name="datefilter"]').on('apply.daterangepicker', function(ev, picker) {
            $(this).val(picker.startDate.format('DD/MM/YYYY') + ' - ' + picker.endDate.format('DD/MM/YYYY'));
        });
        $('input[name="datefilter"]').on('cancel.daterangepicker', function(ev, picker) {
            $(this).val('');
        });
    });
</script>
```

2. After users fill in all required information, clicking “search” button can help to connect related servlet for required result. AJAX will also need to grab the input data from users. The way of achieving it is shown below. It will check if the information satisfied requirement, if so, the page will jump to search page with input requirement.

```
var datefilter = document.getElementById("datefilter").value;
alert(datefilter);
var location = document.getElementById("location").value;
alert(location);
var guest = document.getElementById("guest").value;
alert(guest);
$.ajax({
    type: "POST",
    url: "/Accommodation/SearchFirstServlet",
    data: {location:location,datefilter:datefilter,guestNum:guest},
    dataType: "json",
    success: function(data){
        console.log(data);
        alert("in ajex");
        if(uname == null){
            window.location.href="searchpage.html?variable_name="+location +"="+datefilter +"="+guest +"="+"noname";
        } else {
            window.location.href="searchpage.html?variable_name="+location +"="+datefilter +"="+guest +"="+uname;
        }

    },
    error: function(error){
        alert(error);
    }
});
return false;
```

Backend:

1. There are three input data (Location, Datefilter, Guest) should recieved in “SearchFirstServlet”.
2. Firstly, after received location data, using split method to separate each keywords and stored into one arraylist.

```
String location = request.getParameter( s: "location");
ArrayList<String> allLocationString = new ArrayList<~>();
for (String str : location.split( regex: "%20")) {
    allLocationString.add(str);
}
```

3. Secondly, using split method again to separate the date filter then get startdate and enddate.

```
String datefilter = request.getParameter( s: "datefilter");
ArrayList<String> allDateString = new ArrayList<~>();
for (String str : datefilter.split( regex: " - ")) {
    allDateString.add(str);
}

String startDateString = allDateString.get(0);
String endDateString = allDateString.get(1);
```

4. For each location keywords, the backend should do once query, then combine the results together.
5. In the end, backend will delete duplicate results and give this accommodation list to the front end.

```
for(int i = 0; i<secondList.size() -1 ; i++){
    int flag = 0;
    for(int j=i+1; j<secondList.size(); j++){
        if(secondList.get(i).getAccommodationID().equals(secondList.get(j).getAccommodationID()) && i!=j ){
            flag = 1;
        }
    }
    if(flag == 0){
        finalList.add(secondList.get(i));
    }
}
```

Visitor Request Module

Frontend:

1. After users jump in this page from homepage, they will need to click “show all” to see the satisfied result. homepage will pass some required information to apply searchfirst servlet again for result.

2. After users selected required filters, they can click “apply filters” button to apply. This button will call related servlet, pass the filter information and get the satisfied result list. You need to click “show all” again for the filtered result.

```
$.ajax({
    type: "GET",
    url: "/Accommodation/SearchSecondServlet",
    dataType: "json",
    data: {sort:lowPrice, whetherPet:whetherPet, guestNum:guestNumber, accommodationType:accommodationType,
        starOne:starOne, starTwo:starTwo, starThree:starThree, starFour:starFour, starFive:starFive, starSix:starSix, starSeven:starSeven,
        CapsuleHotel:capsuleHotel, BoutiqueHotel:boutiqueHotel, LoveHotel:loveHotel, Ryokan:ryokan, bedroom:bedroom, bathroom:bathroom, kitchen:kitchen,
        park:park, gym:gym, wifi:wifi, lift:lift, television:television, House:House, Unit:Unit, Apartment:Apartment, EntirePlace:EntirePlace, PrivateRoom:PrivateRoom,
        ShareRoom:ShareRoom, location: aaa, datefilter:bbb},
```

3. The result list will be displayed in the following way.

```
$.each(res, function (i, obj) {
    console.log(i);
    console.log(obj);
    str = '';
    str += '<div class="column">';
    str += '<div class="content">';
    str += '<img src="'+obj.photoPath0+'" alt="acco5" style="width:100%" onclick="check('+obj.accommodationID+')">';
    str += '<h4>' + 'Accommodation Name: "'+obj.title+'" </h4>';
    str += '<h4>' + 'Accommodation Location: "'+obj.city+'"</h4>';
    str += '<h4>' + 'Price Per Day: "'+obj.pricePerDay+'"</h4>';
    str += '</div>';
    str += '</div>';
});
$("#test").append(str);
```

Backend:

1. Backend will use “SearchSecondServlet” to do this search. Firstly, backend will judge whether the search result should display in price ascending or not. Then if user wants to search personal accommodation, this servlet will invoke “searchPersonal” method, otherwise it will invoke “searchCompany” method.

```
public static List<CompanyAccommodation> searchCompany(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException{
```

```
public static List<PersonalAccommodation> searchPersonal(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException{
```

2. When searching company accommodation, backend will create two arraylists for each filter, the star filter and hotel type filter. When doing each filtering, these arraylists will store corresponding filtered accommodation. After passed all filters, the backend will return a list with all accommodations which passed all filters.
3. When searching personal accommodation, backend will create corresponding arraylists for each filter. When doing each filtering, these arraylists will store corresponding filtered accommodation. After passed all filters, the backend will return a list with all accommodations which passed all filters.

Accomodation Review Module

Review is inside Complete Order page. Because user can just review one time, after review, the apply button need to change to “you had already review”, so we use window entry function to check wetherRate. If no rate before, apply button display, else notice.

```
$(function(){

    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    var orderID = pair[2];
    oid = orderID;
    var uname = "katie";

    $.ajax({
        type: "POST",
        dataType: "json",
        url: "/Accommodation/DisplayOrderServlet",
        data: {orderID:orderID},
        success: function (data) {
            var reviewstate = data.order.whetherRate;
            if(reviewstate == 0){
                document.getElementById("kkk").style.display = "block";
                document.getElementById("kkka").style.display = "none";
            }else{
                document.getElementById("kkk").style.display = "none";
                document.getElementById("kkka").style.display = "block";
```

Use RateUserServlet:

```
function forrate(){
    var i = 0;
    if(document.getElementById("ovewall-1").checked){
        i = 1;
    }
    if(document.getElementById("ovewall-2").checked){
        i = 2;
    }
    if(document.getElementById("ovewall-3").checked){
        i = 3;
    }
    if(document.getElementById("ovewall-4").checked){
        i = 4;
    }
    if(document.getElementById("ovewall-5").checked){
        i = 5;
    }
    var iddd = oid;
    $.ajax({
        type: "POST",
        dataType: "json",
        url: "/Accommodation/RateUserServlet",
        data: {orderID:iddd, rate:i},
        success: function (data) {

            alert("Thank you");
            window.location.reload();
        },
        error: function(error){
            alert(error);
```

If rate successfully, alert information “thank you” and reload page, after reload, notice “You had already review before” should be displayed because of entry function and “apply” button should be hidden.

Frontend will give orderID, rate and userName to the backend. In the “/RateUserServlet”, backend will get the all information of this user by using given userName. Then get the commented times and current rate of this user. After that, backend will calculate the new rate and commented times then update this user in the database. Also, backend will check whether this order is valid, whether this order has been rated before and whether this rate uploaded successfully.

```
int validOrder = 1;
int notRateYet = 1;
int successfulRate = 1;

if(returnOrder == null){
    validOrder  = 0;
}
else if(returnOrder.getWhetherRate() == 1){
    notRateYet = 0;
}
else{
    IndividualUser seller = userdao.displayInformation(inputUser);
    int count = userdao.giveRate(rateNumber, seller);
    if(count != 1){
        successfulRate = 0;
    }
    else{
        int orderCount = dao.updateOrder(returnOrder);
    }
}
```

User Information Management Module

Sign up:

In Sign up, we totally have 4 pages and each pages for different information.

- The first page is update page: this page is for account information. And username text box we need to check username whether have been used. Therefore we use function “check” function, use onblur on the username textbox to invoke this function when mouse move away from this textbox:

```
<label for="userName">Usename</label>
    <input type="text" id="username" class="form-icon form-icon-user" onblur="check()" placeholder="Username" required />
    <span style="color: red" id="information"></span>
```

  preinstall a span which id = “information” to show notice information.
  in check function, invoke correnspondingservlet: CheckUserNameServlet to check inputed username with username in database

```
function check() {
    var uname = document.getElementById("username").value;
    var password1 = document.getElementById("passwd1").value;
    var password2 = document.getElementById("passwd2").value;
    if (uname == "") {
        document.getElementById("information").innerHTML="Username Cannot be Empty!";
        return false;
    }else{
        document.getElementById("information").innerHTML="";
    }
    $.ajax({
        type: "POST",
        url: "/Accommodation/CheckUserNameServlet",
        data: {userName:uname},
        dataType: "json",
        success: function(data) {
            console.log(data)
            if(data.state == 0) {
                document.getElementById("information").innerHTML = "This username has been used";
                return false;
            }else{
                document.getElementById("information").innerHTML = "";
            }

        },
        error: function(error){
            console.log(error)
        }
    });
```

if data.state = 0, it means this username has existed in database. The span which id = “information” preinstalled before will display a information “This username has been used”
user need to change username into a first time used name. it will convenient for user to realize usability of username and change it timely.
In check function, we also check password whether usability:

```
    if(password1 == "") {
        document.getElementById("information1").innerHTML="Password Cannot be Empty!";
        return false;
    }else{
        document.getElementById("information1").innerHTML="";
    }
    if(password2 == "") {
        document.getElementById("information2").innerHTML="Verify Password Cannot be Empty!";
        return false;
    }else{
        document.getElementById("information2").innerHTML="";
    }
    if(password1.length <8){
        document.getElementById("information1").innerHTML="PassWord Should longer than 8-digit!";
        return false;
    }else{
        document.getElementById("information1").innerHTML="";
    }
    if(password1 != password2){
        document.getElementById("information2").innerHTML="Password not Match!";
        return false;
    }else {
        document.getElementById("information2").innerHTML = "";
    }
```

we also use onblur on password textbox and verify password textbox for check at any time. After each password textbox we all preinstall a span (id = information1 and id = information2) for display notice information:

```
<p><label for="passwd1">Choose a password</label>
    <input type="password" id="passwd1" onblur="check()" placeholder="Password" required />
    <span style="..." id="information1"></span>
<p><label for="passwd2">Verify password</label>
    <input type="password" id="passwd2" onblur="check()" placeholder="Password" required />
<span style="..." id="information2"></span>
```

“Next Step|Personal Information” button will invoke funciton “signup” to send neccssary information into database by correnspongding servlet “SignUpFirstServlet”

```
$.ajax({
    type: "POST",
    url: "/Accommodation/SignUpFirstServlet",
    data: {userName:username,password:password1},
    dataType: "json",
    success: function(data){
        alert("Congratulation!You have registered successfully!")
        console.log(data)
        window.location.href="update1.html?variable_name="+username
    },
    error: function(error){
        console.log(error)
    }

});
```

If success, it will alert information “ Congratulation! You have registered successfully” to inform user. Send username into next page by window.location.href, for next page can catch username to use.
In this step, backend using “/CheckUserNameServlet” to check whether the userID that customer input is already exists, this servlet receive userID and look up “user” table to check whether there are any existing user. If this userID or userName is already used, the backend will return failed signal to the frontend.
Also, backend uses “SignUpFirstServlet” to create new user and individual user in the database. Firstly, backend will encrypt the input password string. Then insert new record into “user” table with userID, userName and encrypted password string. Also, backend will insert a new record into “individualuser” table by the given userName.

- The second page is update1.page. Firstly use windows entry function to get necessary information like username, if user is go back to check information, entry function will get the information he/she input before and display on each text box.

```
$(function(){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    $.ajax({
        type: "POST",
        dataType: "json",
        url: "/Accommodation/DisplayUserServlet",
        data: {
            userName: uname,
        },
        success: function (msg) {
            console.log(msg);

            var salutation = msg.user.salutation;
            document.getElementById("salut").value=salutation;

            var email = msg.user.email;
            document.getElementById("email").value=email;

            var firstname = msg.user.firstName
            document.getElementById("firstname").value=firstname;
```

Click “Prev Step| Back To Homepage” to go back to homepage by window.location.href = “homepage.html”. Also need to called logout servlet to make sure user logout after they move away from sign up page.

```
function logout() {
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    $.ajax({
        type: "POST",
        url: "/Accommodation/LogOutServlet",
        dataType: "json",
        data: {userName:uname},
        success: function (msg) {
            window.location.href="../homepage.html"

        },
        error: function(error){
            console.log(error)
        }
    })

}
```

“Next Step|Payment Information” button will invoke function “upload()”, in this function we use servlet “SignUpSecondServlet”, send necessary information into database by corresponding servlet. We also check the email format to make sure user input a email into email address textbox, not other information.

After receive userName from the front end, Backend using “SignUpSecondServlet” to update the database, if backend could not find this input user, it will return failed signal to the frontend.

```
function upload(){
    var salut = document.getElementById("salut").value;
    var gender = document.getElementById("gender").value;
    var firstname = document.getElementById("firstname").value;
    var lastname = document.getElementById("last-name").value;
    var email = document.getElementById("email").value;
    var phone = document.getElementById("phone").value;
    var zipCode = document.getElementById("zip-code").value;
    var suburb = document.getElementById("suburb").value;
    var address = document.getElementById("address").value;
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    if(email != "") {
        var reg = /^\w+((-\w+)|(\.\w+))*\@[A-Za-z0-9]+((\.|-)[A-Za-z0-9]+)*\.[A-Za-z0-9]+$/;
        isok= reg.test(email );
        if(!isok) {
            alert("Wrong Email format ! ");
            return false;
        }
    };
    $.ajax({
        type: "POST",
        url: "/Accommodation/SignUpSecondServlet",
        data: {userName:uname,salutation:salut,gender:gender,firstName:firstname,
            lastName:lastname,email:email,phoneContact:phone
        ,zip:zipCode,state:suburb,address:address},
        dataType: "json",
        success: function(data){
            console.log(data)
            window.location.href="update2.html?variable_name="+uname
        },
        error: function(error){
            console.log(error)
        }
    });
```

After stored information successfully, jump to update2.html with uname.

- The Third page is update2.html.Use same way in update1.html to get necessary information like username etc. Button “Next step|Confirm your details” use corresponding servlet to stored information in database.
  After receive userName from the front end, Backend using “SignUpThirdServlet” to update the database, if backend could not find this input user or any invalid data, it will return failed signal to the frontend.

```
function upload(){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    var cardType = document.getElementById("cardtype").value;
    var cardNum = document.getElementById("card-number").value;
    var cvc = document.getElementById("cvc").value;
    var cardName = document.getElementById("card-holder").value;
    var expireMonth = document.getElementById("month").value;
    var expireYear = document.getElementById("year").value;


    $.ajax({
        type: "POST",
        url: "/Accommodation/SignUpThirdServlet",
        data: {userName:uname,cardType:cardType,cardNum:cardNum,
            cvc:cvc,cardName:cardName,expireMonth:expireMonth,expireYear:expireYear},
        dataType: "json",
        success: function(data){
            console.log(data)
            window.location.href="update3.html?variable_name="+uname
        },
        error: function(error){
            console.log(error)
        }
    });
}
```

After success stored information into database, jump to next page with username.
“Pre Step|Personal Information” use function “back()” to jump back to update1.html with username:

```
function back() {
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    window.location.href="update1.html?variable_name="+uname
}
```

- The fourth page is update3.html, use window entry function to invoke DisplayUserServlet to get information from database and display on web for user to check.
  “DisplayUserServlet” will look up the input userName in the “individualuser” table, then return all the information of that user to the frontend. If backend could not find input user, it will reaturn failed signal to the frontend.

```
$(function(){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    $.ajax({
        type: "post",
        dataType: "json",
        url: "/Accommodation/DisplayUserServlet",
        data: {
            userName: uname,
        },
        success: function (msg) {
            console.log(msg);
            var name = msg.user.userName;
            document.getElementById("username").value=name;

            var email = msg.user.email;
            document.getElementById("email").value=email;
```

Log in:

In log page, after input username and password, click login button to invoke function “check()”, use LoginServlet to check username and password with database.

Firstly, backend will encrypted input password string because all the password in the database are encrypted, so if backend wants to do the matching for the input password string, must encrypted input string first. Then using “/LoginServlet” to check whether user input the correct password for the input userID. This servlet will search the “user” table to lookup the information. If userID and corresponding password matches, then return success signal to frontend, otherwise return failed signal. If the user logged in successfully, then backend will create a session about this login record, the backend will use this session in “IsLoginServlet” and “LogOutServlet”. In the “IsLoginServlet”, backend will check session which with login record directly to report to the frontend whether this user logged in.

```
function check() {
    var uname = document.getElementById("login").value;
    var passwd1 = document.getElementById("password").value;
    if (uname == "") {
        alert("UserName Cannot be Empty!");
        return false;
    }else if(passwd1 == "") {
        alert("PassWord Cannot be Empty!");
        return false;
    }
    $.ajax({
        type: "POST",
        url: "/Accommodation/LoginServlet",
        data: {userName:uname,password:passwd1},
        dataType: "json",
        success: function(data){
            console.log(data)
            if(data.state == 0){
                alert("Username or password is wrong");
                return false;
            }else{
                var content = data.content;
                //window.location.href="<%=basePath %>index.jsp?userName="+content.userName
                window.location.href="homepage.html?variable_name="+uname
            }
        },
        error: function(error){
            alert(error);
        }
    });
    return false;
```

Personal Information:

There are two pages used in here.In personal information page, it can display all information by entry function:

```
$(function(){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    document.getElementById("person").value = "Hi, " + uname;
    $.ajax({
        type: "POST",
        dataType: "json",
        url: "/Accommodation/DisplayUserServlet",
        data: {
            userName: uname,
        },
        success: function (msg) {
            console.log(msg);

            var salutation = msg.user.salutation;
            document.getElementById("salutation").value=salutation;

            var email = msg.user.email;
            document.getElementById("email").value=email;

            var firstname = msg.user.firstName
            document.getElementById("firstname").value=firstname;

            var lastname = msg.user.lastName
            document.getElementById("lastname").value=lastname;

            var gender = msg.user.gender
            document.getElementById("gender").value=gender;
```

Use “DisplayUserServlet” to get data in database, by give this servlet a username, it will search the user table where username is the input username and return all the information of that user. After received all the information of that user from the backend, those information will be display on the web page by document.getElementById.value. All textboxes are “read only” to avoid user careless touching.

If user want to change information, click “edit information”, it will jump to next page “edit information.html” by `window.location.href="editInformation.html?variable_name="+uname`.”Edit information” page is almost same with personal page however all text boxes can be changed.Click “Submit” button, invoke “upload()”function to upload all information:

```
function upload(){
    var salut = document.getElementById("salut").value;
    var gender = document.getElementById("gender").value;
    var firstname = document.getElementById("firstname").value;
    var lastname = document.getElementById("lastname").value;
    var email = document.getElementById("email").value;
    var phone = document.getElementById("phone").value;
    var zipCode = document.getElementById("zip-code").value;
    var suburb = document.getElementById("suburb").value;
    var address = document.getElementById("address").value;
    var cardType = document.getElementById("cardtype").value;
    var cardNum = document.getElementById("card-number").value;
    var cvc = document.getElementById("cvc").value;
    var cardName = document.getElementById("card-holder").value;
    var expireMonth = document.getElementById("month").value;
    var expireYear = document.getElementById("year").value;
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    if(email != "") {
        var reg = /^\w+((-\w+)|(\.\w+))*\@[A-Za-z0-9]+((\.|-)[A-Za-z0-9]+)*\.[A-Za-z0-9]+$/;
        isok= reg.test(email );
        if(!isok) {
            alert("Wrong Email format ! ");
            return false;
        }
    };
    $.ajax({
        type: "POST",
        url: "/Accommodation/UpdateUserServlet",
        data: {userName:uname,salutation:salut,gender:gender,firstName:firstname,
            lastName:lastname,email:email,phoneContact:phone
            ,zip:zipCode,state:suburb,address:address,cardType:cardType,cardNum:cardNum,
            cvc:cvc,cardName:cardName,expireMonth:expireMonth,expireYear:expireYear},
        dataType: "json",
        success: function(data){
```

Use “UpdateUserServlet” to to upload all needed data. This servlet will check whether this user exists, if this user exists then update “individualuser” table with all the input value, otherwise give failed signal to the front end. After frontend received success signal, it will jump back to personalPage.html with username by:`window.location.href="personalPage.html?variable_name="+uname`

Post Accommodation Information:

First use window entry function, catch all items which posted by this user. Use PostItemServlet, give it a username, then this servlet will return back all information we needed.

After backend received userName, it will search all data from accommodation table where owner name is input user. Then return a list of order to front end.

```
window.onload=function () {
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    $.ajax({
        type: "GET",
        url: "/Accommodation/PostItemServlet",
        dataType: "json",
        data: {userName:uname},
        success: function (data) {
            document.getElementById("rented").style.display = "none";
            var ls = data.returnList
            for(var i = 0; i< ls.length;i++){
                var item = ls[i];
                var name = item.title;
                var photo = item.photoPath0;
                var idd = item.accommodationID;
                if(photo == ""){
                    photo = "images/no_image.jpeg"
                }
                $("#all").append("<tr>\n" +
                    "                <td style=\"border-style: solid; border-width: 1px;border-color: b
                    "                    <img src="+photo+" style=\"width:200px\">\n" +
                    "                </td>\n" +
                    "                <td style=\"border-style: solid; border-width: 1px;border-color: b
                    "                    <p style='vert-align: middle'>"+name+"</p>\n" +
                    "                </td>\n" +
                    "                <td style=\"border-style: solid; border-width: 1px;border-color: b
                    "                     <input style=\"cursor:pointer;text-align:center;color:blue;bo
                    "                </td>\n" +
                    "                <td style=\"border-style: solid; border-width: 1px;border-color: b
                    "                     <input style=\"cursor:pointer;font-size: 14px;color:red;text-
                    "                </td>\n" +
                    "            </tr>")
            }
        }
```

“Click to View”invoke “check(obj)” function, “Delete”invoke “deleteitem(obj)” function

```
td style=\"border-style: solid; border-width: 1px;border-color: black;height:100px;width: 150px\">\n" +
   <img src="+photo+" style=\"width:200px\">\n" +
/td>\n" +
td style=\"border-style: solid; border-width: 1px;border-color: black;height:100px;width: 500px\">\n" +
   <p style='vert-align: middle'>"+name+"</p>\n" +
/td>\n" +
td style=\"border-style: solid; border-width: 1px;border-color: black;height:100px;width: 200px\">\n" +
    <input style=\"cursor:pointer;text-align:center;color:blue;border-color: transparent;font-size: 14px\" value='Click to View' readonly onclick='check("+idd+")' style='cursor:pointer;bo
/td>\n" +
td style=\"border-style: solid; border-width: 1px;border-color: black;height:100px;width: 200px\">\n" +
    <input style=\"cursor:pointer;font-size: 14px;color:red;text-align:center;border-color: transparent;\" value='Delete' readonly onclick='deleteitem("+idd+")' style='cursor:pointer;bord
/td>\n" +
```

In check(obj) function, because we need jump to corresponding page, so we need to send accommodationID and username(to keep login). argument : obj is accommodationID, for use in check function.

After user clicked “Click to View”, frontend will call “/ConcreteAccommodationAllServlet” and give backend accommodationID, the backend will search accommodation table and return all the information of that accommodation which matched the input accommodationID.

```
function check(s) {
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];
    var accommodationID = s;
    $.ajax({
        type: "GET",
        url: "/Accommodation/ConcreteAccommodationAllServlet",
        dataType: "json",
        data: {accommodationID: accommodationID},
        success: function (data) {
            console.log(data)
            if(data.state == 1){
                window.location.href="companyaccoinfo.html?variable_name="+accommodationID +"="+uname;
            } else if (data.state == 2){
                window.location.href="accoinformation.html?variable_name="+accommodationID +"="+uname;
            }
        },
        error: function(error){
            alert(error);
        }
    });
}
```

Delete function use the same way, give delete function a argument obj: obj is accommodationID, for use in delete function. “/DeleteAccommodationServlet” uses this id to find corresponding place in database and let it sold out.If it success, alert information to tell uses they have deleted successfully, then reload page .

After got the accommodationID, “/DeleteAccommodationServlet” will set ‘whetherComplete’ attribute of that accommodation to be 2 and return success to the frontend. If the ‘whetherComplete’ of input accommodation is already 2, then return failed signal to the frontend.

```
function deleteitem(s){
    var accommodationID = s;
    $.ajax({
        type: "GET",
        url: "/Accommodation/DeleteAccommodationServlet",
        dataType: "json",
        data: {accommodationID: accommodationID},
        success: function (data) {
            alert("You had delete" +" "+ accommodationID + " " +"Accommodation!");
            window.location.reload();
        },
        error: function(error){
            alert(error);
        }

    });
    return false;
}
```

Order Information:

Order list page use the same way to display all order with item information: Window entry function, use “BoughtOrderServlet”, give it a username and it will return back all information we needed, then display in suitable place.

After backend received userName, it will search all data from accommodation order table where buyer name is input user. Then return a list of order to front end.

Change Password:

In Change Password, use UpdateUserBasicServlet to change password,

```
function change(){
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = uname = pair[1];
    var old = document.getElementById("old").value;
    var new1 = document.getElementById("new").value;

    $.ajax({
        type: "POST",
        url: "/Accommodation/UpdateUserBasicServlet",
        data: {userName:uname,oldPassword:old,newPassword:new1},
        dataType: "json",
        success: function(data){
            if(data.state == 1){
                alert("success");
                console.log(data)
                window.location.href="homepage.html"
                $.ajax({
                    type: "POST",
                    url: "/Accommodation/LogOutServlet",
                    dataType: "json",
                    data: {userName:uname},
                    success: function (msg) {
                        window.location.href="homepage.html"

                    },
                    error: function(error){
                        console.log(error)
                    }
                })
            }else{
                alert("Wrong Old Password! ");
                return false;
            }
```

Firstly, servlet will get userName, new password and old password from frontend, then check whether the old password user typed matches the password in the database. If user typed in correct old password, then encrypt new password and stored into the database.
If servlet return back data.state = 0, it means old password is wrong, alert information to user “Wrong Old Password!”
if servlet return back data.state = 0,it means change password successfully, jump back to homepage.

Log out:

After frontend called “/LogOutServlet”, this servlet will check Http Sessions from browser, if any session contains user login information, then remove this session and return number “1” to frontend to show user logged out successfully. On the other hand, if there is no sessions about login, then return “0” to frontend to show user log out failed.

```
function logout() {
    var arg1 = window.location.toString().split('?')[1];
    var pair = arg1.split("=");
    var uname = pair[1];

    $.ajax({
        type: "POST",
        url: "/Accommodation/LogOutServlet",
        dataType: "json",
        data: {userName:uname},
        success: function (msg) {
            window.location.href="homepage.html"

        },
        error: function(error){
            console.log(error)
        }


    })

}
```

Third Party Functionalities

Google Map Module

In this part we used google map to help. Enter a location position into google map and search, click share, then choose embed a map, copy the link and add into suitable places on html. It will display a google map with wanted position.

Implementation HighLights and Challenges

1. AES Encryption and Decryption (Success):
   Using “1122334455667788” as key, then get SecretKeySpec by using javax.crypto. After getting the SecretKeySpec, the cipher could be generated. It should be mentioned that the cipher returned should be in Base64.
2. Generate random name for pictures (Success):

There are two methods. The first one is using static method in UUID class to generate random name for user images then store in the localhost. The second method is create a ArrayList to store all the existing filename then using the random method from RandomStringUtils class to generate the random name. After the name genrated, check whether it is exists. If the name exists, generate a new one until it does not match any existing name.

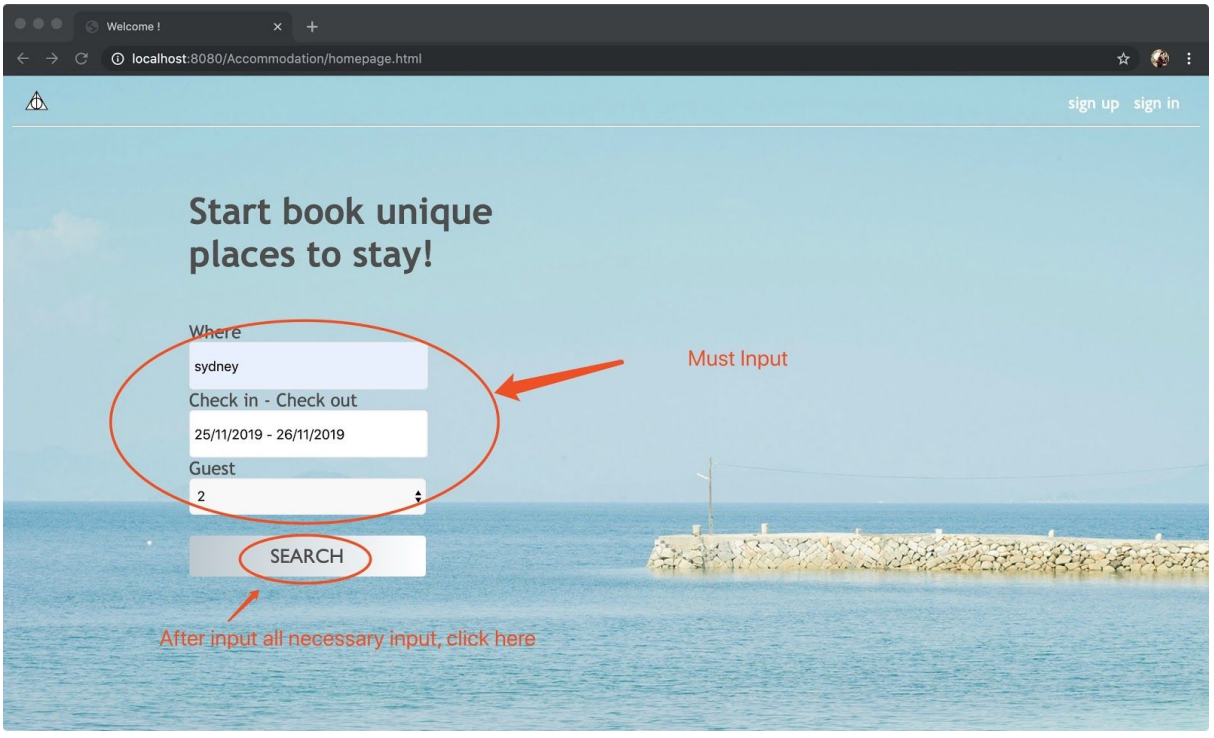
```
String fileName = RandomStringUtils.random( count: 20, chars: "abcdefghijklmnopqrstuvwxyz1234567890");
request.setAttribute( s: "realFileName", fileName);

String basePath = "/Users/yimingdong/COMP4920/AuctionTrade/web/images";
String relativePath = "images/" + fileName;
File file = new File(basePath, fileName);
```

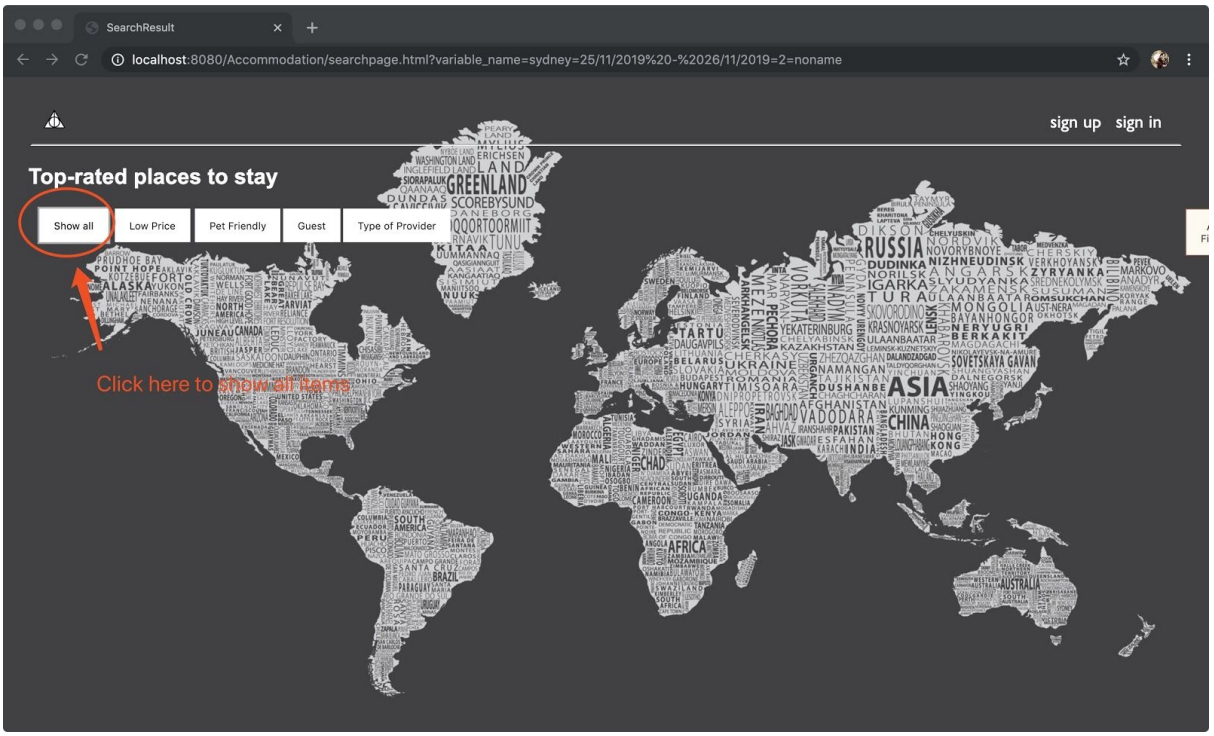
3. Location base searching (Failed):
   Generate a table with all post code and relative suburb grabbed from the website. Then add the latitude and longitude of each suburb to the database. After the frontend get the latitude and longitude of user device, the backend can return all the accommodations nearby. The database is created and relevant servlet created. However frontend could not have enough time to connect backend so this function failed.

User Documentation

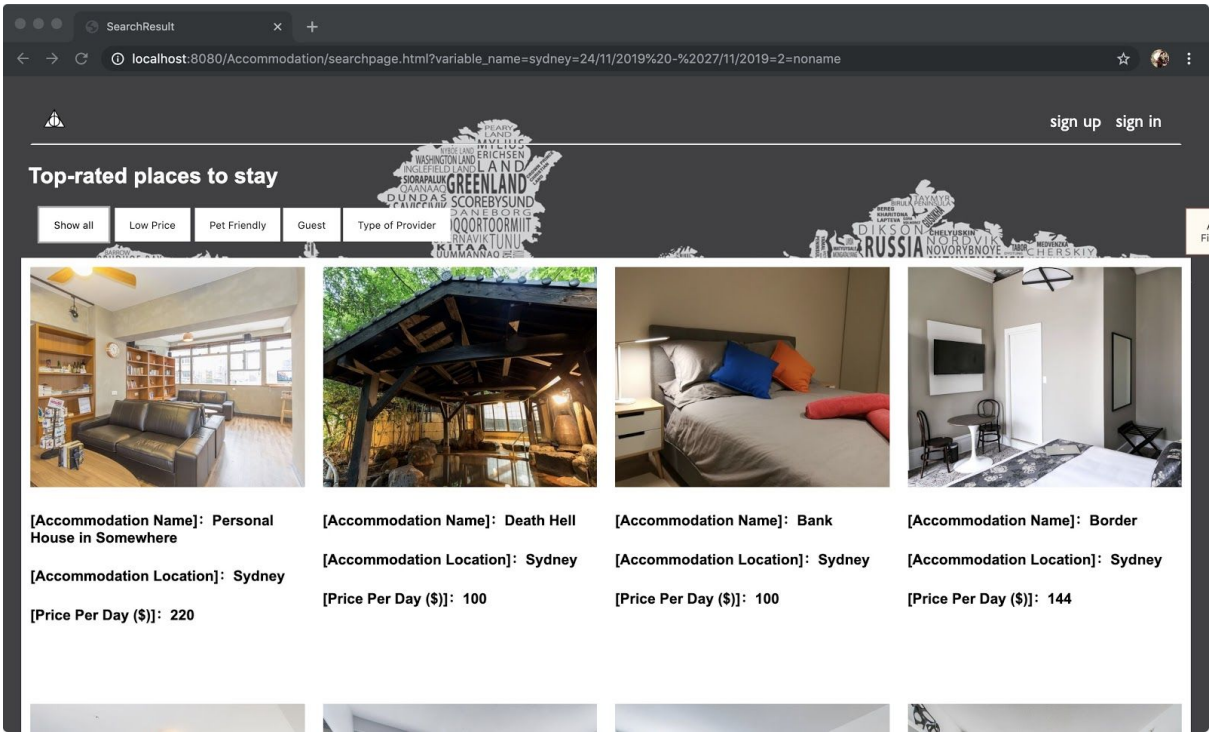
This is the homepage of the website, users can do searching before they log in.



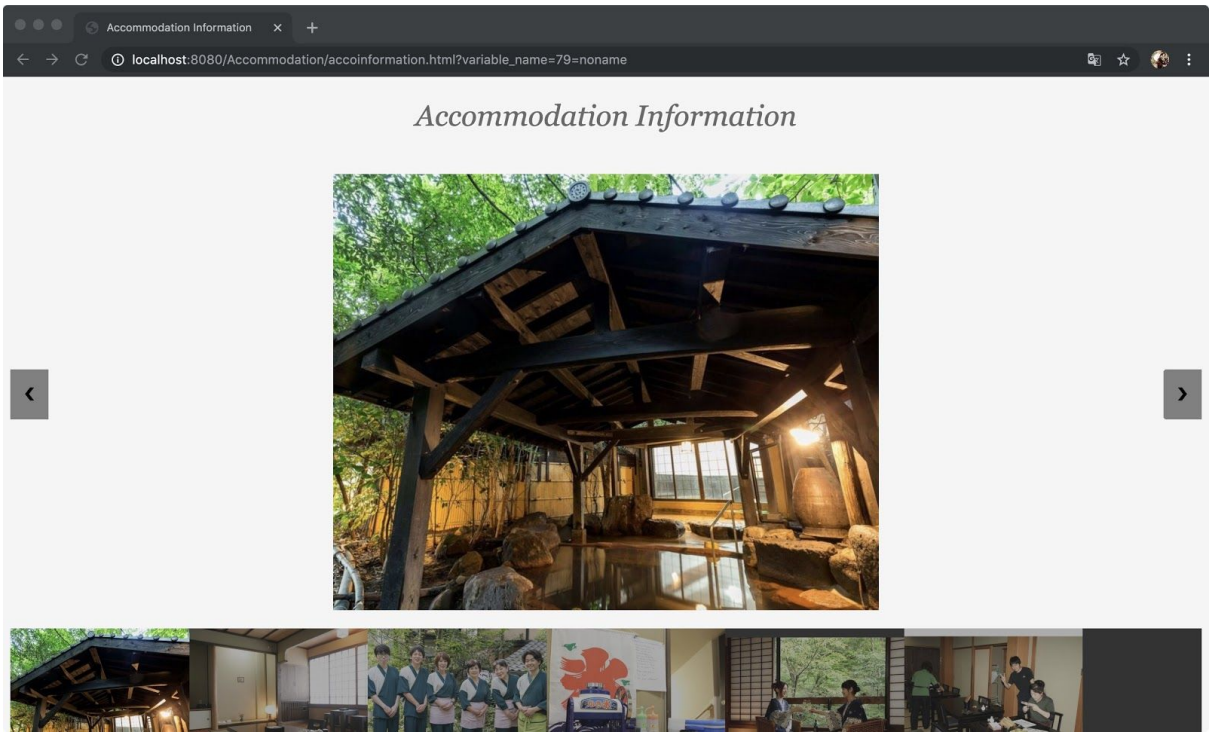
To see the search result, please click show all button



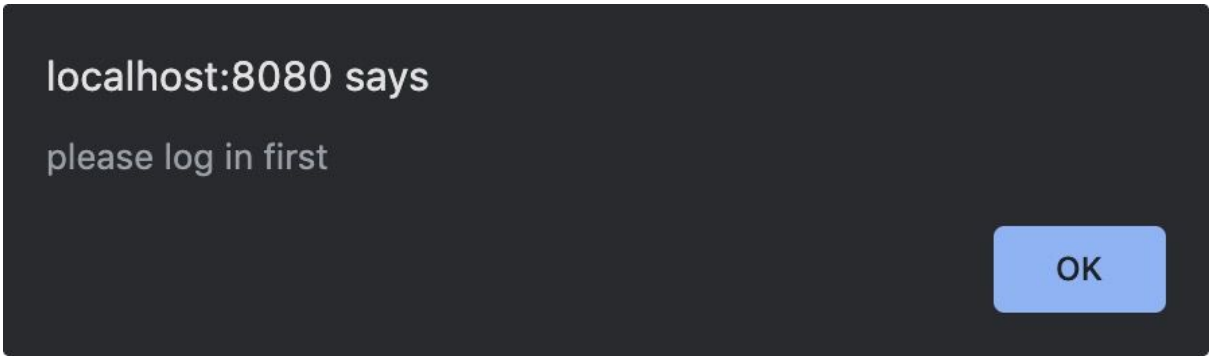
After set the filtering, please click “apply filter” to update, and click show all for result.

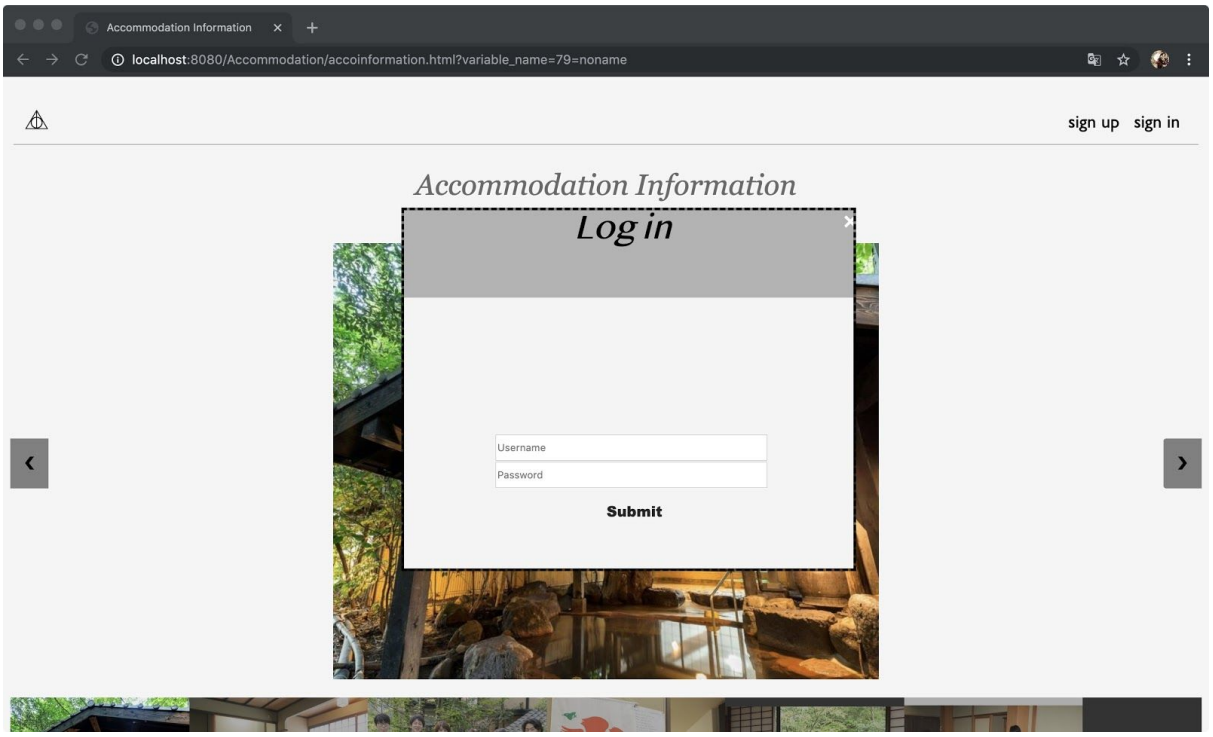


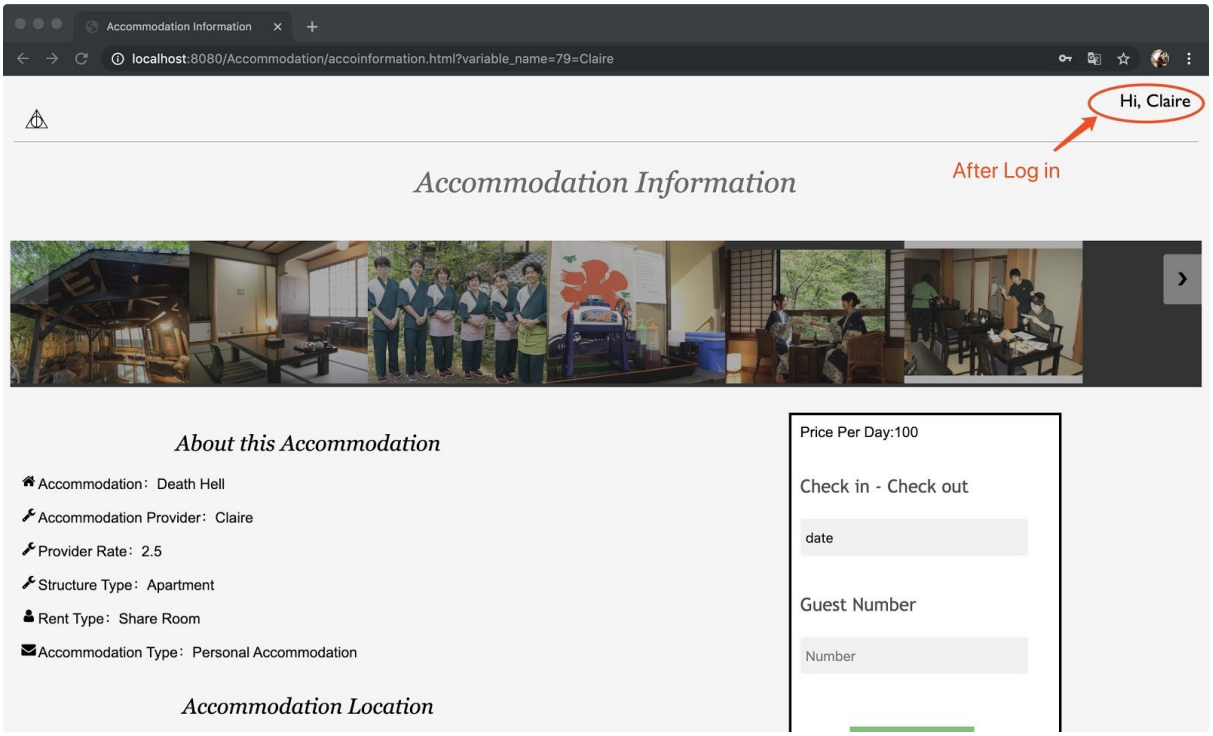
This is the page to display accommocation information.

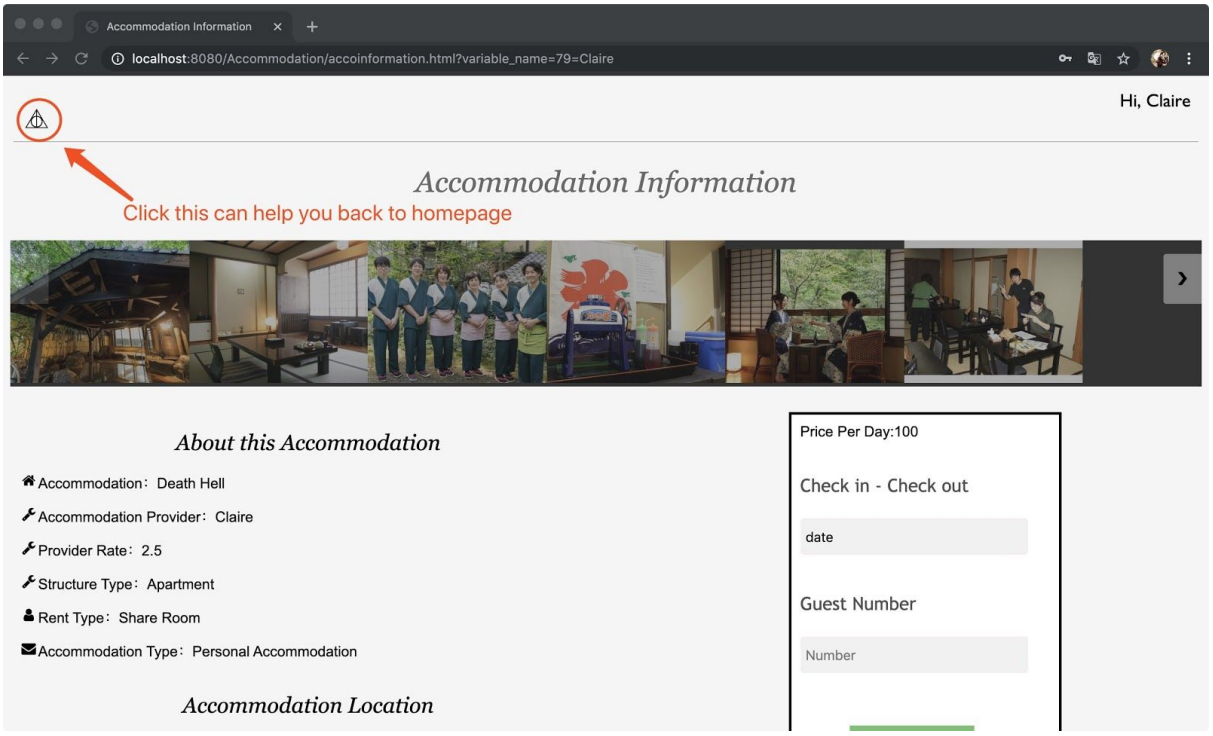


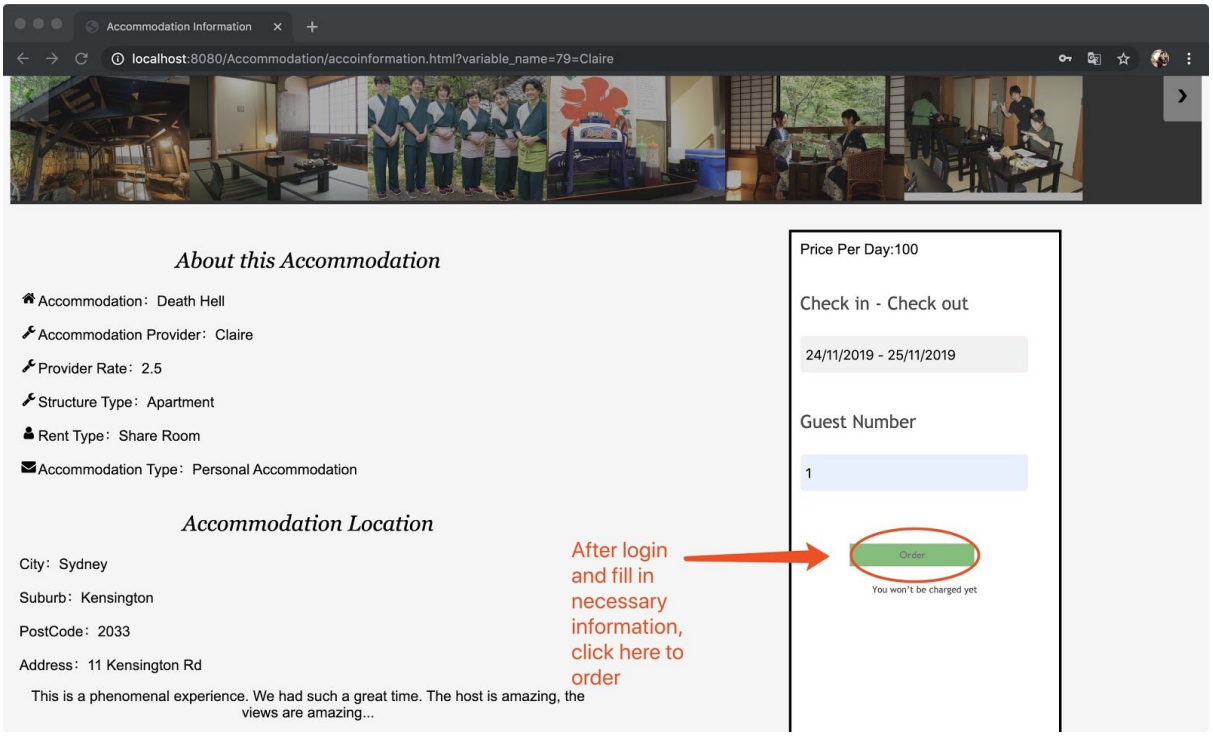
Users can also order in this page if they logged in. The login module can popup in this page.

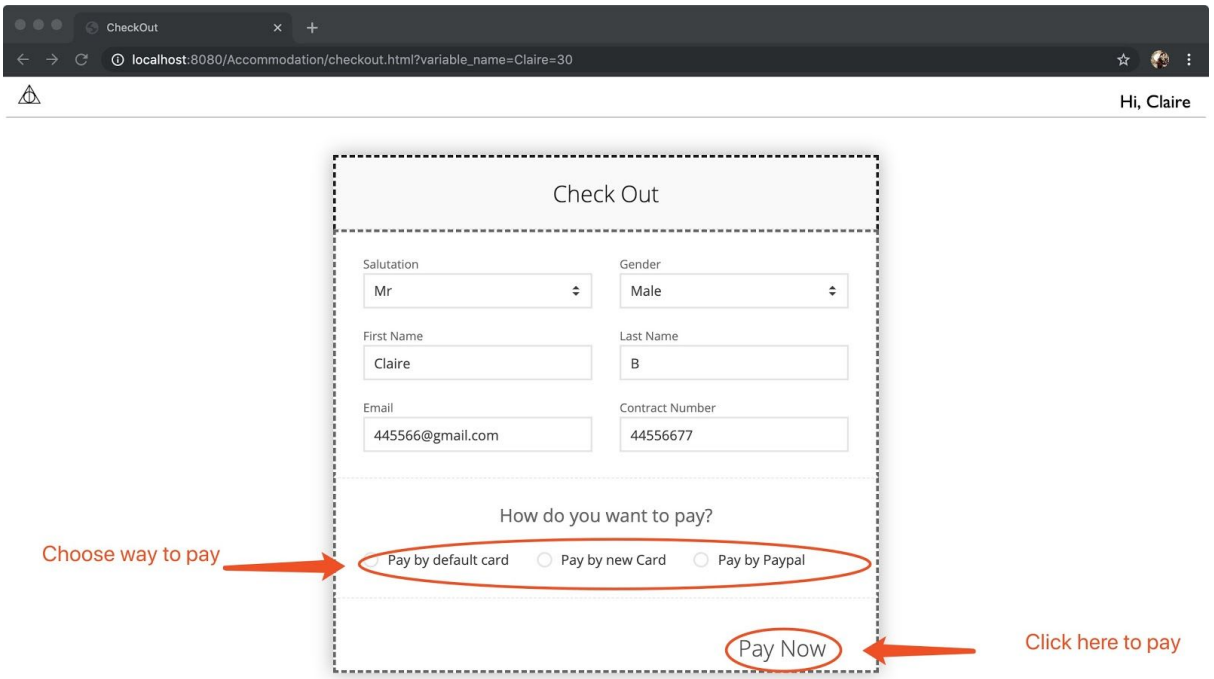


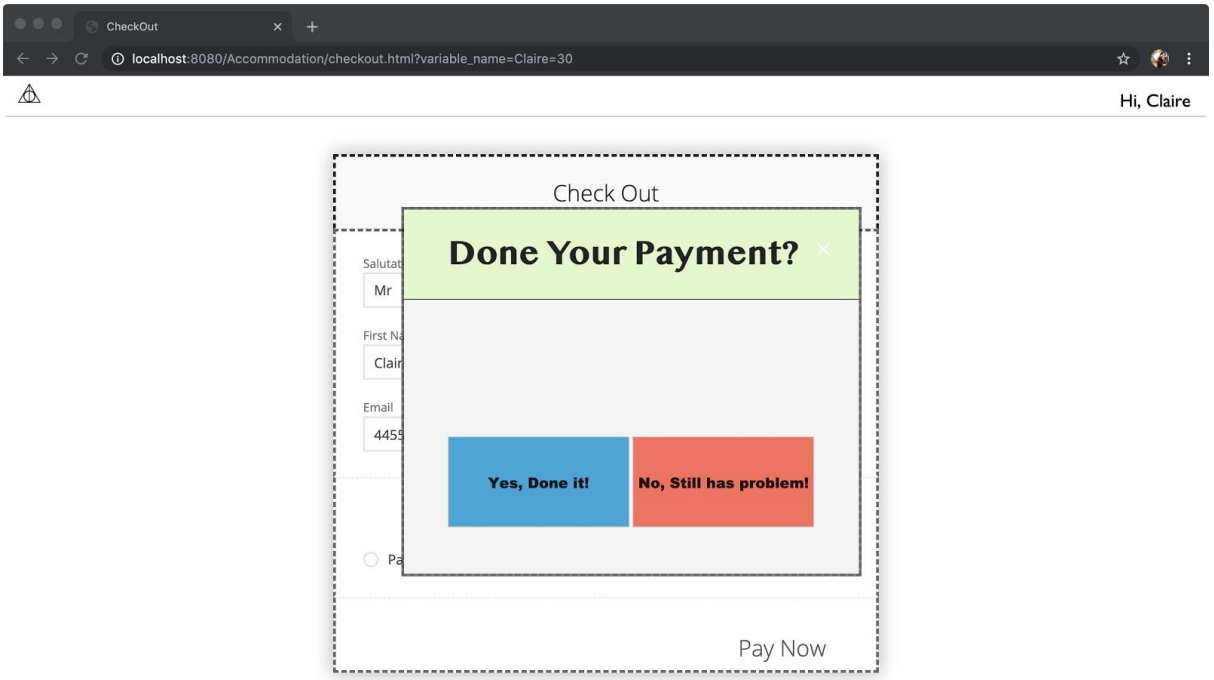

Accommodation Information
localhost:8080/Accommodation/accoinformation.html?variable_name=79=noname
sign up
sign in
Accommodation Information
Log in
Username
Password
Submit

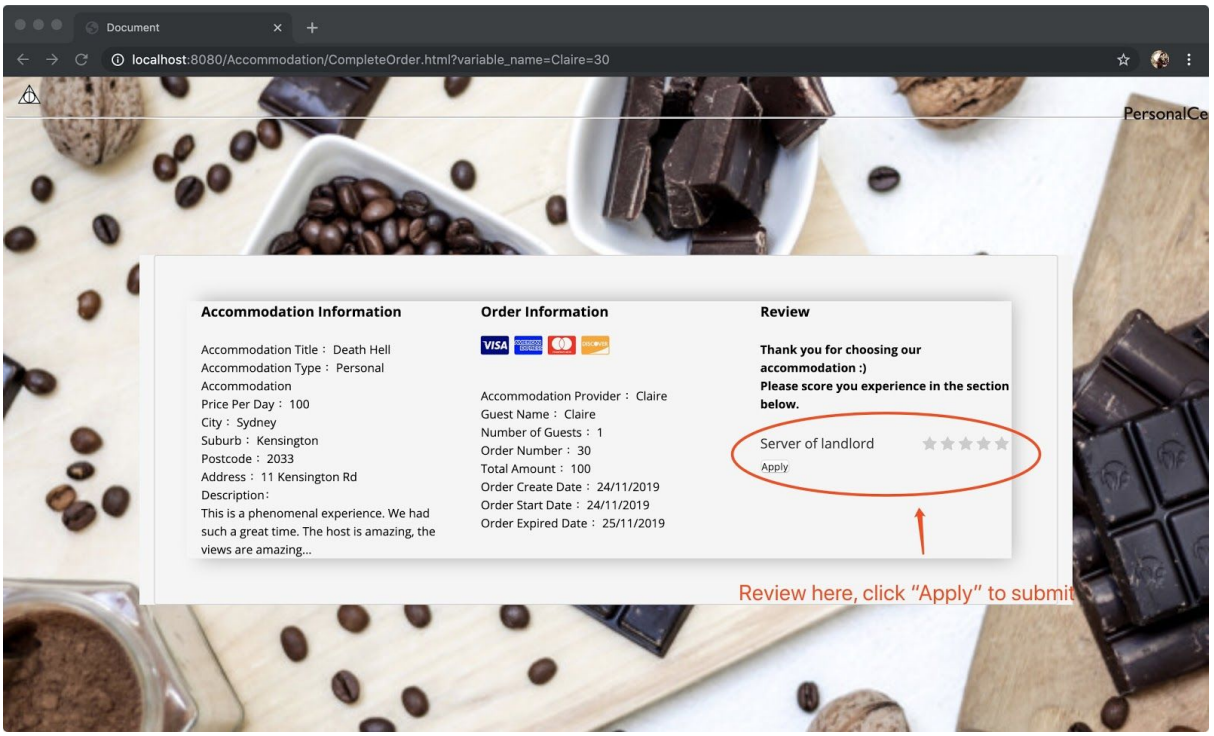

Accommodation Information
localhost:8080/Accommodation/accoinformation.html?variable_name=79=Claire
Hi, Claire
After Log in
Accommodation Information
About this Accommodation
Accommodation: Death Hell
Accommodation Provider: Claire
Provider Rate: 2.5
Structure Type: Apartment
Rent Type: Share Room
Accommodation Type: Personal Accommodation
Accommodation Location
Price Per Day:100
Check in - Check out
date
Guest Number
Number

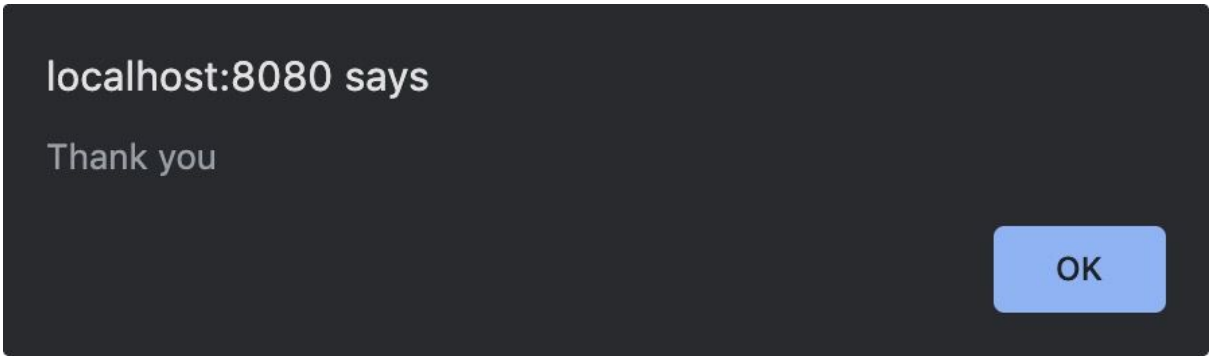

Accommodation Information
localhost:8080/Accommodation/accoinformation.html?variable_name=79=Claire
Hi, Claire
Accommodation Information
Click this can help you back to homepage
About this Accommodation
Accommodation: Death Hell
Accommodation Provider: Claire
Provider Rate: 2.5
Structure Type: Apartment
Rent Type: Share Room
Accommodation Type: Personal Accommodation
Accommodation Location
Price Per Day:100
Check in - Check out
date
Guest Number
Number

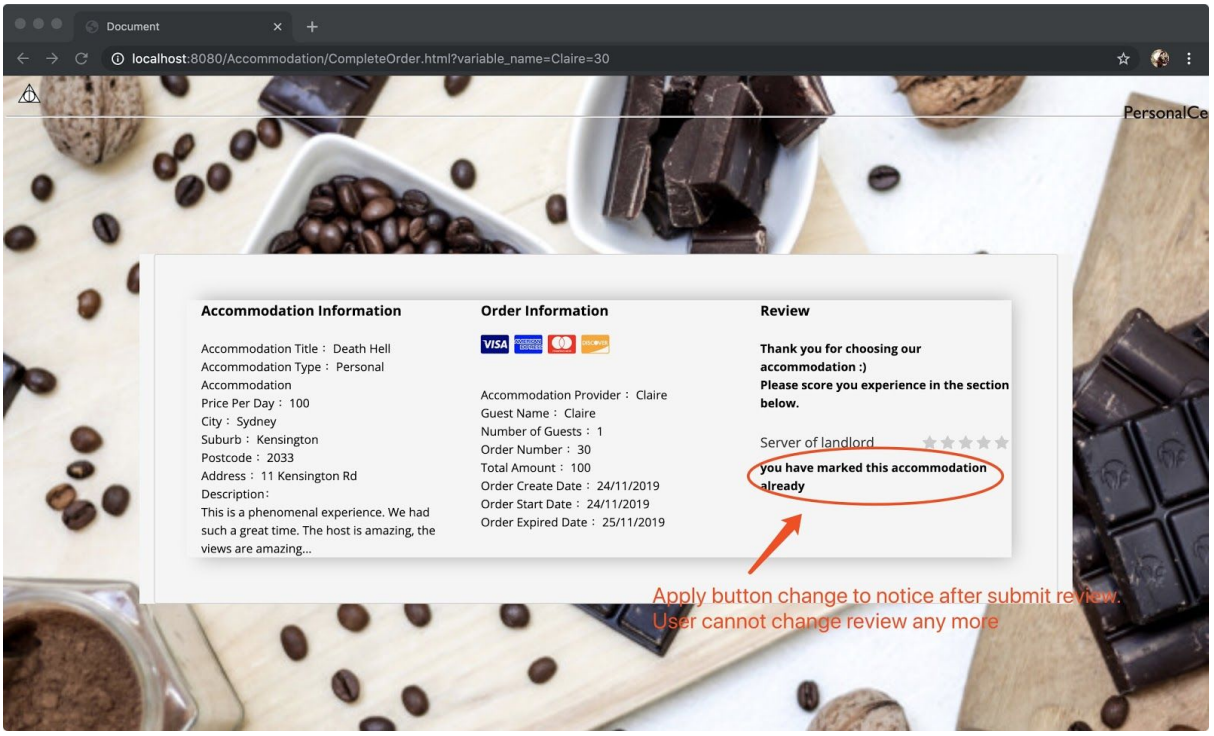

Accommodation Information
localhost:8080/Accommodation/accoinformation.html?variable_name=79=Claire
About this Accommodation
Accommodation: Death Hell
Accommodation Provider: Claire
Provider Rate: 2.5
Structure Type: Apartment
Rent Type: Share Room
Accommodation Type: Personal Accommodation
Accommodation Location
City: Sydney
Suburb: Kensington
PostCode: 2033
Address: 11 Kensington Rd
This is a phenomenal experience. We had such a great time. The host is amazing, the views are amazing...
After login and fill in necessary information, click here to order
Price Per Day:100
Check in - Check out
24/11/2019 - 25/11/2019
Guest Number
1
Order
You won't be charged yet

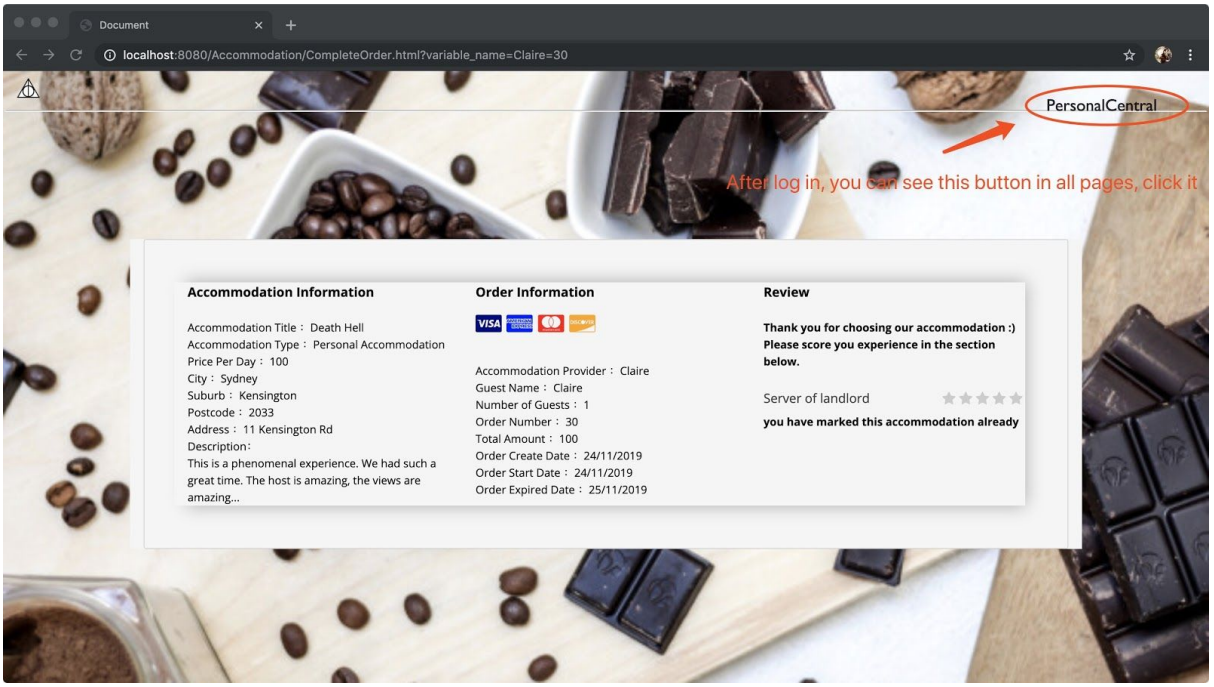

CheckOut
localhost:8080/Accommodation/checkout.html?variable_name=Claire=30
Hi, Claire
Check Out
Salutation
Mr
Gender
Male
First Name
Claire
Last Name
B
Email
445566@gmail.com
Contract Number
44556677
How do you want to pay?
Choose way to pay
Pay by default card
Pay by new Card
Pay by Paypal
Pay Now
Click here to pay

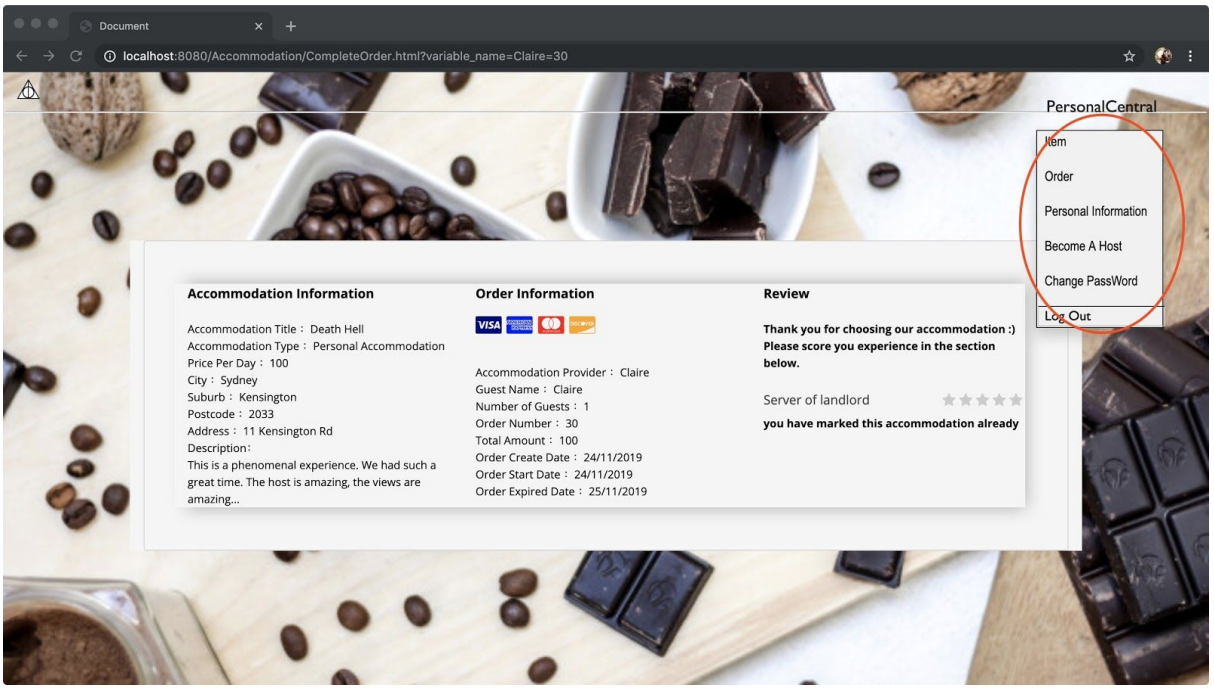

CheckOut
localhost:8080/Accommodation/checkout.html?variable_name=Claire=30
Hi, Claire
Check Out
Done Your Payment?
Yes, Done it!
No, Still has problem!
Pay Now


Document
localhost:8080/Accommodation/CompleteOrder.html?variable_name=Claire=30
PersonalCe
Accommodation Information
Accommodation Title： Death Hell
Accommodation Type： Personal Accommodation
Price Per Day： 100
City： Sydney
Suburb： Kensington
Postcode： 2033
Address： 11 Kensington Rd
Description：
This is a phenomenal experience. We had such a great time. The host is amazing, the views are amazing...
Order Information
Accommodation Provider： Claire
Guest Name： Claire
Number of Guests： 1
Order Number： 30
Total Amount： 100
Order Create Date： 24/11/2019
Order Start Date： 24/11/2019
Order Expired Date： 25/11/2019
Review
Thank you for choosing our accommodation :)
Please score you experience in the section below.
Server of landlord
Apply
Review here, click “Apply” to submit


localhost:8080 says
Thank you
OK


Document
localhost:8080/Accommodation/CompleteOrder.html?variable_name=Claire=30
PersonalCe
Accommodation Information
Accommodation Title： Death Hell
Accommodation Type： Personal Accommodation
Price Per Day： 100
City： Sydney
Suburb： Kensington
Postcode： 2033
Address： 11 Kensington Rd
Description：
This is a phenomenal experience. We had such a great time. The host is amazing, the views are amazing...
Order Information
Accommodation Provider： Claire
Guest Name： Claire
Number of Guests： 1
Order Number： 30
Total Amount： 100
Order Create Date： 24/11/2019
Order Start Date： 24/11/2019
Order Expired Date： 25/11/2019
Review
Thank you for choosing our accommodation :)
Please score you experience in the section below.
Server of landlord
you have marked this accommodation already
Apply button change to notice after submit review.
User cannot change review any more


Document
localhost:8080/Accommodation/CompleteOrder.html?variable_name=Claire=30
PersonalCentral
After log in, you can see this button in all pages, click it
Accommodation Information
Accommodation Title： Death Hell
Accommodation Type： Personal Accommodation
Price Per Day： 100
City： Sydney
Suburb： Kensington
Postcode： 2033
Address： 11 Kensington Rd
Description:
This is a phenomenal experience. We had such a great time. The host is amazing, the views are amazing...
Order Information
Accommodation Provider： Claire
Guest Name： Claire
Number of Guests： 1
Order Number： 30
Total Amount： 100
Order Create Date： 24/11/2019
Order Start Date： 24/11/2019
Order Expired Date： 25/11/2019
Review
Thank you for choosing our accommodation :) Please score you experience in the section below.
Server of landlord
you have marked this accommodation already


Document
localhost:8080/Accommodation/CompleteOrder.html?variable_name=Claire=30
PersonalCentral
Item
Order
Personal Information
Become A Host
Change PassWord
Log Out
Accommodation Information
Accommodation Title： Death Hell
Accommodation Type： Personal Accommodation
Price Per Day： 100
City： Sydney
Suburb： Kensington
Postcode： 2033
Address： 11 Kensington Rd
Description:
This is a phenomenal experience. We had such a great time. The host is amazing, the views are amazing...
Order Information
Accommodation Provider： Claire
Guest Name： Claire
Number of Guests： 1
Order Number： 30
Total Amount： 100
Order Create Date： 24/11/2019
Order Start Date： 24/11/2019
Order Expired Date： 25/11/2019
Review
Thank you for choosing our accommodation :) Please score you experience in the section below.
Server of landlord
you have marked this accommodation already

ItemList

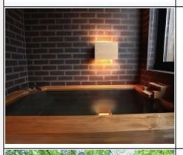
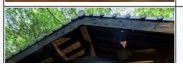
localhost:8080/Accommodation/itemList.html?variable_name=Claire

Item List

All Items List | Rented Items List

view this accommodation details

delete this accommdation

	Personal House in Somewhere	Click to View	Delete
	Try to live in	Click to View	Delete
	Hohoho	Click to View	Delete
			

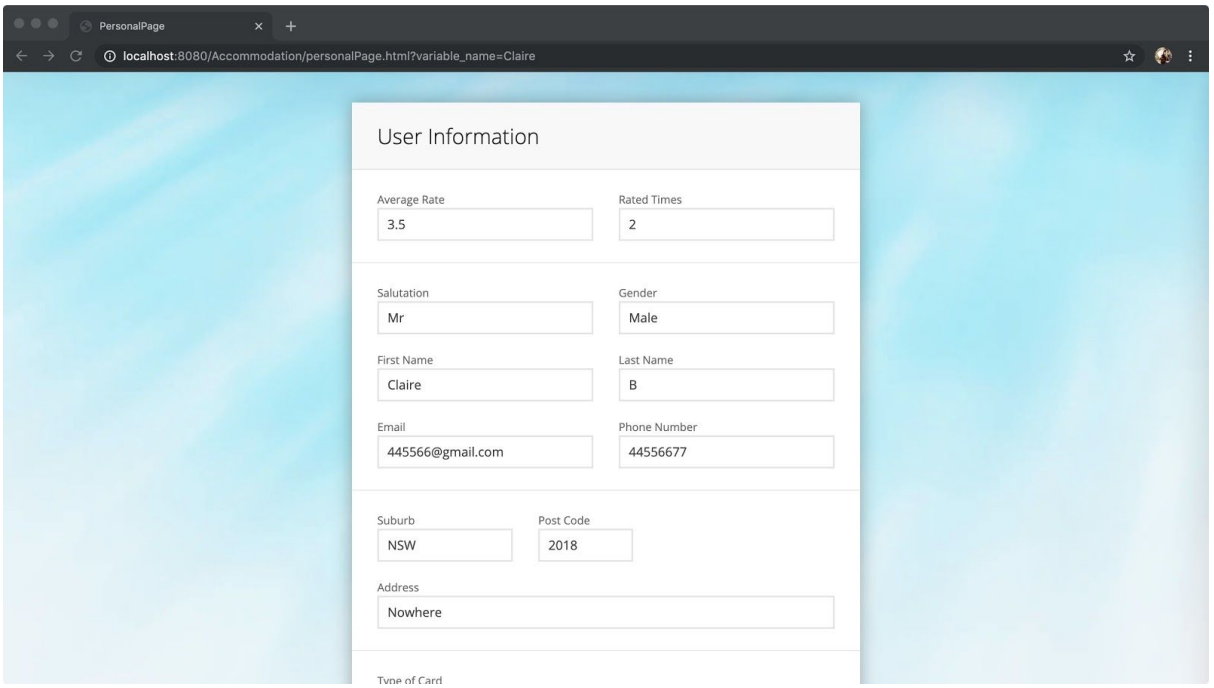
OrderList

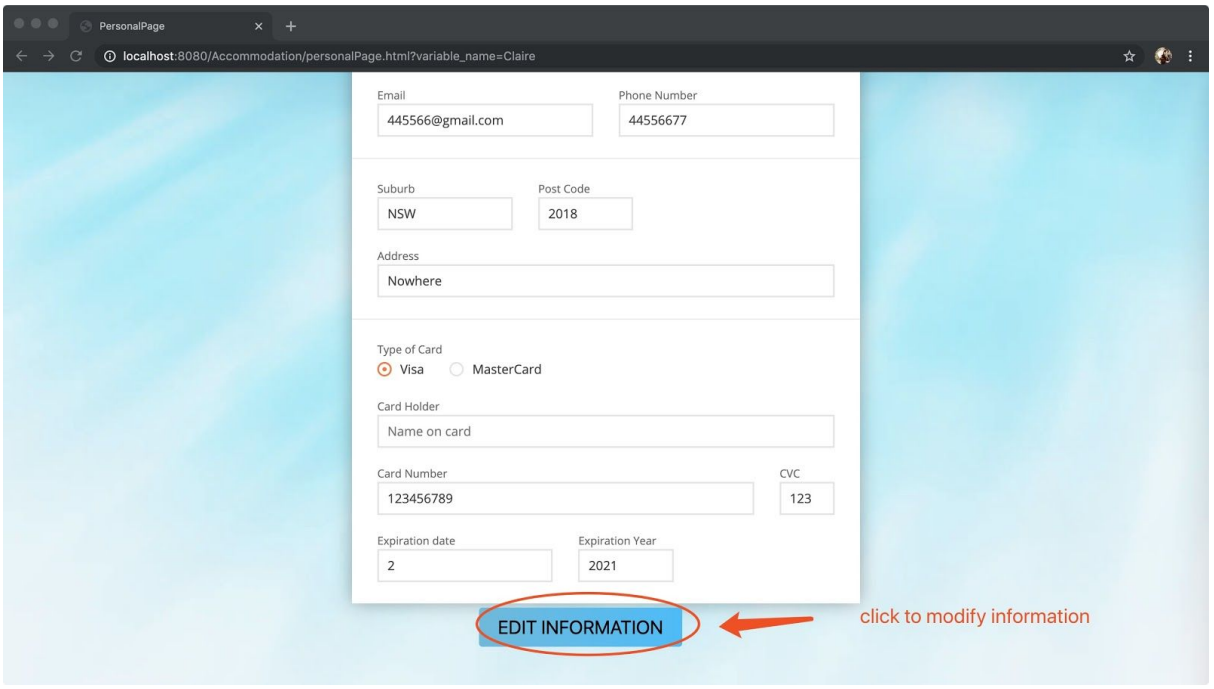
localhost:8080/Accommodation/orderList.html?variable_name=Claire

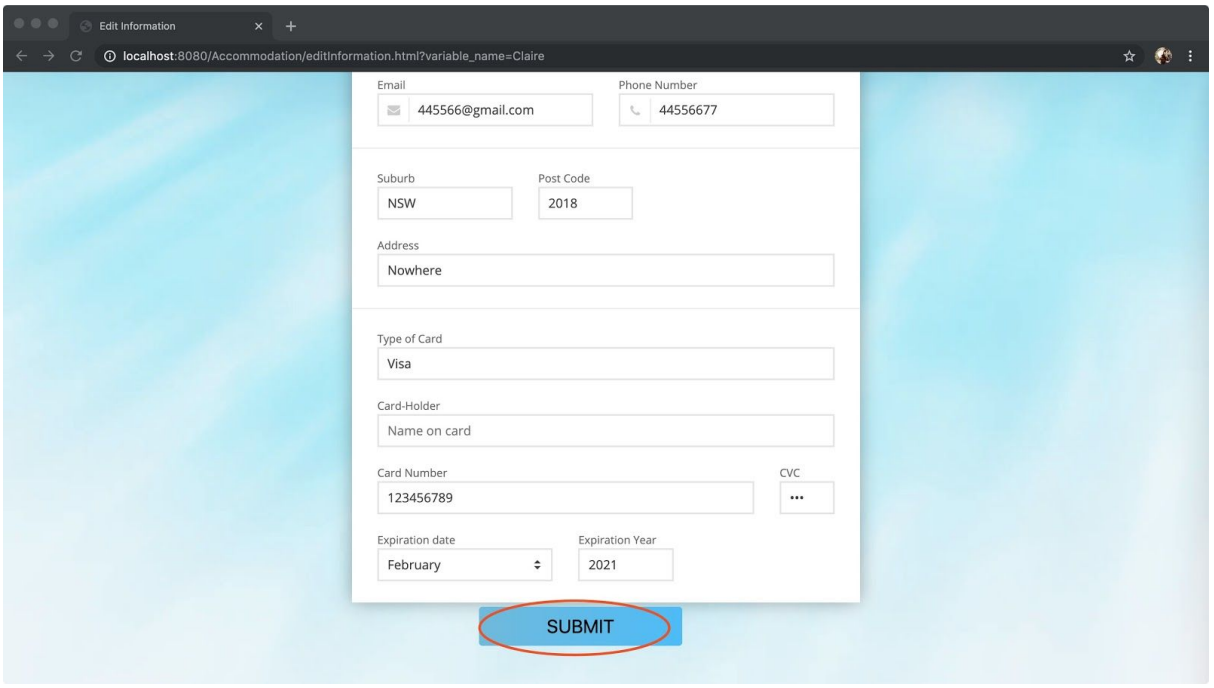
PersonalCentral

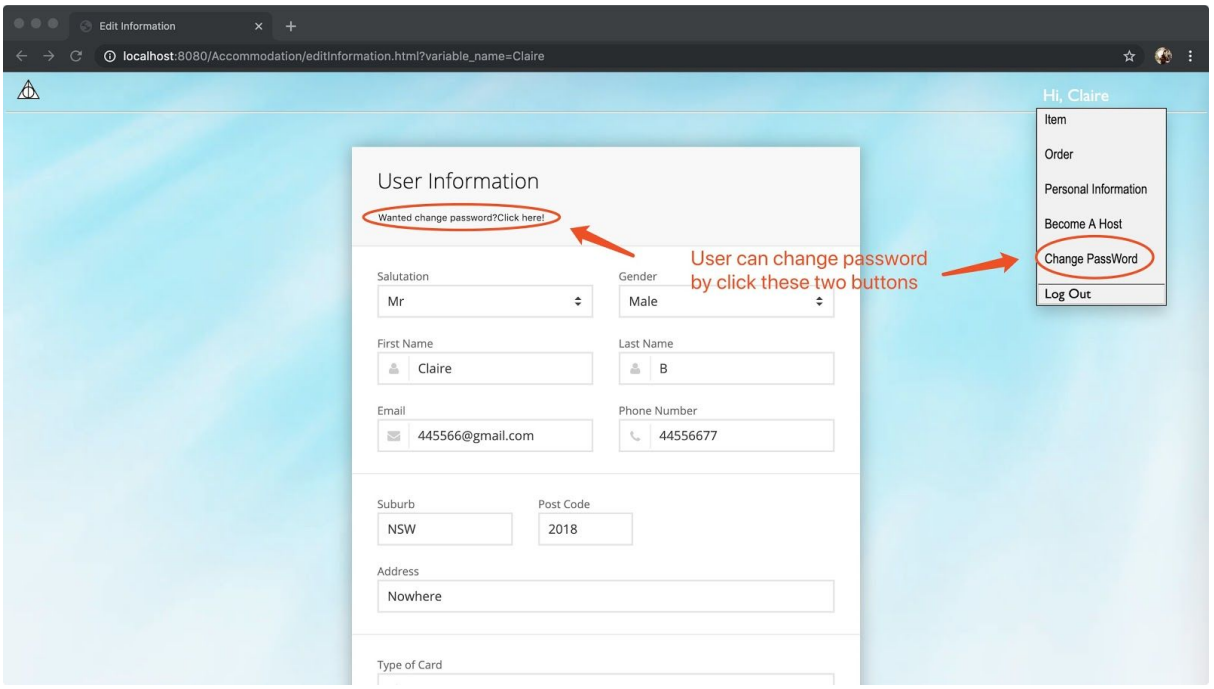
Order List

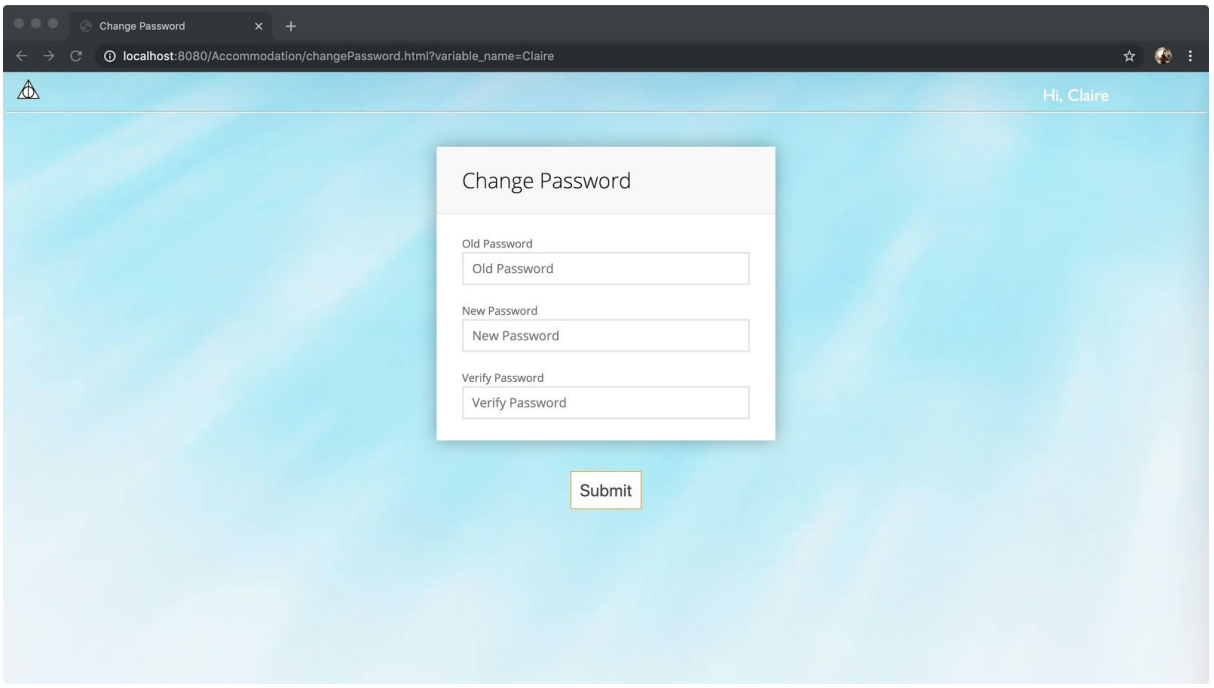
OrderId	Buyer Name	Order Create Date	Start Date	End Date	Price	
30	Claire	24/11/2019	24/11/2019	25/11/2019	100	Review

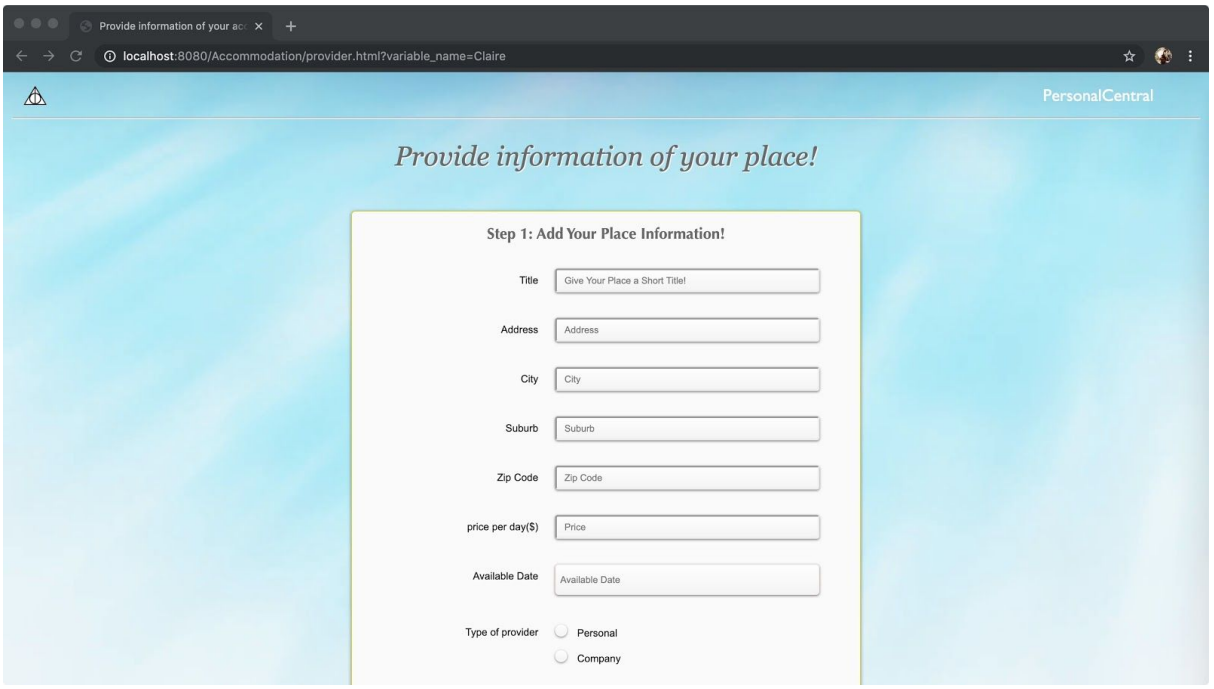

PersonalPage
localhost:8080/Accommodation/personalPage.html?variable_name=Claire
User Information
Average Rate
3.5
Rated Times
2
Salutation
Mr
Gender
Male
First Name
Claire
Last Name
B
Email
445566@gmail.com
Phone Number
44556677
Suburb
NSW
Post Code
2018
Address
Nowhere
Type of Card

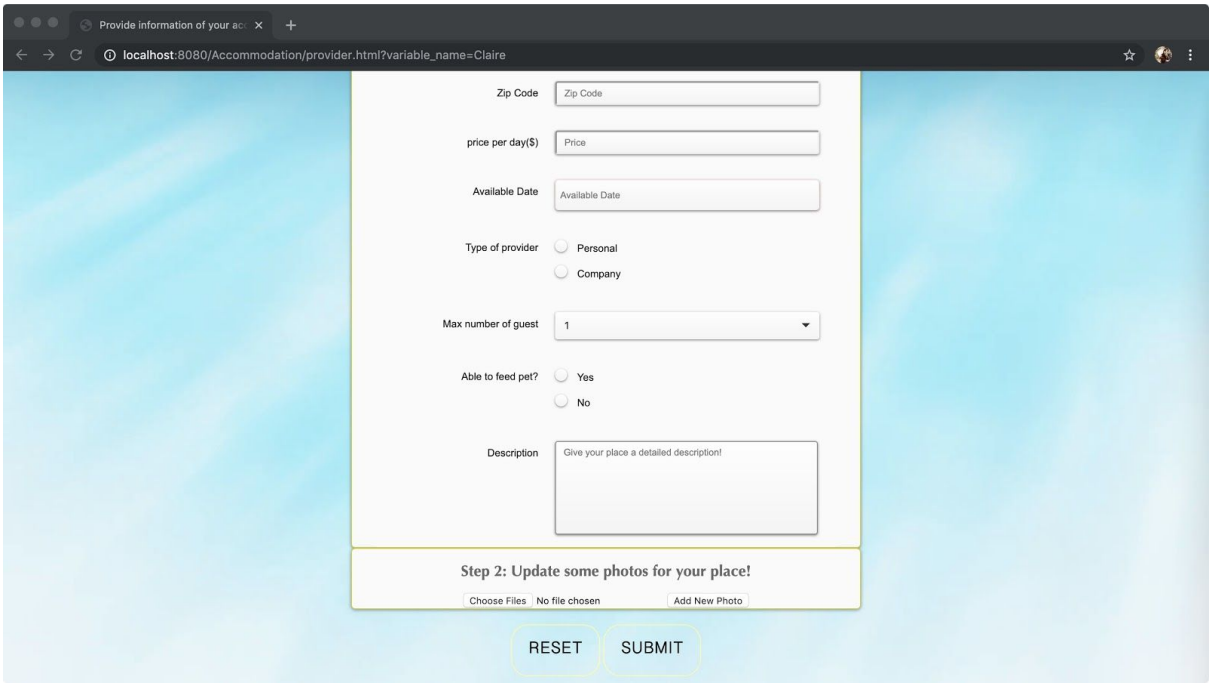

PersonalPage
localhost:8080/Accommodation/personalPage.html?variable_name=Claire
Email
445566@gmail.com
Phone Number
44556677
Suburb
NSW
Post Code
2018
Address
Nowhere
Type of Card
Visa
MasterCard
Card Holder
Name on card
Card Number
123456789
CVC
123
Expiration date
2
Expiration Year
2021
EDIT INFORMATION
click to modify information

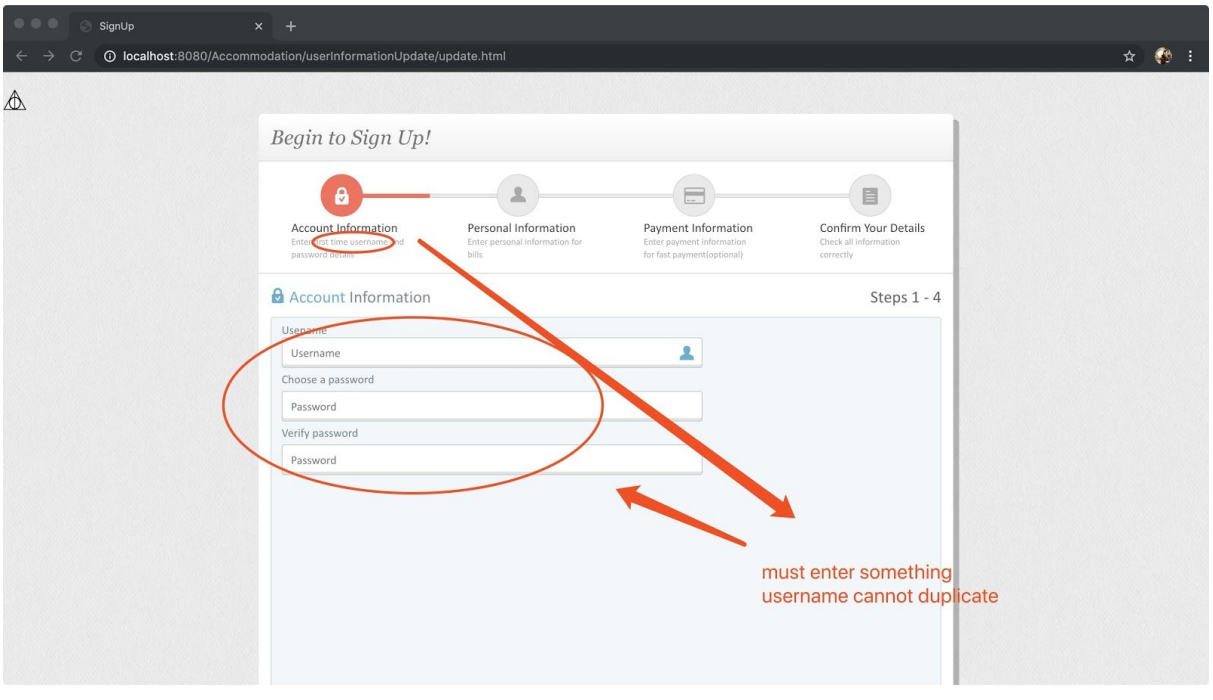

Edit Information
localhost:8080/Accommodation/editInformation.html?variable_name=Claire
Email
445566@gmail.com
Phone Number
44556677
Suburb
NSW
Post Code
2018
Address
Nowhere
Type of Card
Visa
Card-Holder
Name on card
Card Number
123456789
CVC
Expiration date
February
Expiration Year
2021
SUBMIT

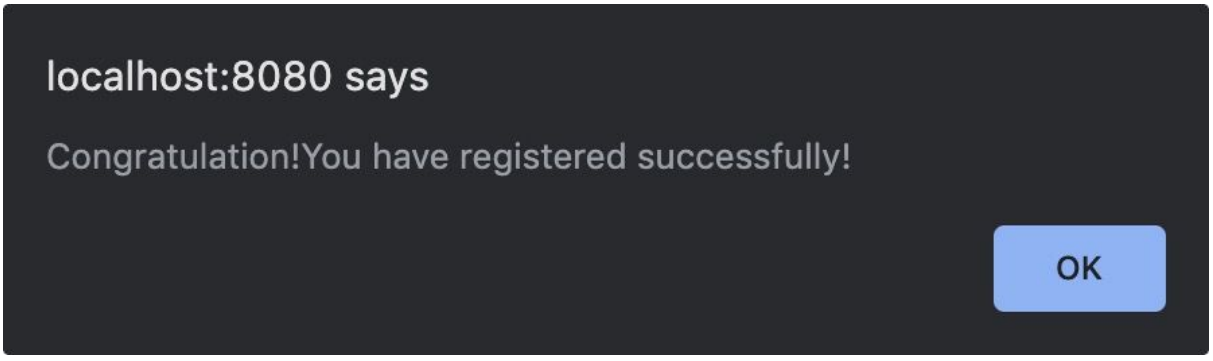

Edit Information
localhost:8080/Accommodation/editInformation.html?variable_name=Claire
Hi, Claire
Item
Order
Personal Information
Become A Host
Change PassWord
Log Out
User Information
Wanted change password?Click here!
User can change password
by click these two buttons
Salutation
Mr
Gender
Male
First Name
Claire
Last Name
B
Email
445566@gmail.com
Phone Number
44556677
Suburb
NSW
Post Code
2018
Address
Nowhere
Type of Card

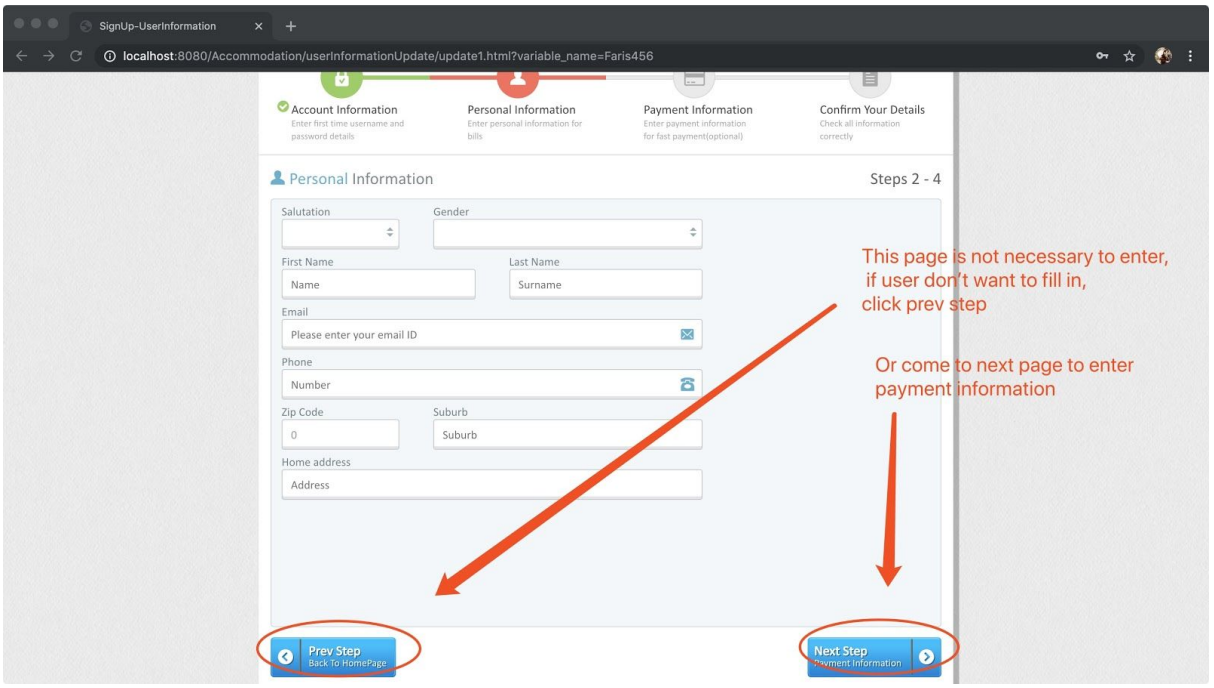

Change Password
localhost:8080/Accommodation/changePassword.html?variable_name=Claire
Hi, Claire
Change Password
Old Password
Old Password
New Password
New Password
Verify Password
Verify Password
Submit

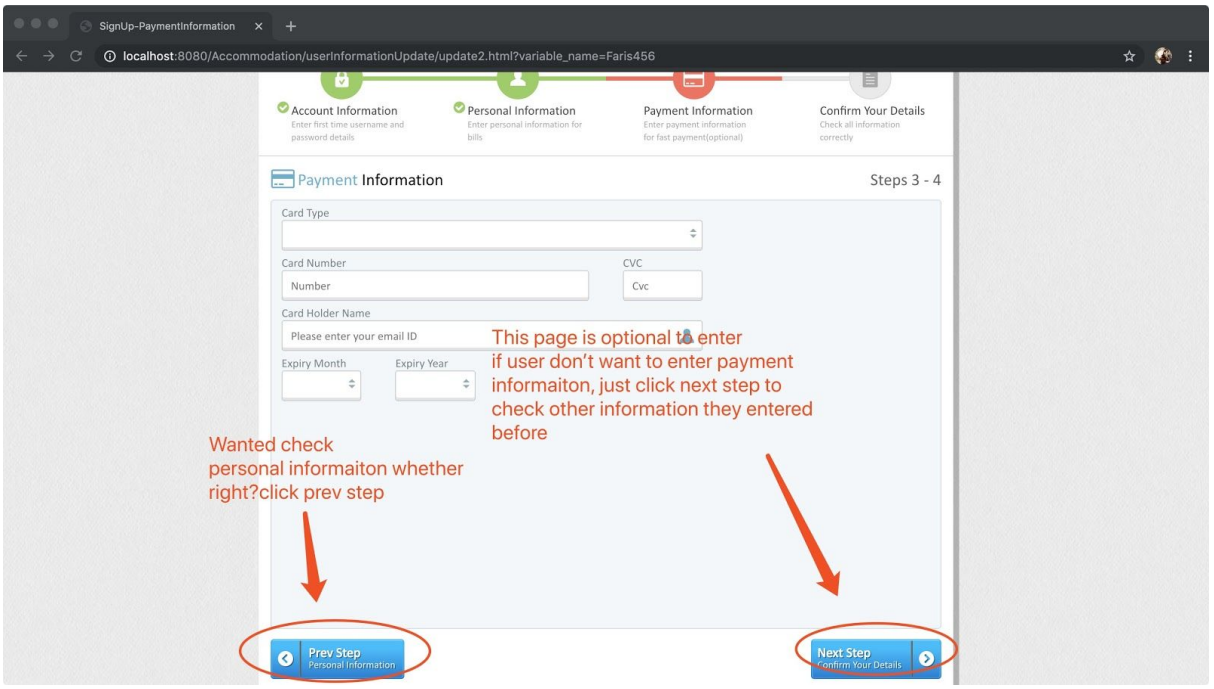

Provide information of your ac
localhost:8080/Accommodation/provider.html?variable_name=Claire
PersonalCentral
Provide information of your place!
Step 1: Add Your Place Information!
Title
Give Your Place a Short Title!
Address
Address
City
City
Suburb
Suburb
Zip Code
Zip Code
price per day($)
Price
Available Date
Available Date
Type of provider
Personal
Company

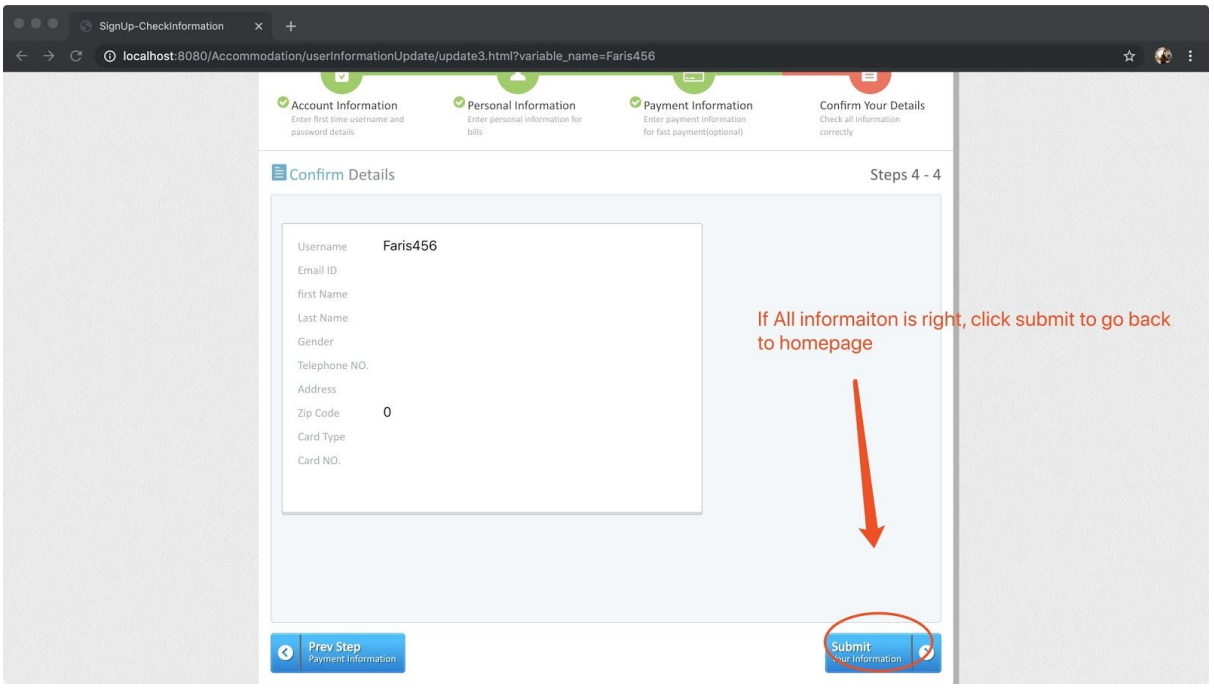

Provide information of your ac
localhost:8080/Accommodation/provider.html?variable_name=Claire
Zip Code
Zip Code
price per day($)
Price
Available Date
Available Date
Type of provider
Personal
Company
Max number of guest
1
Able to feed pet?
Yes
No
Description
Give your place a detailed description!
Step 2: Update some photos for your place!
Choose Files
No file chosen
Add New Photo
RESET
SUBMIT

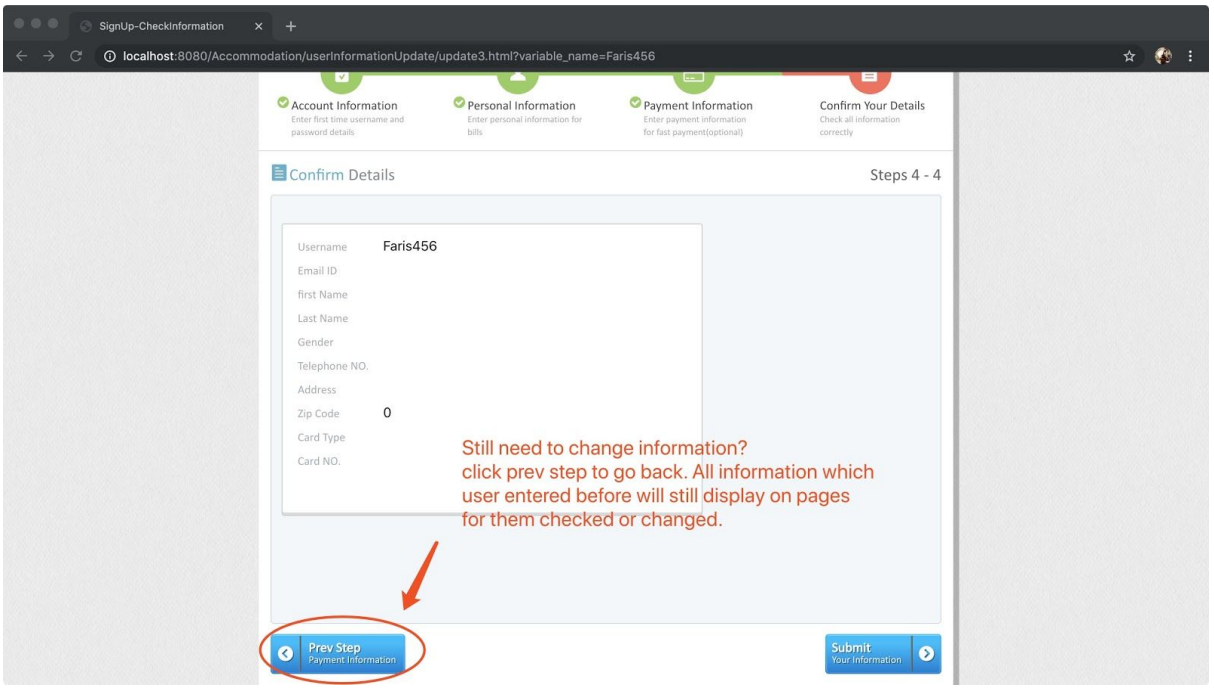

SignUp
localhost:8080/Accommodation/userInformationUpdate/update.html
Begin to Sign Up!
Account Information
Personal Information
Payment Information
Confirm Your Details
Account Information
Steps 1 - 4
Username
Choose a password
Password
Verify password
Password
must enter something
username cannot duplicate


localhost:8080 says
Congratulation!You have registered successfully!
OK


SignUp-UserInformation
localhost:8080/Accommodation/userInformationUpdate/update1.html?variable_name=Faris456
Account Information
Enter first time username and password details
Personal Information
Enter personal information for bills
Payment Information
Enter payment information for fast payment(optional)
Confirm Your Details
Check all information correctly
Personal Information
Steps 2 - 4
Salutation
Gender
First Name
Name
Last Name
Surname
Email
Please enter your email ID
Phone
Number
Zip Code
0
Suburb
Suburb
Home address
Address
This page is not necessary to enter, if user don't want to fill in, click prev step
Or come to next page to enter payment information
Prev Step
Back To HomePage
Next Step
Payment Information


SignUp-PaymentInformation
localhost:8080/Accommodation/userInformationUpdate/update2.html?variable_name=Faris456
Account Information
Enter first time username and password details
Personal Information
Enter personal information for bills
Payment Information
Enter payment information for fast payment(optional)
Confirm Your Details
Check all information correctly
Payment Information
Steps 3 - 4
Card Type
Card Number
Number
CVC
Cvc
Card Holder Name
Please enter your email ID
Expiry Month
Expiry Year
This page is optional to enter if user don't want to enter payment informaiton, just click next step to check other information they entered before
Wanted check personal informaiton whether right?click prev step
Prev Step
Personal Information
Next Step
Confirm Your Details





Reference List

https://www.w3schools.com

https://codecanyon.net/category/css

https://getbootstrap.com/docs/4.0/components/forms/

https://www.runoob.co