Hello Sir,

As Mrs. Orunima provided a problem. Here's a solution from my end which may solve all the issues she mentioned.

Before jumping to any task let’s discuss how this API is structured.

**Database Name:** orunimasshop

**Username:** root

**Password:** “”

**$servername**=localhost

First, let us connect our frontend with our database including a **connection.php** file.

**Task-1: Store the product**

To store the products first, I’ve created a table named, “**products**” which has **id, name, price, quantity, weights, taste, details, make-date, status**.I will get the values from user input and populated into “**products**” table through **addproduct.php**

**Task-2: Keep the order**

For this task I’ve created a table named, “**orderlist**” which has **id, invoice, product name, product id, price, quantity, weight, taste, user id, userphone, user address, inside dhaka, delivery charge, make date, status.** While the user places an order from the cart system will get all the product id inputs. If the user selects location inside Dhaka then the system will populate 60tk as delivery charge, else the system will populate 100tk as delivery charge. If a user selects multiple items and checkout from the cart it will generate the same invoice id for that particular user. All of these data will be populated with the inclusion of **placeorder.php** file.

**Task-3: Deliver & Update stock**

Admin can see all the order lists with the help of **orderlist.php.**  I can update the inventory with the help of “**status**” if an admin inputs any invoice id as “shipped to delivery man” then the status will be updated from 1 (default status) to 2. If the delivery man successfully delivered the product then the status will be updated from 2 to 3. Which means the products are delivered. If the delivery man fails to deliver the product then the status will be changed to 4 from 2. All this process can be found in **deliveryupdate.php.**

**Task-4: Offer & Pre-order**

For this task I’ve created a table named, “**voucher**” which has **id, discount code, amount, is percent, item no, days, make date, status.** This discount code can be claimed in cart when all the requirements are matched in this case if the item no is greater than or equal to 5 only then this voucher code can be applicable or else the user can’t use this voucher code. If any user successfully uses voucher code then I will be able to get the voucher id. Then I add two columns in our **orderlist** table named, “is discount” and “discount price” if any user place an order without availing discount code then the “is discount” value will be 0 and “discount price” will be 0. If the user avail any discount code then “is discount” ‘s value will be 1 and “discount price” will be dependent on voucher id where I will be able to see if the “is percent” is true or false (0 or 1) in our case.

If “is percent” is true then the user will get a discount (certain percent) on all the products s/he included in the cart on the amount. So the price will be reduced by the given percentage and will be saved in the database in “price” and “discount price” columns. In the price column I will get “updated product price after discount” and in the “discount price” column I will get “the amount which I discounted”.

If “is percent” is false which means the admin selected a discount in the fixed amount. Let’s say, the voucher is worth 120tk. Total amount is 400taka and there are 4 items 100tk each items so the system will deduct 100 taka from the first item in the “price” column. The updated “price” will be 0 and the “discount price” will be 100 still 20 taka is still left. It will be deducted from the second order so for the second item the new price will be 100-20=80 taka.

To solve the voucher availability problem I used cronjob where at every 12am the system automatically updates voucher status. Status 0 means disable and 1 means active. It will update the status depending on days and make date column.

For pre order I created a table where all the pre ordered items will be stored. If any product isn’t available then the buyer will see that the product isn’t available but the system will let the user order that product as pre-order and all those pre ordered products will be stored there and admin has the access to update the stock information. Admin can notify the user that the product is available and ready to order again.

Task 5: Recommend other items

For this task similar items can be recommended. For test purposes if any item is out of stock then other items will be recommended according to taste and price. If the price difference is between 100 taka then the system will recommend this product. For demo purposes, when a user places an order then the system will automatically recommend another product as the product s/he chooses is out of stock. This function can be seen when **placeorder.php** is run as one of the items is out of stock thus it will recommend other products. .