Class Diagram changes Justifications

Business Layer

Redesigned BL according to the feedback from MS1:

-User:

now doesn't contain board actions, it has useful constants fields to avoid magic numbers, decided to move the login/logout methods to UserController responsibility(explanation in the next point) and now User has only helper methods such as validating password rules and making sure the password is correct.

-UserController:

contains two more fields- a user dal controller to update the Users DB table and connectedUser which contains the email of the connected user at the moment. We wanted to avoid dependency between UserController and BoardController so when the service is initialized this field is set to be a shared resource of UC and BC so BC can know who is connected right now without aggregating UC. Thus we decided to move login/logout responsibility to UC and moved the helper methods towards User.

-Column:

A class that has been added to increase modularity(been told to do it in MS1 feedback). Now Board contains 3 columns instead of 3 Dictionaries of tasks.

Hold a column dto to modify its relevant record in DB when is needed.

-Board:

Added useful constants field to avoid Magic Numbers. Generalized methods(i.e. AdvanceTask replaces BacklogToInProgress and InProgressToDone), Column missions are now moved into Column class which manages tasks directly. Contains column dal controller field to insert its columns while created and remove them when is deleted.

-BoardController:

ConnectedUser field has been added(same reason as in UC). BoardMemberDalController and BoardDalController field were added to update DB when is needed. IdCounter added to give a unique address for each board. Boards field has been changed into Dictionary<Board,IList<string>>, it holds pairs of a board and emails of its members to have to "Join" feature.

Added JoinBoard method and moved column/task/any "lower level" missions to Column/Board classes.

-Task:

Added useful constants fields to avoid magic numbers, holds task dto to modify the relevant DB record when is needed.

*Added LoadData() and Delete() methods in the classes to enable loading data from DB and clearing the DB when it's wanted.

*Renamed some of the methods to fit with c# conventions.

Data Access Layer

-DTO:

Removed id fields and added instead primary keys list and primary values list- because some objects are uniquely identified by a combination of primary keys and not a single one(for example- a column is identified by the pair of board id and the column name).

-UserDTO: added user id field(more comfortable for the sql queries)

-TaskDTO: added column ordinal and board id fields because a task id is not unique in our system but the combination (boardId,taskId) actually is and because we need to know which column does a task belong to.

-ColumnDTO: new class which is needed due to modularity.

-BoardMemberDTO: new class which represents a user who participates(created/joined) in a board, needed due to the join feature and the need to know which users are members of each board.

-BoardDTO: added user id field(more comfortable for the sql queries)

Removed unnecessary fields which are being taken care by column dto and task dto(and their corresponding getters/setters)

-DalController: changed update methods to be suitable with the new DTO changes, added helper method makeWhereCond to deal with multiple keys where condition in sql queries, added deleteAll method which clears the relevant table

-TaskDalController- added methods for removing and getting tasks related to a specific board, added StringToBool method to convert DB string an appropriate Boolean for the IsDone field

-ColumnDalController- has been added to access Columns DB table

-BoardMemberDalController- has been added to access BoardMembers DB table