Online Musical Instruments

CS/SE 6301.501 Web Programming Languages

Team: Code Crackers

Team members: Praveen Sagar
Sainath Reddy
Bharath G.Marulasiddappa

Problem Description:

To design an e-commerce website. We designed and implemented E-commerce website where a user can chose and buy a musical instrument of his choice. User would be able view different type of musical instruments available on our website. He would also be able to do a checkout of the selected instruments using his login credentials.

Technologies used:

Client: HTML5, CSS, HTML 5 templates, AJAX, JQuery and JavaScript.
Server: XAMPP (Apache, MySQL, PHP)

Description of Open API's - Web services (RESTful)

a) Product availability:

Description: User enters the name of a product. On click of a button we make a web service call via Product Availability API, which interacts with the MySQL DB and gets the availability of the product. An API returns JSON data, which contains available units and status availability of the product.

b) Musical instrument manual download:

Description: User would be able to download a specific manual. Download API will returns a file if its available else returns not found.

c) Email subscription:

Description: Our website allows user to subscribe to our emails. Through Subscription API user's email id is saved in the server. API returns the success message when successfully stored in the DB.

Online Musical Instruments

Core Features

Authentication/Authorization (Login):
Our login system is a cookie based user authentication system. Whenever the user tries to use the cart on our website we require that the user is logged into his account. If he has not logged in it redirects him to the login page.
At the registration time we add a nonce by using the user name date the user registered and a nonce salt that is already defined. The combination of all is hashed using md5 to create the nonce. This nonce along with another salt are added to the user password and hashed using sha512. This hashed password is stored in the database as the user password. Whenever the user tries to log back in after completing the registration. We recomputed the nonce as we know the user registered date, user name, the nonce salt and generate the hashed password so that we can check if this is the same user.. After this we rehash the hashed password from the user, but this time using a different SALT and store it in a cookie. So whenever we get request we regenerate this double hash and check the cookies value.

High Performance:
This is achieved using distributed caching technique. Implemented using Memcache, which is distributed. Once an entry is placed in the cache all machines in the cluster can retrieve the same-cached item. Invalidating an entry in the cache invalidates it for everyone.
Installed Memcache. PHP requires a special DLL to access the Memcache. Downloaded php_memcache.dll and copied to the PHP folder. Now everything is ready to use Memcache. We then initialized Memcache by providing a default port, chunk size, allow_faliover.

Encryption:
Implemented using SSL which stands for secure sockets layer and is a form of security for sites that handle sensitive information such as customer names, phone numbers, addresses and credit card numbers. SSL is essential for any site that sells goods or services as it ensures that all information handled stays private and secure.
We created the self signed certificate and key using Open SSL and configured the httpd (in Xampp/apache/conf/extra/http-ssl.conf)

Online Musical Instruments

Compression: To make the data flow faster from the server to client compressing technique is used. Compression is useful because it helps reduce resource usage, such as data storage space or transmission capacity. We configured Apache/httpd.conf file to load these modules Mod_filter.so and Mod_deflate.so. We also specified the path and files that should be allowed for compression. For example, compress http files and not CSS files.

Asynchronous web service call: In the context of a web page, it's more likely that the service is synchronous, but that it is being accessed asynchronously through out our web design. The service is being called via AJAX. The call is made to the service, and the page then continues. When the response comes in, either the success or the failure functions are executed, asynchronously. We have consumed all open APIs using AJAX.

Challenges/Issues faced

a) Issue: We have the following flow. When we request a service through client it navigates from Client (written in JS and PHP) to Server (PHP to web service) and returns a response. This flow was working for every web service except the download web service.

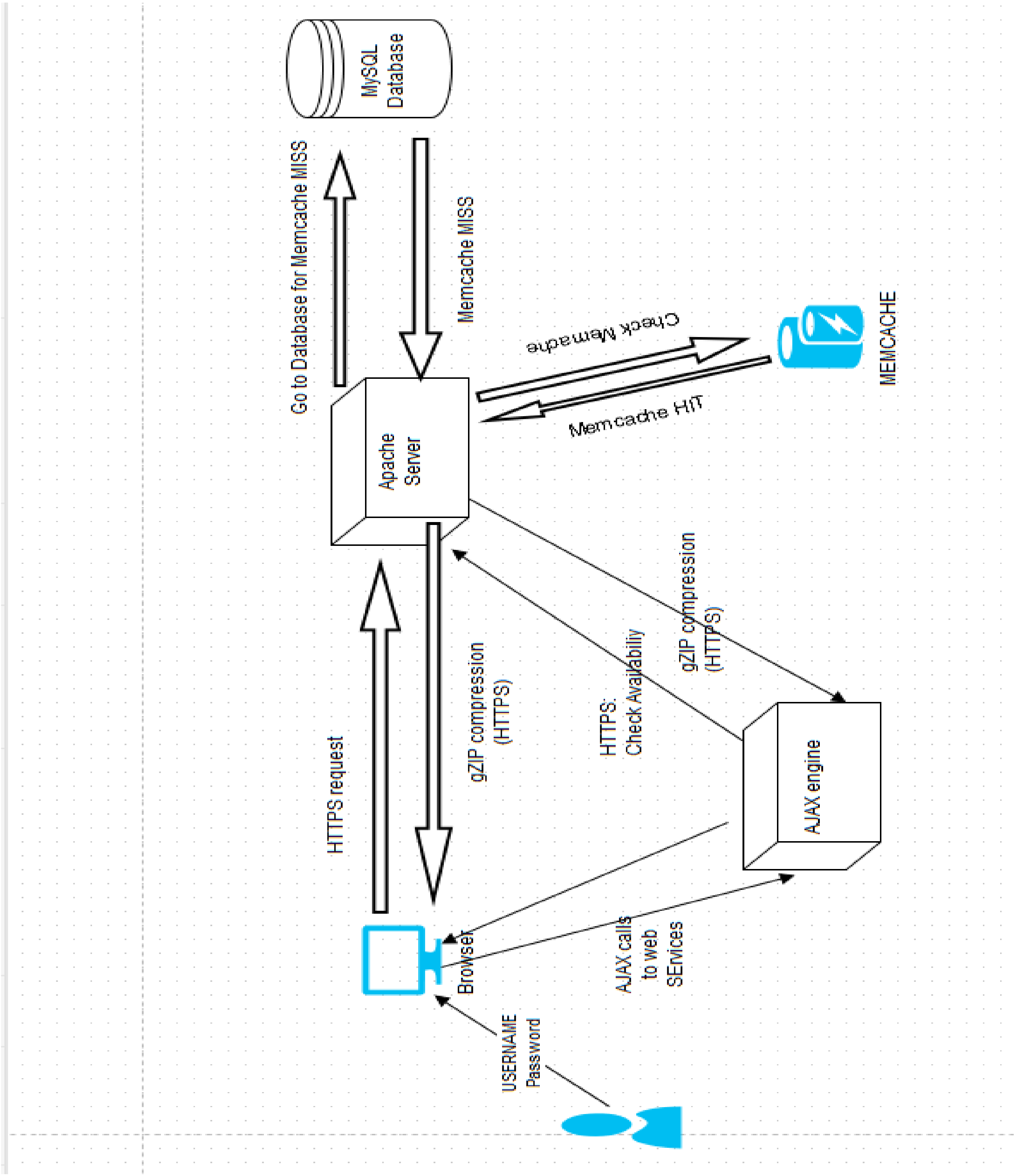
Solution: Invoked the download web service directly from client. Server will look for that file returns the file.

b) Issue: As we have implemented SSL for all web services with the self-signed certificate, the ***curl*** command in PHP won't accept self signed certificates and we did not get any data.

Solution: Fix was that we made **curl** to accept all certificates.

Online Musical Instruments

Architecture :



Online Musical Instruments

References:

To consume https web service
http://unitstep.net/blog/2009/05/05/using-curl-in-php-to-access-https-ssltls-protected-sites/

gZip compression
http://stackoverflow.com/questions/21945446/gzip-compression-not-working-on-xampp
http://stackoverflow.com/questions/6993320/how-to-enable-gzip-compression-in-xampp-server

Memcache integration to xampp
Please download memcached 1.4.4 Windows 32-bit from http://blog.couchbase.com/memcached-144-windows-32-bit-binary-now-available
Please read the readme file and install it following the steps accordingly.
Download php_memcache.dll from
http://windows.php.net/downloads/pecl/releases/memcache/3.0.8/ -> php_memcache-3.0.8-5.5-ts-vc11-x64.zip
http://www.leonardaustin.com/blog/technical/how-to-install-memcached-on-xampp-on-windows-7/

Login
https://www.youtube.com/watch?v=X6RNprqUPQc
http://php.net/manual/en/index.php