**Assignment 1 Report**

**i. how to run your application**

**ii. design of the data model**

*The use of JavaScript Object Notation JSON.*

*One to many relationships? Many to many relationships?*

*The relationship or the structure of the database/csv files.*

*The framework used to store the data*

Under WorkloadProject, The Program.cs file returns the data from the csv files using the *ListOfWorkload.GetWorkloads()* function. These data are placed in Models.

The models are composed of Batch, ClienRFW, LisfOfWorkload, ServerRFD, and Workload.

**Batch:** BatchId

**ClientRFW:** RFW Id, BenchmarkType (where it initialises DVDTesting, DVDTraining, NDBenchTesting , and NDBenchTraining), WorkloadMetric (where it initialises CPU, NetworkIn, NetworkOut, Memory), BatchUnit, BatchId, BatchSize.

**ListOfWorkload:** That gets the list of Dell DVD testing and training, and the list of NDBench testing and training.

**ServerRFD:** RFDID (Response For Data), LastBatchID, and list of Batchs.

**Workload:** PUUtilization\_Average, NetworkIn\_Average, NetworkOut\_Average, MemoryUtilization\_Average, and FinalTarget.

**iii. methods used to data serialization/de-serialization**

The serialization is the process that converts DotNet objects into bytes streams in order to store data from the csv files. Deserialization is the opposite process that converts bytes streams to objects. These data serialization and deserialization methods have been done in two ways: in a text based Json and in binary (gRCP Protole)

**iv. how ii) and iii) are applied in the data communication of your Application**

The serverController.cs file, under WorkloadProject/Controllers, is responsible for managing the data calls and defining the logics of mapping rules of the application. It uses the build-in framework automatically to get the “Get” from the APIs. A server object (serverRFD) is created to access the batches along with its parameters such as BatchID and RequestedSample.

The workload list (*List<Models.Workload> workloadList*) object stores the data fetched from the scv files like DVD-testing.

The method *GetAllWorkloadColumnValues* uses these workload lists and takes all the column values list (*List<double> allColumnValuesList*). It also uses the workloadMetric that comes from the client, verifies the start input, chosen by the user, and the endIndex, that depends on the batch size, to put them in the batches as double.

The method *GetAllBatches()* calls the list of the column values by verifying the batch units to organize all the batches and returns a *List<Batch> allBatchesList.*

Then the program starts a loop to go through all the batches and add them to the serverRFD object before returning it. This loop depends on the user input parameters. For instance, if the user wants to start at the BatchID: 10, the start index will get the value 10, and ends at the batch size.

In the GrpcServer, the Workload service under Services folder, the structure and the logic are quite similar to the workload project except it has a Service instead of a Controller and the data is transmitted by Proto instead of Json.

The gRPC is an open source framework made by Google. It allows defining and Requests and Responses for Remote Procedure Calls by using Protocols Buffers (Proto3). Both, the client and the server use the work.proto files for the request and response workloads.

The client program.cs asks the user to enter RFWID (request for workload id), the benchmark type, the workload metric, the batch unit, the batch id, and the batch size. These inputs form the workload request object to be sent to the server.

**v. discuss the libraries or software packages you choose to deal**

**with data serializations (e.g. pros or cons given your**

**experience)**

Framework ASP.NET core with C#. It’s an open source used to create server client applications. It has many benefits compared to .NET core.

gRBC framework.

**vi. Screenshots of running your application with SUCCESSFUL results.**