Architecture-  


Intro- Application to order food from restaurants with elements like-  
>>Facebook login/logout  
>>Add address  
>>Place orders  
>>Track orders and view order history   
--------------------------------------------------------------------------------------------------------------------------  
  
Application

>>When installing it in new system, point to the package.json in command prompt and do npm init to load packages like “cors”, “request” etc  
In mongoDB database, create the necessary database and collection given in Sample Data.doc

>>First page is login.ejs which has js code from Facebook js SDK. Search “Facebook JavaScript SDK” to know about the SDK <https://developers.facebook.com/docs/javascript>   
Using it we access a unique ID and username. This unique ID acts as primary key in the database collection “users”

>>After proceeding a constant header will be available in the entire app whose code is written in header.ejs

>>Form submit that is action to some HTTP link and onclick features can take place in togetherness. This can be done by-  
  
<form id=”form”>  
</form>  
<script>  
…  
…  
document.getElementById(**'form1'**).**action** = **"/home"**;  
document.getElementById(**'form1'**).submit();

</script>

>> Value of name in the <input> forms a component in the link later which is posted (at the time when form action is taking place) and not the value of id. Thus mentioning name is mandatory. You can either access the <input> tag by document.getElementById('') or document.getElementsByName('')[0].

>> 4 states for status in mongoDB collection orders: pending, placed, rejected, delivered

>>Initially I wasn’t aware about HTTP POST request using the ‘request’ library thus I was using form action using submit button extensively, in few places it is better to post using  
request.post() to a route in api.js or index.js, like-

>>In jquery ajax post request  
$(document).on('click', '#reject', function(){...}  
is used over  
$('#reject').on('click', function(){...}  
as second one is not called after GET request is over.

>>You can’t simply access one domain from another, example localhost:3000 accessing localhost:5000, thus   
CORS- cross origin resource sharing is done. To implement that, “cors” library is used in app.js and all the other routes, as shown…

var cors=require('cors');  
router.use(cors());  
…  
router.get(‘…’, cors(), function(){…});

>>At different ports different HTML pages are rendered initially as if different applications are running in different ports. All of these are defined in app.js  
8886-rider  
8887/8/9-mcd/kfc/dominos

>>Access data from form action/submission  
usID=parseInt(request.query.usID);  
  
>>To create APIs in json from use code  
res.json(docs);

>>(Research more)  
As json wasn’t working in AJAX jquery post, I used dataType as “text” !  
The data received was of the form-  
{ '{"\_id":"577032e536526758050c7a5a"}': '' }  
In this data, {"\_id":"577032e536526758050c7a5a"} is acting as the first key, thus

var arr = Object.keys(req.body);

var id=JSON.parse(arr[0]).\_id;  
will give the id

>>Different between json parse and json stringify

<http://stackoverflow.com/questions/17785592/difference-between-json-stringify-and-json-parse>

>>mycollection.update($set:{ }) to change existing key:value pairs  
mycollection.update($push:{ }) to add extra key:value pairs

>>For a GET request from some place  
router.get(‘…’, cors(), function(){….});

>>For a POST request from some place  
router.post(‘…’, cors(), function(){….});  
NOTE: Observe, the commands in this function will be mostly related to requesting values which are posted, it looks like as if it is GET but as it is a consequence of a POST request thus, router.post();