The typical e-shop has the following functionality:

Authetication/registration page, cart, check-out system, search page.

Authetication/registration page:

Authetication:

User enters data, data is checked, if he entered correct data, he gets access.

Registration:

Data entered, checked for uniqueness, answer is retuned.

Cart:

Checks if there is something in cart. If so – show. If not, tell there is nothing inside.

Check-out system:

Takes data from cart, creates secure sesion, asks for card info, proceeds request. If payment passed, then sends request to delivery service.

3-layer system:

It takes input data from user(1st layer), then creates new mysql connection,adds salt to password(if salting is enables), rehashes/hashes password(2nd layer), looks for math in database(3rd layer).

In case of registration before creating new connection, the password is checked on presentation layer using javascrtipt(regular expressions) to be stong engough. Data proceeded to server, where it is being checked for uniquness of e-mail and login if they are, then server sends verification key to e-mail and then returnes success and promt to confirm e-mail if login and e-mail are correct to presentation layer. In case of Authorisation, if typed login/passwords mathed, then it will proceed to direcrtories of web-site that accessible only by authorized users. In case of failure, it will tell that something is wrong with login or password.

Cart:

When user(even unautorized) clicks on button “add to cart” near desired item, it puts id and amount to two possible locations:

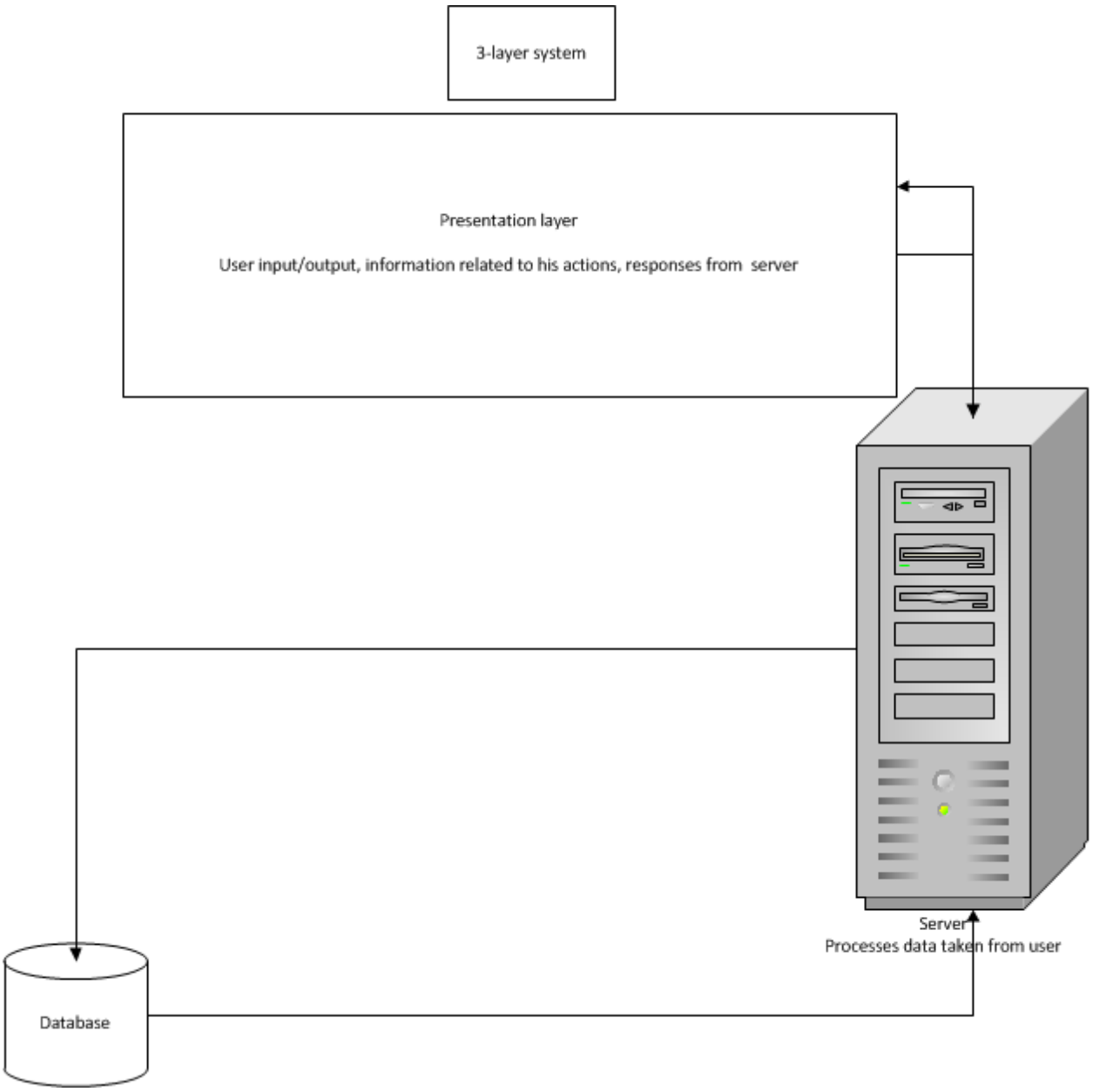
a) If user is not authorized(no session or cookies alive), then website uses user’s cookies.
b) If user is authorized(alive session), data is sent to server, which creates connetion to database, puts into the cart table id of user, id of product, and the amout desired.

Cart can be viewed any time.

When user goes to cart page, it generates query to database to get all the stuff user wants(using it’s id) to buy or looks for data in cookies. If nothing in db or cookies, then it will return a simple promt “nothing in cart”. If there is something in database or in cookies, it will show items, its prices and net price above with button proceed to checout.

Check-out system:

On the 1st layer user's data about his credit card and adress of delivery is taken. On the 2nd layer server tries to withdraw sum that should be payed by user, using db to get this sum(3rd layer). If succcess, then server returns to user "Success, wait for your order" and sends request to delivery service, if not, then it tells user what wend wrong.




2-layer system using fat-client system:

Authetication:

User has an app, app is connected to database. User inputs data in app, it is processed using client's resources, and then app calls for db.

Registration and Authentication works same, their behavior depends on data returned from db.

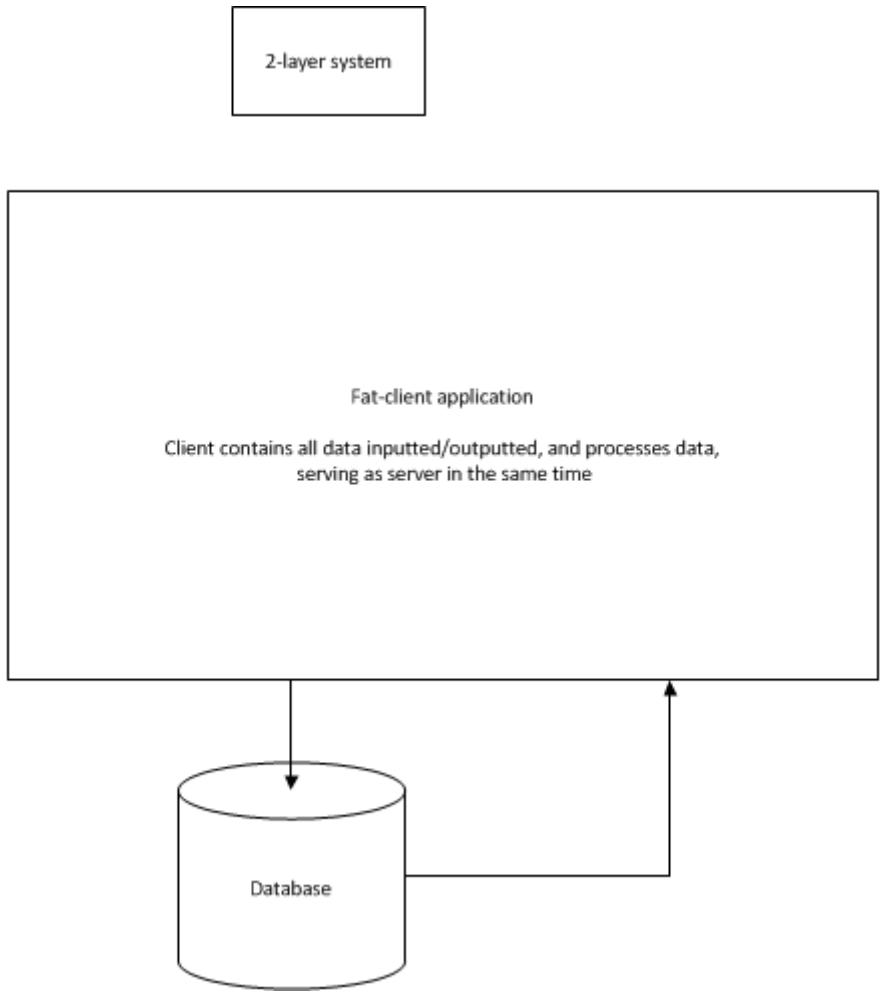
Cart:

Clicking on "add to cart" In app creates temprorary storage on device (if unautorized), which keeps id of goods in cart.

For authorized users there is a table in db which plays same role as above.

Check-out system:

App takes input from user, processes payment, and connects to payed_orders db to add info for delivery service.




1-layer system.

Everything is wrapped into 1 web page. This page has storage(connection to myslq db), js and php for data manipulating and html to display data on monitor.

Authentication/registration:

On monitor user sees fields that should be filled. The data input is being processed on the page(by js and php), then mysql queries are done, and page dynamically reloads, showing response, which was generated by page itself.

Cart:

Cart is filled when user(auth and non auth) clicks on “add to cart” button. When he clicks, js and php processing is done, saving id in some new array.

Check-out system:

Data about cards inputted, then processed by js and php right inside the page, which then sends a request to delivery service.