**Team Member**

Pawan Araballi

Kranthi Kiran Chinnakotla

[Indra Kiran Reddy Nandipati](https://uncc.instructure.com/courses/43655/users/2732)

**Introduction:**

The assignment was to create a API for the user authentication functionality with all the error handling. So to implement this, we choose to implement with php and mysql, the client side to be as an iOS Application.

**MYSQL:**

The backend (MYSQL) has been deployed in Amazon Web Services. It contains a USER table which maintains the user details which contains email used by the user “email”, name of the user “name” and the password of the user “password”.

**PHP:**

The PHP has been deployed to a free hosting website 000webhost.com. It contains 2 different files

The request types used are GET and POST and the response would be in the json format.

* **signup.php:** It contains two parameters namely email, name and password which is used for the creation of the user with the parameters given by queries the amazon aws database. Different scenarios handled on signup action
  + Check for amazon aws database connectivity, if it is not reachable then there is a response called *database not reached*
  + On signup success we would get a response saying *registration successful* and with a status code 200. The json response is show below

{

"status": 200,

"status\_message": "registration sucesfull",

"id": 1

}

* + On using same name, then the response is the *user exists* with the status code as 200

{

"status": 200,

"status\_message": "user already registered",

"id": 0

}

* + On *invalid request* the status code would be 404.

{

"status": 404,

"status\_message": "Invalid request",

"id": null

}

* **login.php:** It contains the php code which queries to the database with the name and the password as the parameters. Different scenarios handled on the login action
  + On login success we would get a row of data and an alert would be displayed saying *login successful* with a status code as 200

{

"status": 200,

"status\_message": "login sucesfull",

"id": 1,

"data": {

"0": "pawan@gmail.com",

"1": "Pawan",

"2": "pawan",

"email": "pawan@gmail.com",

"name": "Pawan",

"password": "pawan"

}

}

* + On login failed we would get an alert displaying *invalid credentials* and the response status is 200

{

"status": 200,

"status\_message": "invalid credentials",

"id": 0,

"data": null

}

* + On *invalid request* the status code would be 404.

{

"status": 404,

"status\_message": "Invalid request",

"id": null

}

**iOS Mobile Application:**

This is a basic user authentication application which uses the API which has been created to serve the user authentication functionalities. Considering the requests and response paradigm we have created different views to handle the functionality.

* **SignUpViewController:** In this view controller we are hitting the API [http://www.advance\_mad.comxa.com/signup.php?signup=signup&email=\**(email)**&password=\**(password)**&name=\**(name)**](http://www.advance_mad.comxa.com/signup.php?signup=signup&email=\(email)&password=\(password)&name=\(name)) with email, password and name taken from the mobile application. It requests the API for signup. If the user is created successfully then we capture the response and show the relevant alert dialog with the message. All the basic error handling of null values on email, name and the password have been handled in Mobile Application itself.
* **LoginViewController:** In this view controller we are hitting the API <http://www.advance_mad.comxa.com/login.php?login=login&password=\(password!)&email=\(email!)> with email and password as the credentials taken from the mobile input. If the user exists, the appropriate response would be delivered in JSON as a response from the API. The appropriate response message would be displayed in the UIAlertDialog. All the basic error handling of null values on email, name and the password have been handled in Mobile Application itself.
* **FinalViewController:** It contains the user specific data with the email and the name of the logged in user.