**Chapter 8**

**jQuery Front-End Calling NodeJS Express Back-End**

**Building a Simple Front-End Client using jQuery**

Now that you have learnt how to create the back-end application, let’s create a simple front-end web client to consume the web service you have developed. We will create a log-in page that checks if the log-in credentials exist in the database by calling the backend api at <http://localhost:8081/user/login>. The JSON format will be used for all data exchange between the front-end and back-end. If the log-in credentials exist, we will retrieve the profile information (user name, email, role and profile picture) of the user and return them to the log-in page. A token will also be generated and passed back to the log-in page together with the profile information. This token will be required for updating user’s profile information.

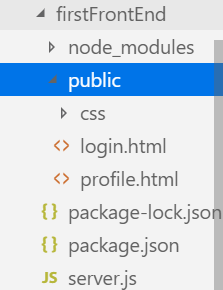
For this exercise, please make sure you add in a new column called pic (varchar(255)) in the database table for user.

Upon successful execution of the api, the user profile information and the token will be persisted in local storage before redirecting the user to the Profile page.

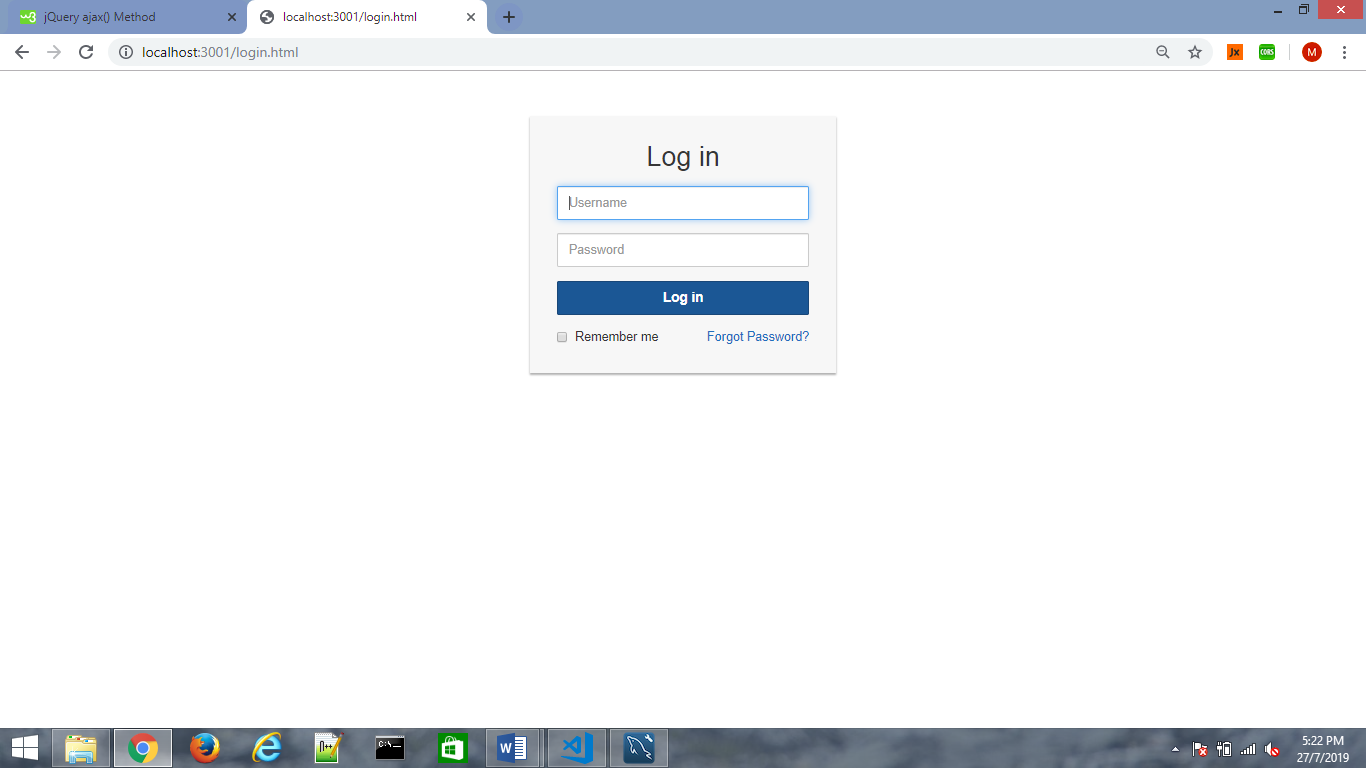
The profile page displays the user’s profile information such as user name, email, role and profile picture (note: only the file url of the image is stored in the database). The profile page also allows updating of profile information by executing the api at <http://localhost:8081/user/> with PUT method.

The log-out button executes javascript code which will clear all local storage variables.

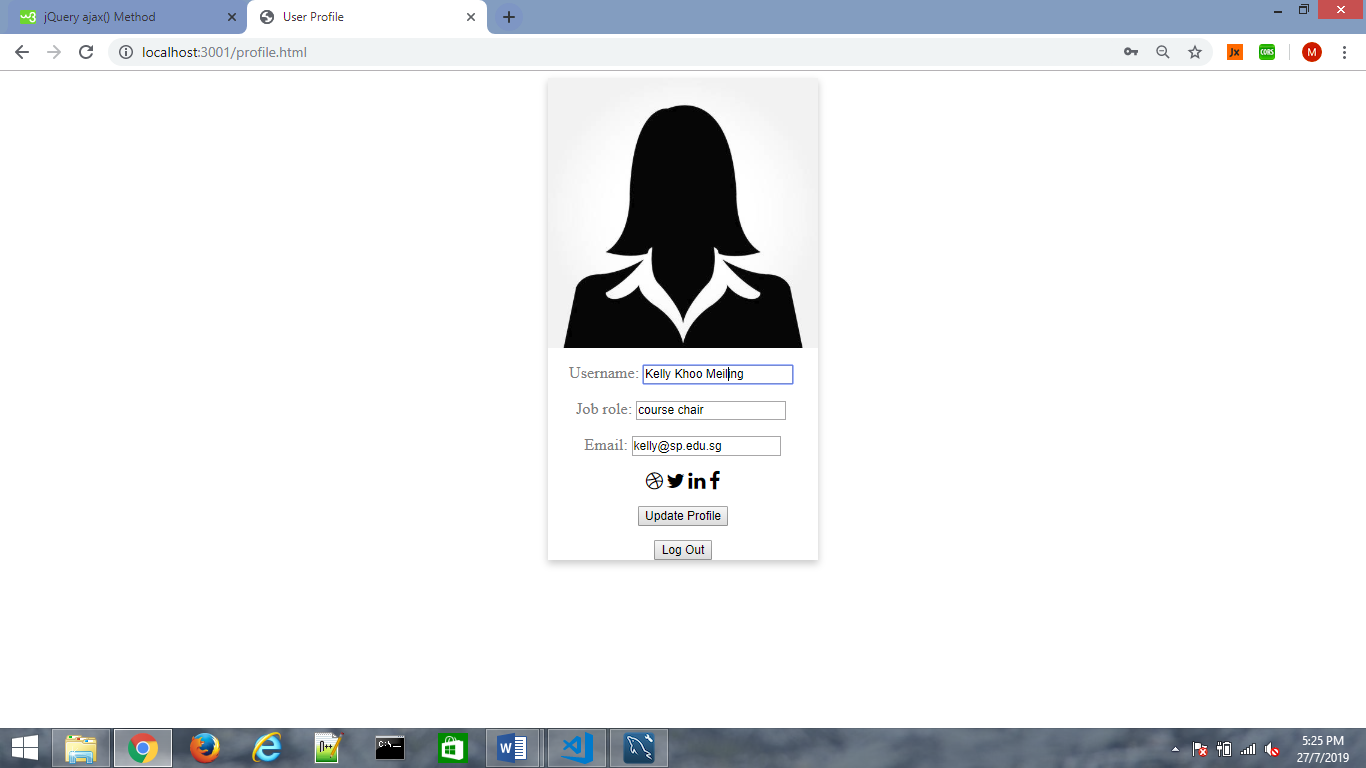
Let’s create a new project called firstFrontEnd. The following shows the directory structure of the project:



The following shows the Login and Profile pages:



Login page



Profile page

The above project has been created and is available for download from blackboard. The following is what you need to do to make the code work:

1. Update the database connection configuration in the databaseConfig.js file to connect to your own database.
2. Create the following 2 html pages:
   1. login.html
   2. profile.html
3. Ensure you have a profile image in the public folder of your web service and ensure the web service has uses serve-static node library to serve files in the public folder.
4. Run the web service in Visual Studio Code Terminal using node server.js and access your application on the browser at <http://localhost:3001/login.html>.

**jQuery**

jQuery is a JavaScript Library. It simplifies JavaScript programming and is easy to learn.

**jQuery ready() Method**

The ready event occurs when the DOM (document object method) has been loaded.

Because this event occurs after the document is ready, it is a good place to have all other jQuery events and functions.

The ready() method specifies what happens when a ready event occurs.

Use ready() to make a function available after the document is loaded:

|  |
| --- |
| $(document).ready(function(){   $("button").click(function(){     $("p").slideToggle();   }); }); |

**jQuery ajax() Method**

The ajax() method is used to perform an AJAX (asynchronous HTTP) request.

All jQuery AJAX methods use the ajax() method. This method is mostly used for requests where the other methods cannot be used.

Syntax

|  |
| --- |
| $.ajax(*{name:value, name:value, ... }*) |

The parameters specifies one or more name/value pairs for the AJAX request.

We will be using the following name/value pairs:

|  |  |
| --- | --- |
| **Name** | **Value/Description** |
| url | Specifies the URL to send the request to. Default is the current page. |
| type | Specifies the type of request. (GET or POST) |
| data | Specifies data to be sent to the server. |
| contentType | The content type used when sending data to the server. Default is: “application/x-www-form-urlencoded”. |
| dataType | The data type expected of the server response. |
| success(result,status,xhr) | A function to be run when the request succeeds. |
| error(xhr,status,error) | A function to run if the request fails. |

For a complete list of the possible names/values refer to:

<https://www.w3schools.com/jquery/ajax_ajax.asp>

**Submit A Form Without Page Refresh using jQuery**

In this exercise, we will learn how to submit a login form that sends the login credentials to the back-end without page refresh using jQuery.

Full source code for Login.html with explanation:

|  |
| --- |
| <!DOCTYPE html>  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  <!—customised CSS -->  <link href="css/login.css" rel="stylesheet" />  <!—compiled & minified CSS -->  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">  <!—jQuery library -->  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>  <!—compile JavaScript -->  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>  Beginning of jQuery code.  <script>  $(document).ready(function () {  Event handler for Login button.  $("#Login").click(function () {  var id=$('#email').val();  Retrieve login information from input tags.  var pwd=$('#pwd').val();  Format login data into a JSON object.  var data = "{\"email\":\""+id+"\", \"password\":\""+pwd+"\"}";    $.ajax({  url: 'http://localhost:8081/user/login' ,  Processing form submission with jQuery’s AJAX function to handle login.  type: 'POST',  data: data,  contentType: "application/json; charset=utf-8",  dataType: "json",  success: function (data, textStatus, xhr) {  if (data != null){  localStorage.setItem('token',data.token);  localStorage.setItem('userInfo',data.UserData);  window.location.assign("http://localhost:3001/profile.html");  Processing form submission with jQuery’s AJAX function to handle login.  } else {  console.log("Error");  }  },  error: function (xhr, textStatus, errorThrown) {  console.log('Error in Operation');  }  });  return false;  Prevents reloading of the page.  });  });  </script>  </head>  <body>  <div class="login-form">  <form action="">  <h2 class="text-center">Log in</h2>  <div class="form-group">  <input type="text" class="form-control" id="email" placeholder="Username" required="required">  </div>  <div class="form-group">  <input type="password" class="form-control" id="pwd" placeholder="Password" required="required">  </div>  <div class="form-group">  <button type="submit" class="btn btn-primary btn-block" id="Login">Log in</button>  </div>  <div class="clearfix">  <label class="pull-left checkbox-inline"><input type="checkbox"> Remember me</label>  <a href="#" class="pull-right">Forgot Password?</a>  </div>  </form>  </div>  </body>  </html> |

Basically what the jQuery’s AJAX function does is it processes the values from the string called data with the api called ‘login’ at url: 'http://localhost:8081/user/login', using the ‘POST’ method (type:“POST”). If the script processed successfully, the code then:

* persists the token and the user profile data in local storage,
* redirects the user to the Profile page at <http://localhost:3001/profile.html>

Finally the jQuery code returns false so the page does not reload.

**Display User Profile**

The next exercise displays the user’s profile information and picture.

Full source code of Profile.html with explanation:

|  |
| --- |
| <html>  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">  <link href="css/profile.css" rel="stylesheet" />  <Title>User Profile</Title>  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>  <script>  $(document).ready(function () {    $("#Update").click(function () {  jQuery code to handle Update button clicked.  var tmpUName = $('#username').val();  Retrieve values from input tags.  var tmpEmail = $('#email').val();  var tmpRole = $('#role').val();  var tmpPwd = $('#pwd').val();  Retrieve token from local storage. This token was created during successful login.  var tmpToken = localStorage.getItem('token');  var data = "{\"username\":\""+tmpUName+"\", \"email\":\""+tmpEmail+"\", \"role\":\""+tmpRole+"\", \"password\":\""+tmpPwd+"\"}";  Format user data into a JSON object.  $.ajax({  headers: {'authorization': 'Bearer '+tmpToken},  url: 'http://localhost:8081/user',  type: 'PUT',  data: data,  contentType: "application/json; charset=utf-8",  Processing form submission with jQuery’s AJAX function to handle updating of user profile information.  dataType: 'json',  success: function (data, textStatus, xhr) {  if (data != null && data.success){  $('#msg').html('Record updated successfully!');  } else {  console.log("Error");  }  },  error: function (xhr, textStatus, errorThrown) {  console.log('Error in Operation');  }  });  });  $("#Logout").click(function () {  Code to handle Logout button clicked.    window.localStorage.clear();  window.location.assign("http://localhost:3001/login.html");    });  </script>  </head>  <body>  <div class="card">  <img src="http://localhost:8081/image/pic.jpg" id="profilepic" alt="Profile" style="width:100%">  <p class="username"> <label>Username: </label>  <input type="text" id="username"> </br></p>  <p class="role"><label>Job role: </label>  <input type="text" id="role"> </br></p>  <p class="email"><label>Email: </label>  <input type="text" id="email"> </p>  <p><span id="msg"></span></p>  <a href="#"><i class="fa fa-dribbble"></i></a>  <a href="#"><i class="fa fa-twitter"></i></a>  <a href="#"><i class="fa fa-linkedin"></i></a>  <a href="#"><i class="fa fa-facebook"></i></a>  <p><input type="button" id="Update" value="Update Profile" /></p>  <p><input type="button" id="Logout" value="Log Out" /></p>  </div>  Retrieve user information from local storage. This information are passed over from login.html.  <script>  var userData = localStorage.getItem('userInfo');  Convert user information from string to JSON format.  var userJsonData = JSON.parse(userData);  var username = userJsonData[0].username;  var useremail = userJsonData[0].email;  Retrieve user information from JSON object and store in variables.  var userrole = userJsonData[0].role;  var userpic = userJsonData[0].pic;  document.getElementById("username").value = username;  document.getElementById("email").value = useremail;  Display user information on html document.  document.getElementById("role").value = userrole;  document.getElementById("profilepic").src = userpic;  </script>  </body>  </html> |

Full source code for Login.html.

|  |
| --- |
| <!DOCTYPE html>  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">  <link href="css/login.css" rel="stylesheet" />  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>  <script>  $(document).ready(function () {  $("#Login").click(function () {  var id=$('#email').val();  var pwd=$('#pwd').val();  var data = "{\"email\":\""+id+"\", \"password\":\""+pwd+"\"}";  console.log(data);  $.ajax({  url: 'http://localhost:8081/user/login' ,  type: 'POST',  data: data,  contentType: "application/json; charset=utf-8",  dataType: "json",  success: function (data, textStatus, xhr) {  if (data != null){  localStorage.setItem('token',data.token);  localStorage.setItem('userInfo',data.UserData);  window.location.assign("http://localhost:3001/profile.html");  } else {  console.log("Error");  }  },  error: function (xhr, textStatus, errorThrown) {  console.log('Error in Operation');  }  });  return false;  });  });  </script>  </head>  <body>  <div class="login-form">  <form action="">  <h2 class="text-center">Log in</h2>  <div class="form-group">  <input type="text" class="form-control" id="email" placeholder="Username" required="required">  </div>  <div class="form-group">  <input type="password" class="form-control" id="pwd" placeholder="Password" required="required">  </div>  <div class="form-group">  <button type="submit" class="btn btn-primary btn-block" id="Login">Log in</button>  </div>  <div class="clearfix">  <label class="pull-left checkbox-inline"><input type="checkbox"> Remember me</label>  <a href="#" class="pull-right">Forgot Password?</a>  </div>  </form>  </div>  </body>  </html> |

Full source code for Profile.html.

|  |
| --- |
| <html>  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">  <link href="css/profile.css" rel="stylesheet" />  <Title>User Profile</Title>  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>  <script>  $(document).ready(function () {    $("#Update").click(function () {  var tmpUName = $('#username').val();  var tmpEmail = $('#email').val();  var tmpRole = $('#role').val();  var tmpToken = localStorage.getItem('token');  var data = "{\"username\":\""+tmpUName+"\", \"email\":\""+tmpEmail+"\", \"role\":\""+tmpRole+"\", \"password\":\""+tmpPwd+"\"}";  $.ajax({  headers: {'authorization': 'Bearer '+tmpToken},  url: 'http://localhost:8081/user',  type: 'PUT',  data: data,  contentType: "application/json; charset=utf-8",  dataType: 'json',  success: function (data, textStatus, xhr) {  if (data != null && data.success){  $('#msg').html('Record updated successfully!');  } else {  console.log("Error");  }  },  error: function (xhr, textStatus, errorThrown) {  console.log('Error in Operation');  }  });  });  $("#Logout").click(function () {  window.localStorage.clear();  window.location.assign("http://localhost:3001/login.html");  });  });  </script>  </head>  <body>  <div class="card">  <img src="http://localhost:8081/image/pic.jpg" id="profilepic" alt="Profile" style="width:100%">  <p class="username"> <label>Username: </label>  <input type="text" id="username"> </br></p>  <p class="role"><label>Job role: </label>  <input type="text" id="role"> </br></p>  <p class="email"><label>Email: </label>  <input type="text" id="email"> </p>  <p><span id="msg"></span></p>  <a href="#"><i class="fa fa-dribbble"></i></a>  <a href="#"><i class="fa fa-twitter"></i></a>  <a href="#"><i class="fa fa-linkedin"></i></a>  <a href="#"><i class="fa fa-facebook"></i></a>  <p><input type="button" id="Update" value="Update Profile" /></p>  <p><input type="button" id="Logout" value="Log Out" /></p>  </div>  <script>  var userData = localStorage.getItem('userInfo');  var userJsonData = JSON.parse(userData);  var username = userJsonData[0].username;  var useremail = userJsonData[0].email;  var userrole = userJsonData[0].role;  var userpic = userJsonData[0].pic;  document.getElementById("username").value = username;  document.getElementById("email").value = useremail;  document.getElementById("role").value = userrole;  document.getElementById("profilepic").src = "http://localhost:8081/image/"+userpic;  </script>  </body>  </html> |

Additional Exercise helpful for assignment

Your respective lecturer will go through a new page, viewUsers.html which will demo how you can use jquery to retrieve multiple users and update the div on screen. The concept of retrieving a query string parameter with javascript is also taught through the clicking of a hyperlink(button) which will trigger an update of the dom with the new record information.

