**7. Implementation**

We have implemented this project as a OData endpoint and a web interface through which the user can interact with the data according to the requirements. The code base that we have used for implementation includes libraries of OData available in php and object oriented php to represent and interpret data in the front end. We developed the OData Producer that exposes the Music Sphere data (songs, users, albums, artists etc.). The consumer should query the producer with the required parameters and the producer generates the results based on the query and the output is generated in the form of XML or JSON, whichever form needed.

The database was setup using MySQL and the endpoint is hosted as a service on a server where the queries were to be sent as a request to get the required data in response by the front end website which we made. The endpoint and the web site might not exist on the same server so the required URI for the endpoint has to be passed while querying from the web site.

**8. Discussion (showing examples of how the software is used)**

There are two components that have to be setup before using all the functionalities of the web site and the OData Endpoint. The first is the OData Endpoint for which we have to host the service for it on a webserver, may it be local to the same machine on which the web site is hosted or some other machine. The endpoint, once its up and running, should then be able to respond to queries coming from the web site or directly via the service URI. The request to the OData endpoint can be directly made through the service URI as shown below.

*http://<hostname>:<port>/MusicSphere.svc/$metadata*

The given query would return the metadata in XML format like it’s given below:

**((((@Sagar – Place a screenshot of metadata results over here))))**

After the endpoint is up and running the web site is to be hosted on a web server, which again might be on the same machine or some different machine, and the users can access the web site via its name or IP address.