# **Gift Delivery Mobile Web App**

# **(Version 1, partially completed)**

# **Explanation of Implemented JavaScript Code**

# **Sign Up page**

**Javascript**

First of all it is to set up an event listener for "pagebeforeshow" event on the document. "pagebeforeshow" is triggered just before a page is shown to user. When this event occurs on an element with the id "signuppage", the fallback function is executed.

$(document).on("pagebeforeshow", "#signuppage", function() {

After that, following line of code sets up an event listener for the "click" event on the element with the id "signupbutton".

$("#signupbutton").on("click", function() {

Next is to validate the "signupform" using JQuery's validation plugin. valid()function checks the form for the correctness of the filled data according to the specified validation rules.

if ($("#signupform").valid()) {

serializeArray() method is used to serialize the form data contained within the form elements selected by a jQuery selector into an array of JavaScript objects.

Each object in the array represents a form field, and it contains two properties name and field.

var formData = $("#signupform").serializeArray();

var userData = {};

Following loop iterates over each field in the formData array and adds it to the userData object. ForEach loop iterates over all of the elements in an array and its "**field.value**" is added to userData object in **"field.name**" key.

formData.forEach(function(field) {

userData[field.name] = field.value;

});

userInfo is set on the browser's local storage consisting userData achieved from iteration.

localStorage.setItem("userInfo", JSON.stringify(userData));

Now the updated array of users is stored back into the local storage after converting it to a JSON string , also updates userInfo in local storage from currently signed up details for ensuring login when switching to homepage. JSON.stringify(allUsers) function in JavaScript converts "**allUsers"** JavaScript object or value to a JSON string.

var allUsers =JSON.parse(localStorage.getItem("allUsers")) || [];

allUsers.push(userData);

localStorage.setItem("allUsers",JSON.stringify(allUsers));

$("#signupform")[0] is retrieving first DOM element that matches the selector **"#signupform"**. In this case, it's HTML form element with the ID "signupform". reset()function is used to clear the element.

$("#signupform")[0].reset();

alert("Registration successful!");

$.mobile.changePage("#homepage");

$.mobile.changePage("#homepage") is to switch the page to home page after completion.

}

});

});

Following blocks define validation rules, messages, and actions to be taken upon form submission.

$("#signupform").validate({

focusInvalid: false,

onkeyup: false,

submitHandler: function (form) {

submitHandler serializes validated form data, incorporates user and item details and saves this information in localStorage.

var formData =$(form).serializeArray();

            var Info = {};

Above two lines of code retrieves the formData and serializes it into an array of objects using jQuery's serializeArray() method. Then, it initializes an empty object called Info.

Following loop iterates over each field in the formData array and adds it to the Info object. ForEach loop iterates over all of the elements in an array and its "**data.value**" is added to userData object in **"data.name**" key.

            formData.forEach(function(data){

                Info[data.name] = data.value;

            });

            var userInfo = JSON.parse(localStorage.getItem("userInfo"));

            Info.customerfName = userInfo.firstName;

            Info.customerlName = userInfo.lastName;

            localStorage.setItem("orderInfo", JSON.stringify(Info));

            if (debug)  alert(JSON.stringify(Info));

Above line of code is a conditional statement that checks whether a variable "debug" is true or not. If debug is true then it displays an alert dialog box containing the JSON string representation of the object Info.

Validation rules for signup form. **Required** key denotes that the following input cannot be true and so on other keys are used to describe the validation rules for form.

rules: {

email: {

required: true,

email: true

},

password: {

required: true,

rangelength: [3, 10]

},

firstName: {

required: true,

rangelength: [1, 15],

validateName: true

},

lastName: {

required: true,

rangelength: [1, 15],

validateName: true

},

phoneNumber: {

required: true,

mobiletxt: true

},

address: {

required: true,

rangelength: [1, 25]

},

postcode: {

required: true,

posttxt: true

},

},

Validation Messages for Signup Form, all the input with its "name" value is used to set up messages. Various attributes like "required" is used for mandatory input field and "rangelength" is used for defining require input length.

messages: {

firstName: {

required: "Please enter your firstname",

rangelength: $.validator.format("Contains a maximum of{1}characters"),

},

lastName: {

required: "Please enter your lastname",

rangelength: $.validator.format("Contains a maximum of{1}characters"),

},

phoneNumber: {

required: "Phone number required",

},

address: {

required: "Delivery address required",

rangelength: $.validator.format("Contains a maximum of{1}characters"),

},

postcode: {

required: "Postcode required",

},

email: {

required: "Please enter your email",

email: "The email format is incorrect"

},

password: {

required: "Password cannot be empty",

rangelength: $.validator.format("Minimum Password Length:{0}, Maximum Password Length:{1}。")

}

}

});

For back button in header I looked up in homepage button and used same code to implement it in signup page.

**HTML**

<div data-role="page" id="signuppage">

<!------------------------Header---------------------------->

<div data-role="header" data-theme="d" data-position="fixed">

<a href="javascript:history.go(-1)" class="ui-btn ui-icon-carat-l ui-btn-icon-notext ui-corner-all">go back</a>

<h1>User Registration</h1>

</div>

<!------------------------Main content---------------------------->

<div data-role="content" id="content">

<div class="wrap">

<p class="fcolor"><b>Register Info</b></p>

<form id="signupform" method="get" action="">

<!--Fill info-->

<input type="email" name="email" placeholder="Email">

<input type="password" name="password" placeholder="Password">

<input type="text" name="firstName" placeholder="\*First Name">

<input type="text" name="lastName" placeholder="\*Last Name">

<p class="fcolor">Please select sta te: </p>

<select class="ostate" name="state" >

<option value="ACT">ACT</option>

<option value="VIC">VIC</option>

<option value="NSW">NSW</option>

<option value="QLD">QLD</option>

<option value="SA">SA</option>

<option value="TAS">TAS</option>

</select>

<input type="text" name="phoneNumber" placeholder="Phone number">

<input type="text" name="address" placeholder="Address details">

<input type="text" name="postcode" placeholder="Postcode">

<button id="signupbutton" type="reset" class="ui-btn ui-shadow ui-corner-all ui-btn-d">Submit</button>

<button id="clearOrderButton" type="reset" class="ui-btn ui-shadow ui-corner-all">Clear</button>

</form>

</div>

</div>

# **Updated Home page**

**HTML**

<div data-role="page" id="homePage">

<!------------------------Header---------------------------->

<div data-role="header" data-theme="d" data-position="fixed">

<a href="#loginPage" rel="external" class="ui-btn ui-icon-power ui-btn-icon-notext ui-corner-all"></a>

<h1>Gift Delivery</h1>

</div>

<!------------------------Main content---------------------------->

<div data-role="content">

<div class="wrap pb">

<ul class="ilist" id="menuList">

<ul> Creates an unordered list for menu items. Each <li> represents the menu. <a href="#selectPage">:- Defines a link to another page with the ID "selectPage".

<li>

<a href="#selectPage">

<img src="img/flower.png" />

<div>

<h3 class = "fcolor">Order a gift item</h3>

</div>

</a>

</li>

<li>

<a href="#PastOrdersPage">

<img src="img/order.png" />

<div>

<h3 class = "fcolor">See past orders</h3>

</div>

</a>

</li>

<li>

<a href="#UserPage">

<img src="img/login.png" />

<div>

<h3 class = "fcolor">User Profile</h3>

</div>

</a>

</li>

</ul>

</div>

</div>

<!--footer-->

<div data-role="footer" data-theme="d" data-position="fixed" class="footer" style="text-align:center;">

<p>Copyright © Gift Delivery 2024</p>

</div>

</div>

# **Order List Page**

**Javascript**

At first, event listener for the "pagebeforeshow" event on the element with the ID "PastOrdersPage" is initiated. When this event occurs, the provided fallback function is executed.

$(document).on("pagebeforeshow", "#PastOrdersPage", function() {

$('#pastorders').html("");

$('#pastorders').html("") empties the content of the element with the ID "pastorders".

(localStorage.allOrders != null) condition checks if there is any data stored in the local storage under the key "allOrders". *localStorage.getItem("allOrders")* retrieves the value stored under the key "allOrders".

if (localStorage.allOrders != null) {

var allOrders = JSON.parse(localStorage.getItem("allOrders"));

//console.log(allOrders)

(allOrders.length > 0) checks if the "allOrders" array has a length greater than zero. If condition is true then the code proceeds to next line.

if (allOrders.length > 0) {

$('#pastorders').append('<br><table><tbody>') selects the element with the id "pastorders" and it appends HTML content to element. It adds a line break <br>, followed by an opening <table> tag and an opening <tbody> tag which is the start of table structure to display the past orders.

$('#pastorders').append('<br><table><tbody>');

allOrders.forEach(function(orderInfo, index) {

$('#pastorders').append('<tr><td>Order No. : </td><td><span class=\"fcolor\">' + (index+1001) + '</span></td></tr>');

<tr><td>Order No. : </td><td><span class=\"fcolor\">' + (index+1001) + '</span></td></tr> is the HTML content that will be appended. It creates a table row <tr> with two table data cells <td>. First cell contains "Order No. : ", and the second cell contains a <span> element with the class "fcolor" and a dynamically generated order number (index+1001). (index+1001) is used to display order numbers starting from 1001, where index represents the index of the order in the array.

This is repeated for all of the other rows.

$('#pastorders').append('<tr><td>Customer: </td><td><span class=\"fcolor\">' + orderInfo.customerfName + ' ' + orderInfo.customerlName + '</span></td></tr>');

$('#pastorders').append('<tr><td>Item: </td><td><span class=\"fcolor\">' + orderInfo.item + '</span></td></tr>');

$('#pastorders').append('<tr><td>Price: </td><td><span class=\"fcolor\">' + orderInfo.price + '</span></td></tr>');

$('#pastorders').append('<tr><td>Recipient: </td><td><span class=\"fcolor\">' + orderInfo.firstName + ' ' + orderInfo.lastName + '</span></td></tr>');

$('#pastorders').append('<tr><td>Phone number: </td><td><span class=\"fcolor\">' + orderInfo.phoneNumber + '</span></td></tr>');

$('#pastorders').append('<tr><td>Address: </td><td><span class=\"fcolor\">' + orderInfo.address + ' ' + orderInfo.postcode + '</span></td></tr>');

$('#pastorders').append('<tr><td>Dispatch Date: </td><td><span class=\"fcolor\">' + orderInfo.date + '</span></td></tr>');

$('#pastorders').append('<tr><td colspan="2"></td></tr>');

$('#pastorders').append('<br>');

});

$('#pastorders').append('</tbody></table>');

} else {

$('#pastorders').append('<h3>There are no orders to display</h3>');

}

} else {

$('#pastorders').append('<h3>No orders found</h3>');

}

});

Overall, Above lines of code loops through each order in the "allOrders" array using a forEach loop and adds table rows <tr> containing details order number, customer name, item, price, recipient name, phone number, address, and dispatch date. Finally, it closes the table structure with </tbody> and </table> tags after processing all orders.

**HTML**

<div data-role="page" id="PastOrdersPage">

<div data-role="header" data-theme="d" data-position="fixed">

<a href="javascript:history.go(-1)" class="ui-btn ui-btn-icon-left ui-icon-carat-l ui-btn-icon-notext ui-corner-all">go back</a>

<h1>Order List</h1>

<a href="#homePage" class="ui-btn ui-btn-icon-left ui-icon-home ui-btn-icon-notext ui-corner-all">home</a>

</div>

<!------------------------Main content---------------------------->

<div data-role="content" id="content">

<div class="wrap">

<div id =pastorders></div>

<div style="margin-top:25px">

<a href="#homePage" rel="external" class="ui-btn ui-shadow ui-corner-all ui-btn-d">Done</a>

</div>

</div>

</div>

<div data-role="footer" data-theme="d" data-position="fixed" class="footer" style="text-align:center;">

<p>Copyright © Gift Delivery 2024</p>

</div>

</div>

# **User Page**

**Javascript**

$(document).on("pagebeforeshow", "#UserPage", function() {

Event listener for the "pagebeforeshow" event on the element with the id "UserPage" is declared. When this event occurs, the provided fallback function is executed.

$('#userdetail').html("");

This line empties the content of the element with the id "userdetail".

Now the (localStorage.userInfo != null) condition checks if there is any data stored in the local storage under the key "userInfo". *localStorage.getItem("userInfo")* retrieves the value stored under the key "userInfo". If there is data stored, the condition will result to true.

if (localStorage.userInfo != null) {

var userInfo = JSON.parse(localStorage.getItem("userInfo"));

//console.log(userInfo);

if (userInfo) {

$('#userdetail').append('<br><table><tbody>');

Now similar to order list page a table structure is created appending table body, table rows and cells.

$('#userdetail').append('<tr><td>Email: </td><td><span class=\"fcolor\">' + userInfo.email + '</span></td></tr>');

$('#userdetail').append('<tr><td>Name: </td><td><span class=\"fcolor\">' + userInfo.firstName + ' ' + userInfo.lastName + '</span></td></tr>');

$('#userdetail').append('<tr><td>Address: </td><td><span class=\"fcolor\">' + userInfo.address + ', ' + userInfo.state + ', ' + userInfo.postcode + '</span></td></tr>');

$('#userdetail').append('<tr><td>Phone number: </td><td><span class=\"fcolor\">' + userInfo.phoneNumber + '</span></td></tr>');

$('#userdetail').append('</tbody></table><br>');

} else {

$('#userdetail').append('<h3>User information not found</h3>');

}

} else {

$('#userdetail').append('<h3>No user information found</h3>');

}

});

Hence, above script checks if user information exists in “localStorage.allOrders”.If it does, it retrieves the user details and shows them using the append function which adds table rows <tr> containing details Email, Name, Address, Phone Number. Finally, it closes the table structure with </tbody> and </table> tags after processing user detail.

**HTML**

<!------------------------User page---------------------------------->

<div data-role="page" id="UserPage">

<div data-role="header" data-theme="d" data-position="fixed">

<a href="javascript:history.go(-1)" class="ui-btn ui-btn-icon-left ui-icon-carat-l ui-btn-icon-notext ui-corner-all">go back</a>

<h1>User Profile</h1>

<a href="#homePage" class="ui-btn ui-btn-icon-left ui-icon-home ui-btn-icon-notext ui-corner-all">home</a>

</div>

<!------------------------Main content---------------------------->

<div data-role="content" id="content">

<div class="wrap">

<div id =userdetail></div>

<div style="margin-top:25px">

<a href="#homePage" rel="external" class="ui-btn ui-shadow ui-corner-all ui-btn-d">Done</a>

</div>

</div>

</div>

<div data-role="footer" data-theme="d" data-position="fixed" class="footer" style="text-align:center;">

<p>Copyright © Gift Delivery 2024</p>

</div>

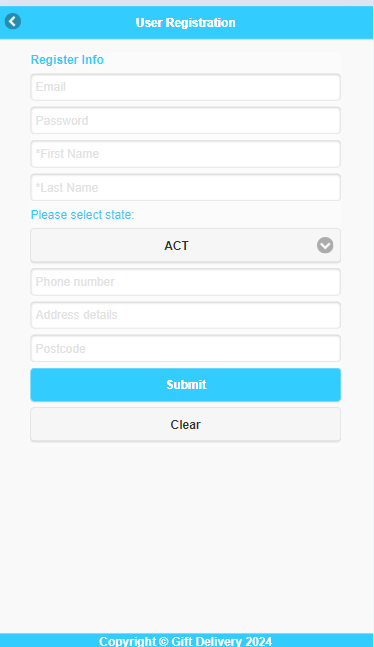
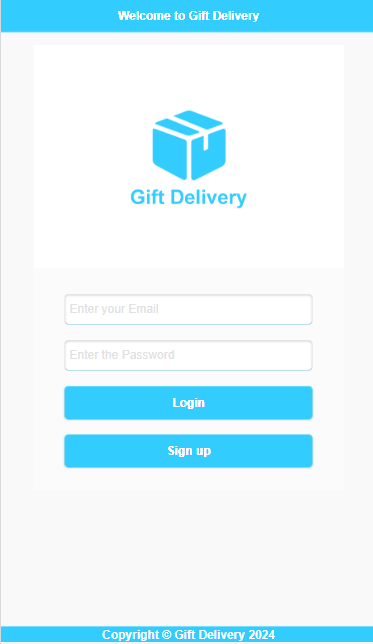
</div>

# **User Acceptance Testing**

1. Registration Testing:
   1. Test scenario: User registration.

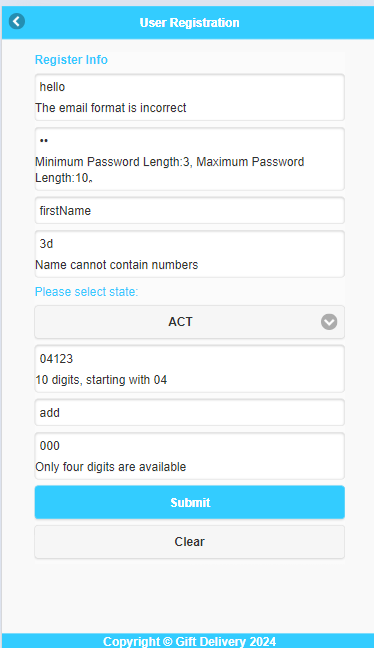
Navigate to the signup page:

When clicked on the sign up button on the login page, it directs us to the user registration page.



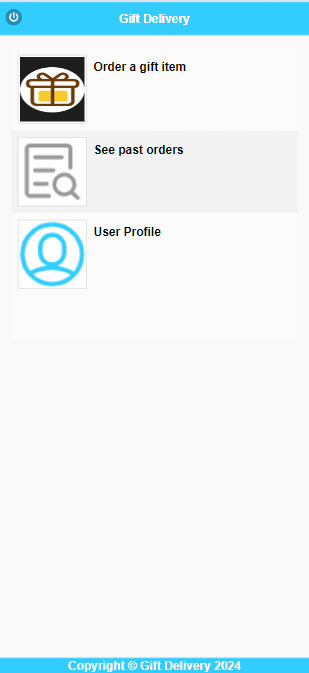
Validate Input Fields:

When all input fields are filled, an error message is shown if any input does not satisfy the required condition for submission.



If all fields are okay then it registers the user and switches the page to the homepage. Back icon directs us to the login page.

1. Homepage Functionality:
   1. Test scenario: Menus in homepage for users after login.
   2. Test steps:
      1. Login with valid credentials.
      2. After successful login, switch to the homepage with three menus showing, **Order a gift item, See past orders and User Profile.**
      3. After clicking on one of the menus it should switch towards the respective page.

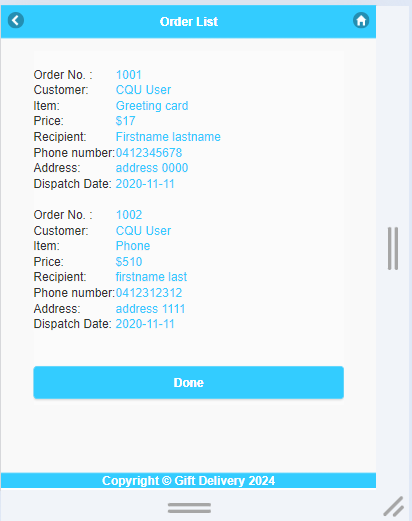


All three menus are successfully shown with image and text description and directs us to the respective page. Logout icon logs out user, removes the userinfo and directs us to the login page.

1. See Past Order Functionality:
   1. Test scenario: Past Order Check.

Navigate to the Order list page:

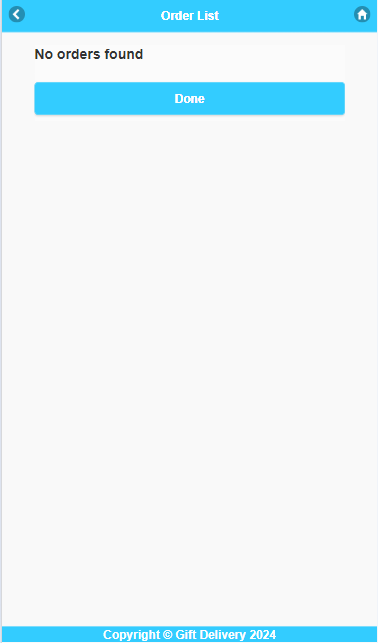
When clicked on the **See past orders** menu on the homepage, it directs us to the order list page.



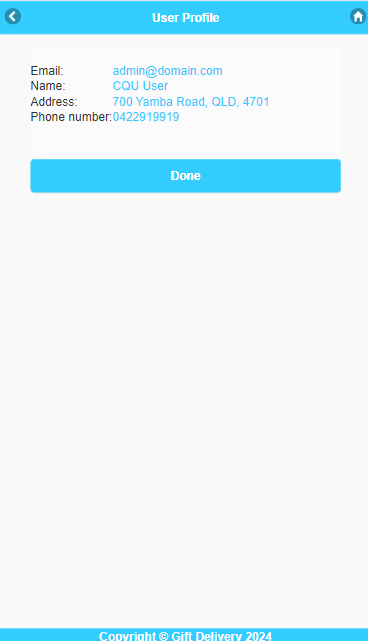
On clicking on the done button or home button it switches to the home page.

Case 2: No order Placed:

If the user has not ordered any items, it shows No order found.



1. User Profile Functionality:
   1. Test scenario: User Wants to see own profile.
   2. Test steps:
      1. Log in to the application using valid credentials.
      2. Navigate to the User Profile page.
      3. Check user details.
      4. Return back to the home page.



User detail is successfully shown which is retrieved from “localstorage.userInfo.

# **Successfully implemented features**

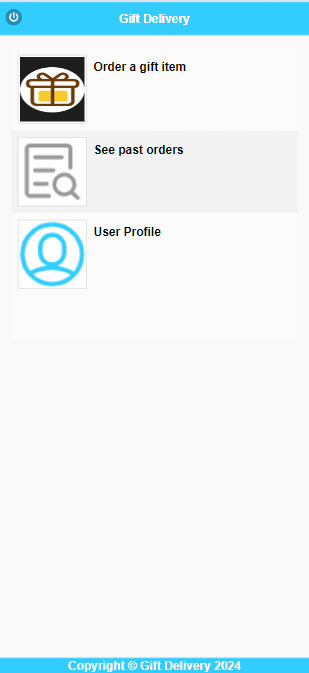
1. **Updated home page**

*Task was to implement two more menus named* ***See past orders*** *and* ***User profile*** *with associated image. Clicking on each link should switch to the respective page.*

**Approach towards updating home page:**

* First of all, I studied the existing HTML code of the menu page and used similar pattern in creating other two menus.
* Then I figured out JavaScript code after examining the existing code for **Order a gift item** for directing towards respective pages**.**
* Lastly, I finished the home page adding two other <li> list with respective title and associated image with <img> tag and added the event listener for newly added menus.

Two more links are added to the home page named, **See past orders,** and **User Profile** with an associated image**.** The updated homepage looks like this (Fig 1).



|  |
| --- |
| Fig 1. Gift delivery app: Updated home page |

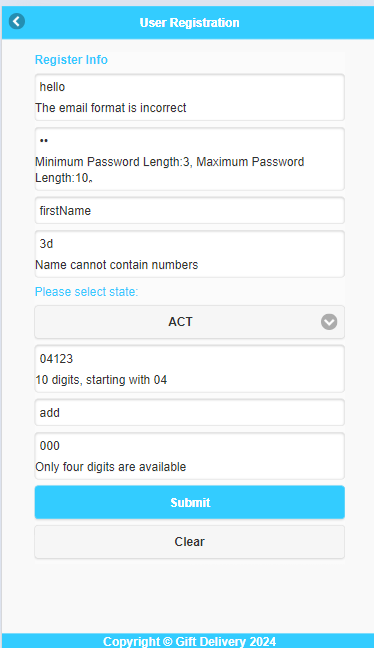
2. **Sign up page**

*Task was to create signup, when clicking on signup button on login page it should redirect to signup page where there should be 8 input fields, submit and clear button. Each fields should be validated.*

**Approach towards creating sign up page:**

* Firstly I noted the requirements of creating signup page and used the same design pattern in creating HTML page.
* For the validation and submission part, I took hint from the order confirmation page or order form validation.
* I checked on local storage "allusers" and "userInfo" for signing up and saved to that storage for successful registration and and switching to homepage with login after successful registration.
* Similarly to order form submission, I implemented validation, submission and clearance code for signup page which is explained above in javascript code implementation.

Sign Up page is created; clicking on signup button on login page directs us to **User Registration** page where user can register. Registered Users are added to "localStorage.allUsers".



|  |
| --- |
| Fig 2. Gift delivery app: User sign up page  Input fields are validated before submission. Clear Button clears all 8 input fields.  Back icon directs to the login page. |

.

**3.** **Order list page**

*Task was to create order list page, the page should show all the past order records retrieving from allorders localstorage.*

**Approach towards creating order list page:**

* First thing was to check the local storage for allorders array in Application/localstorage in browser.
* After the confirmation of local storage I retrieved the allOrders and use loop to iterate over all of the order in that array.
* I implemented that loop using event listener when the page is opened; it triggers the function including the loop code of retrieving array.

**Order List** Page is created. The page can be accessed through the **See past orders** link in the home page**.** The page looks like this (Fig 4).

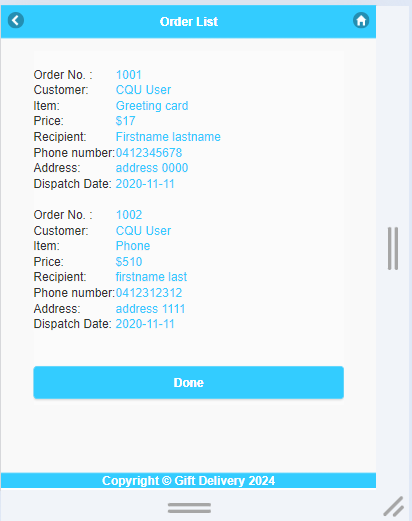


Fig 3. Gift delivery app: Order list page

|  |
| --- |
|  |

App retrieves information from "localStorage.allOrders" and populates the order list page with that information. Clicking on the done button takes us to the homepage.

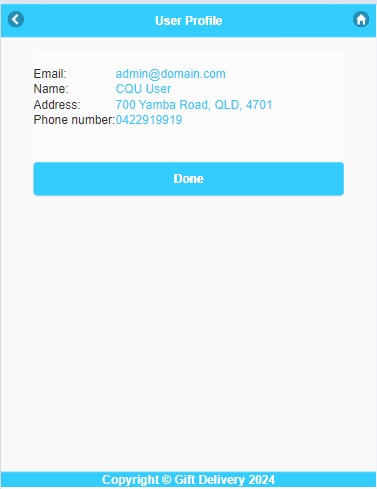
**4.** **User page**

**Approach towards creating User list page:**

* First thing was to check the local storage for userInfo in Application/localstorage in browser.
* After the confirmation of local storage I retrieved the userInfo, structured it into table and used append function to append the HTML table elements in DOM.
* I implemented this using event listener when the page is opened, it triggers the function including the loop code of retrieving array and rendering html content.

*Task was to create user page, the page should show all the user information stored in localstorage.userinfo.*

**User** **Profile** page can be accessed through the **User profile** link in the home page**.** The page looks like this (Fig 4).



|  |
| --- |
| Fig 4. Gift delivery app: User page |

Information about the current user is retrieved from the "localStorage.userInfo".

Clicking on the homepage takes us to the homepage.