Group Report

Jack Williams – jw15326

Liam Wardle – lw15234

Durgesh Pareva - dp15298

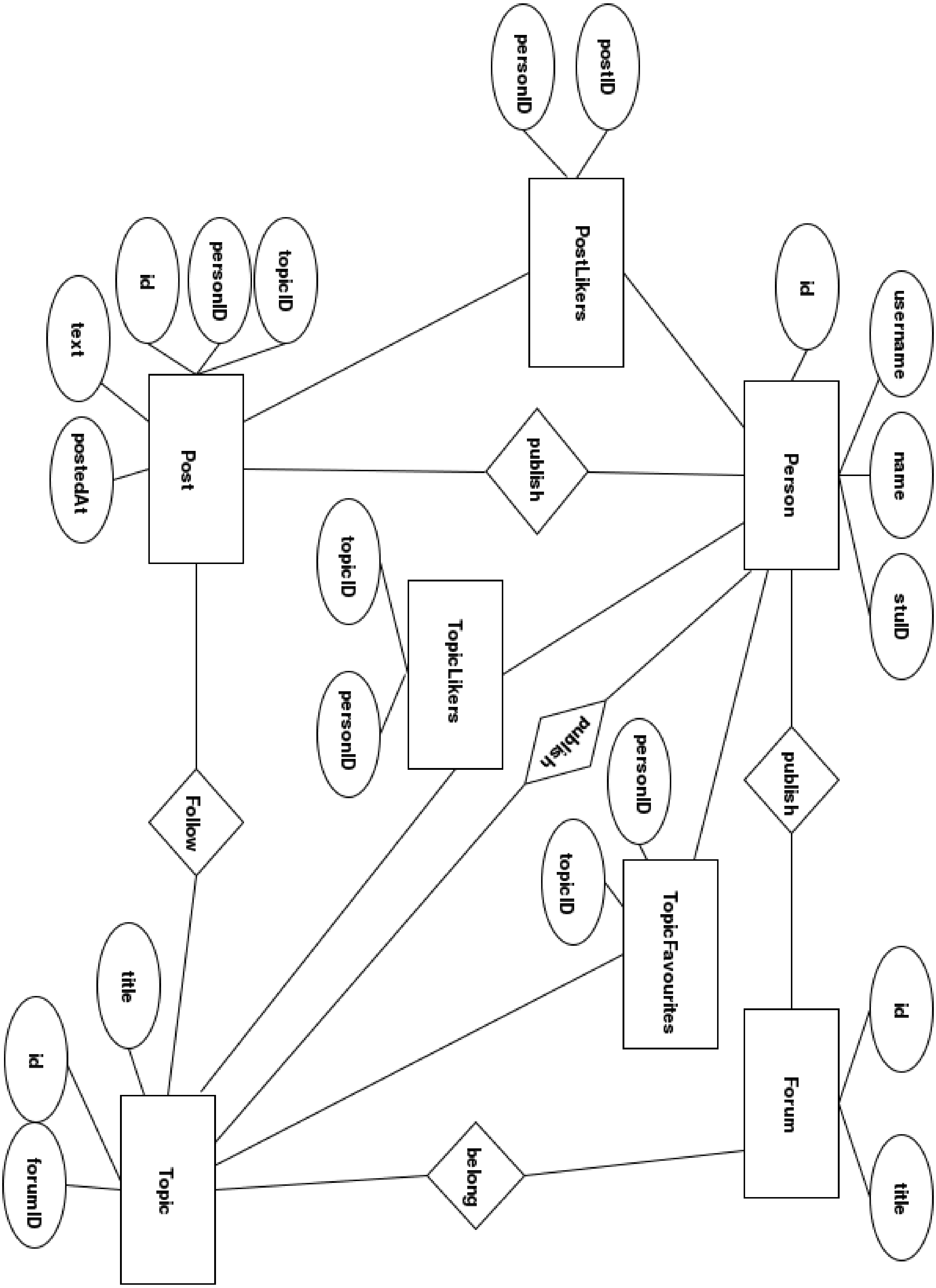
Methods	**Group Member**
getUsers	Durgesh
getPersonView	Durgesh
getSimpleForums	Durgesh
countPostsInTopic	Liam
getLikers	Jack
getSimpleTopic	Liam
getLatestPost	Liam
getForums	Jack
createForum	Durgesh
createPost	Liam
addNewPerson	Durgesh
getForum	Jack
getTopic	
likeTopic	Jack
favouriteTopic	Jack
createTopic	Liam

It is important to note that although this method table indicates each method was purely implemented by a sole individual, many of these methods required modifications and corrections by other members in order to work correctly, for example following a schema change. However, at least 1 method in each level was written individually.

Schema

The schema was extensively modified throughout, with the main additions being provided by Jack and Liam.

ER diagram



PostLikers
postID
personID
Post
id
personID
topicID
text
postedAt
publish
Person
id
username
name
stuID
TopicLikers
topicID
personID
publish
publish
TopicFavourites
personID
topicID
Follow
Topic
title
id
forumID
belong
Forum
id
title

Design choices for schema

To begin with, we took the necessary steps in order to set up four tables relating to Forum, Topic, Person and Post. We were able to decide on all of the relevant attributes per table by studying the constructors within each relevant class. In order to normalise the tables, we aimed for 3NF normal form. To this end, we minimised the number of attributes in each table, for example the Forum would simply contain and 'id' and 'title', and Topic would contain 'id', 'forumId' and 'title'. We took this decision in an attempt to prevent the storage of the same information across multiple tables, thereby increasing normalisation.

As we studied the classes, for example TopicSummaryView and PersonView, we became aware that new tables were required to be added to the schema, including tables to store the number of likes on a topic or post, with references to a personID and post/topicID. This was also the same for TopicFavourites table, where a personID and topicID provides us with a means of storing a favourite topic for a specific person.

We also considered adding an additional attribute to the Topic table, where we store the author of a topic. However, we decided upon storing the first poster within a topic as the author.

Another decision we took to minimise the number of unnecessary attributes was using Java to handle the number of posts or post number in a topic. We felt that this was a viable approach to achieving these calculations.

We initially had difficulties on how to store the time within the Post table, as at one point we were attempting to store it as a string. This proved ineffective, and so after studying the relevant files, we were able to effectively store as an integer and then carry out the necessary calculations to provide the correct time to be displayed within the forum.

For the generation of id numbers, we found the method of the provision of 'nulls' to automatically generate the correct id for each table was an effective means of allowing the database to carry out this task.