This is a sample of a larger project that I created at Healthcare Management Systems.

It was to build a report framework that was easily extensible at incorporate new reports as they are being created. This makes use http protocol microservices so that the reports can be created from any application. The API receives the request and builds the report and responds with an excel file back to the requester. The purpose of using a microservice endpoint is to be able to decouple the application that calls the report API from the actual report builder, and to make the report framework easily extensible.

The requestor application passes all the arguments needed by the reporting framework in order to build the report. Example:

function isJSON($string){

return is\_string($string) && is\_object(json\_decode($string)) ? true : false;

}

function curlAPI($url, $data)

{

$curl = curl\_init();

try

{

curl\_setopt($curl, CURLINFO\_HEADER\_OUT, true);

curl\_setopt($curl, CURLOPT\_HTTPHEADER, array('Expect: '));

curl\_setopt($curl, CURLOPT\_URL, $url);

curl\_setopt($curl, CURLOPT\_RETURNTRANSFER, true);

curl\_setopt($curl, CURLOPT\_SSL\_VERIFYPEER, false);

curl\_setopt($curl, CURLOPT\_POST, true);

curl\_setopt($curl, CURLOPT\_POSTFIELDS, $data);

$res = curl\_exec($curl);

curl\_close($curl);

// decode

try

{

//return json\_decode($res);

return $res;

}

catch (Exception $e)

{

cliLog("JSON\_DECODE\_ERROR: " . $res);

}

}

catch (Exception $e)

{

cliLog("CURL\_ERROR: " . curl\_error($curl));

curl\_close($curl);

}

return false;

}

// Front end user would choose EmployeeReport (class) to run

$results = curlAPI(AUDITOS\_REPORT\_URL, array(

'api\_key' => 'bd991a6d-618c-4339-b2ca-101e67614e4e',

'report\_selected' => 'EmployeeStatusReport',

'project\_id' => '3',

'user\_id' => $\_SESSION['user']['name'],

'report\_grouper' => 'Anthem',

'report\_statuses' => 'complete',

'project' => false,

'multi' => false,

'grouper' => true,

'status' => true,

'type' => false,

'start\_date' => false,

'end\_date' => false,

)

);

$resultRows = json\_decode($results, true);

$helper = new \DC\Views\Helpers\DBTable();

echo "<table class=pad style=\"width:93%;margin:inherit;\"\>";

if(isset($resultRows) && $resultRows)

{

if (is\_array($resultRows) and count($resultRows) > 1)

{

$helper::headerCells($resultRows[0]);

foreach ($resultRows as $row) {

$helper::cells($row);

}

}

else

{

foreach ($resultRows as $k=>$v) {

echo '<tr><td>' . $k . '</td><td>' . $v . '</td></tr>';

}

}

}

else

{

echo '<tr><td>Response to API call is empty</td></tr>';

}

echo "</table>";

This calls the service report.controller in the framework and instantiates the report from a report Factory class, e.g.

try {

DocumentScan::checkPrimaryAuthentication($\_SERVER['REMOTE\_ADDR'], $\_POST['api\_key']);

$report = ReportFactory::CreateReportInstance($\_POST['report\_selected'],

$\_POST['project\_id']);

if(isset($report) && is\_object($report))

{

// $result = $report->getEmployees($\_POST['project\_id']);

# Generate report

$options['project\_id'] = $\_POST['project\_id'];

$options['report\_selected'] = $\_POST['report\_selected'];

$options['report\_grouper'] = $\_POST['report\_grouper'];

$options['report\_statuses'] = $\_POST['report\_statuses'];

if($\_POST['project']) $flags['project'] = true;

if($\_POST['multi']) $flags['multi'] = true;

if($\_POST['grouper']) $flags['grouper'] = true;

if($\_POST['status']) $flags['status'] = true;

if($\_POST['type']) $flags['type'] = true;

if($\_POST['start\_date']) $flags['start\_date'] = true;

if($\_POST['end\_date']) $flags['end\_date'] = true;

$result = $report->generate($flags, $options);

# get Temp Table Name

// $this->result=array('TempTableName' => $report->getTempTableName());

$this->result = is\_object($result) ? $result : (object) $result;

}

else

{

$this->result = (object)$report;

}

// json only response

header('Content-type: application/json');

echo json\_encode($this->result);

}

catch (Exception $exception) {

// Rollback here.

header('Content-type: application/json');

$result = '\*\*\* EXCEPTION - Couldn NOT create report = $\_POST[\'ReportSelected\']; exception = ' .$exception->getMessage();

echo json\_encode(array('exception'=>$result));

}

Factory class instantiates the report name dynamically that is passed in:

class ReportFactory

{

public static function CreateReportInstance($reportClass, $params)

{

try {

return new $reportClass($params, $reportClass);

} catch (Exception $exception) {

$result = '\*\*\* EXCEPTION - Could NOT create report =

$\_POST[\'ReportSelected\']; exception = ' . $exception->getMessage();

return array('exception' => $result);

}

}

}