DELIVERABLE 2:

FUNCTIONAL DEPENDENCIES:

CUSTOMER : C_ID -> { CName, Start_date }
TRAINER : T_ID -> { TName, Salary }
MANAGER : M_ID -> { MName }
EQUIPMENTS : E_ID -> { EName, Price, Qty, Work_Cond, Main_date }
PACKAGE : P_ID -> { Cost, Duration }

NORMALISATION:

In order to show that our relational design follows the third normal form, we need to make sure that it fulfills the following conditions of being in the third normal form:

- It should be in the second normal form.
- It doesn’t have transitive dependencies.

Firstly, transitive dependencies occur only in those relations which have 3 or more attributes. Most of our relations have 3 or more than 3 attributes, but it does not necessarily mean that our relational design is not 3rd normalized.
This is because, the transitive dependency means that if A -> B and B !->A, and further B -> C, then A -> C should hold. But, according to our relational design, this transitive property is not being followed. Because, for example, we take our CUSTOMER relation which has 3 attributes: C_ID, CName, Package. Here, C_ID can identify both CName and Package, but CName alone cannot identify the Package, as there might be 2 people with the same name. Therefore, this rules out the possibility of C_ID -> Package through CName, because even if CName did not exist, we could have find out Packages using C_ID only.
Hence our second condition is fulfilled.

Coming to the first condition, in order for our relational design to be in second form, it should fulfill the following two conditions:

- It should be in first normal form
- It should not have any partial dependency

To prove the second point, let us take our EQUIPMENTS relation. When we use EName to search for a price, we would always get a unique answer because one equipment would not have taken more than 1 price. Now,if we use Qty along with it, it will give us a different answer, which would be sum of all prices, and hence, in this case, if AB -> C, then A implies a different value. And this is why our relational design does not have partial dependency.

Now, we only need to prove that it is in our first normal form, which requires that the following requirements be fulfilled:

- It should have single valued attributes.

All our attributes in all the relations are single valued. So this one is followed.

- Values stored in a column should be of the same domain.

All our relations contain columns whose entries belong to the same domain, for example, ID column will only contain numbers, and no strings.

- All the columns in the table should have unique names.

None of the columns of any relation have the same name. Therefore, this requirement is also fulfilled.

- The order in which the data is stored does not matter.

This requirement is also fulfilled because there is not restriction on the order of data in our relational design.

These requirements are fulfilled and hence our relational design is in first normal form. This fulfills the first requirement of our design to be in second normal form.

And, since our design fulfills the requirement of being in the second normal form, this implies that our first requirement of our design to be in the third normal form is fulfilled.
Since all our requirements of our design to be in third normal form are fulfilled, we can say that our relational design is in Third Normal Form.

Team Members:
Richa Goswami (2016077)
Suyash Singh (2016105)