Presentation Layer Service Layer UserService **BoardService** + changePassword(username: string , oldP: string, newP: string): json + createBoard(Name: string, userEmail: string) + createUser(username: string, password: string): json + addTask(title: String, Description: String, DueDate: String): json + login(username: string, password: string): json + nextState(email: string, boardName: string, State: int, TaskID: int): json + getBoard(BoardName: String): json + deleteBoard(Name: string, userEmail: string) + getInProgress(): json + SetColumnLimit(ColName: string, BoardName: string, ColState: int) + logout(userEmail: string):json SetColumnLimit(ColName: string, BoardName: string, ColState: int, NewLimit: int) TaskService + createTask(Title: string, Description: string, DueDate: string): json + EditTitle(newTitle: string, user: User, board: string): json + EditDueDate(newDate: string, user: User, board: string): json + EditDescription(newBody: string, user: User, board: string): json + GetState(string: username, string: BoardName): json **Business Layer** Board UserController - Backlog(Collection) - users: Dictionary<string, User> - InProgress(Collection) - loggedIn: Collection - Done(Collection) - TotalNumOfUsers: int + Name(String) - userDTOMapper: UserDTOMapper - maxTasks: int = -1 - numTasks: int ListofJoiners(List<string>) - Owner(string) + DeleteUser(email: string): void + UserExists(email: string): boolean -BoardID: int + GetUser(email: string): User + addTask(title: String, Description: String, DueDate: String): void + validatePassword(user: User, password: String): boolean + changeState(TaskD: int, Assignee: string): void + login(email: String, password: String): void + AddToJoinList():void + logout(user: User): void + GetColList(): void + GetInProgress(): void User + GetNameOrdinal(): void + USERNAME: string + GetInProgressByAssignee(string: Assignee) password: string + LeaveTask(string: email): void - userDTO: UserDTO + getters + ChangePassword(oldP: string, newP: string): void + setters + ValidatePassword(password: string): bool Task

Changes Made after submitting Milestone 1:

HasAnyBoard- Check if a BoardController is empty - for exception

HasThisBoard- Checks if a BoardController has a board - for

GetColList - in order to get all tasks in a given column in a board

GetNameOrdinal - Returns the name of a column given its id in

A User object no longer holds a collection of their boards and

The class user is therefore connected only to the UserConroller,

The checks in order to get a user's board and tasks are preformed

UserController.IsLoggedIn(). This allows a better (lower) duality

Setting the max number of tasks in now done from the Board

Changes Made before starting Milestone 2

New methods for all new functional requirements. Data Access

it is an attribute of a board rather than a task. It can be said that

GetAllInProgress - See design changes for explanation

IsLegalPassword- in order to validate the password

IsValidEmail- To validate the email address at registration

Created:

IsHebrew - to validate email

exception handling purposes

Design changes:

which limits our duality.

between the classes.

Created:

Layer DTO's and Mappers.

Design changes:

through calling an instance of

(rather than Task). This is done because

initially we made a design mistake.

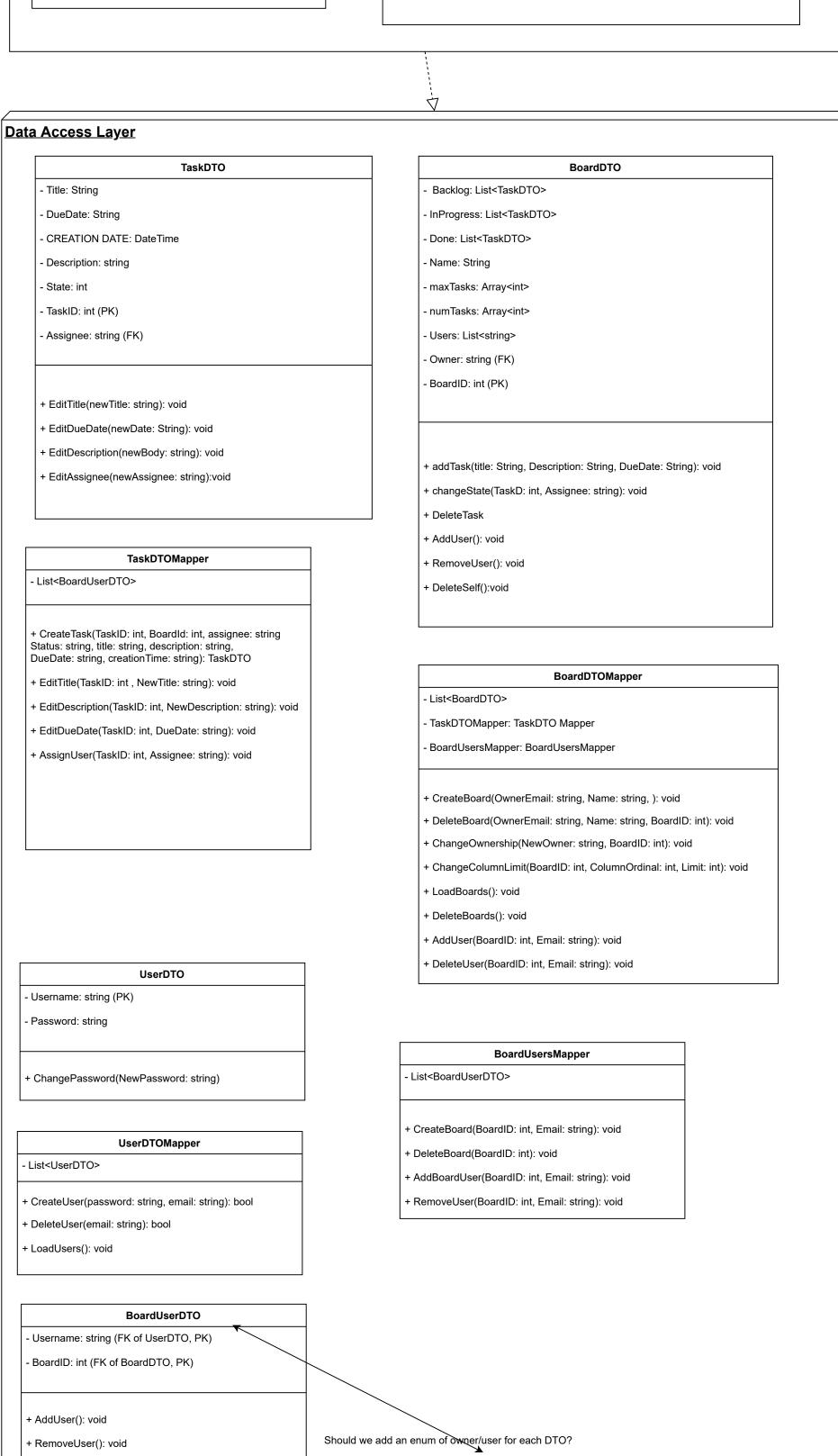
AddTask - same as max tasks.

Added getters and setters in Board

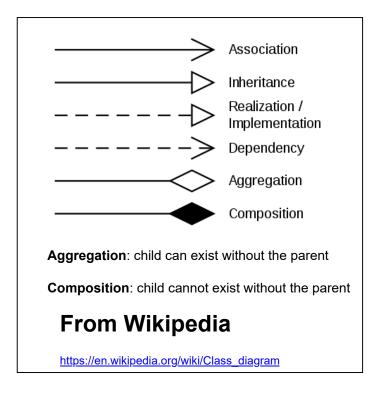
order to fulfill GradingService requirements

handling purposes

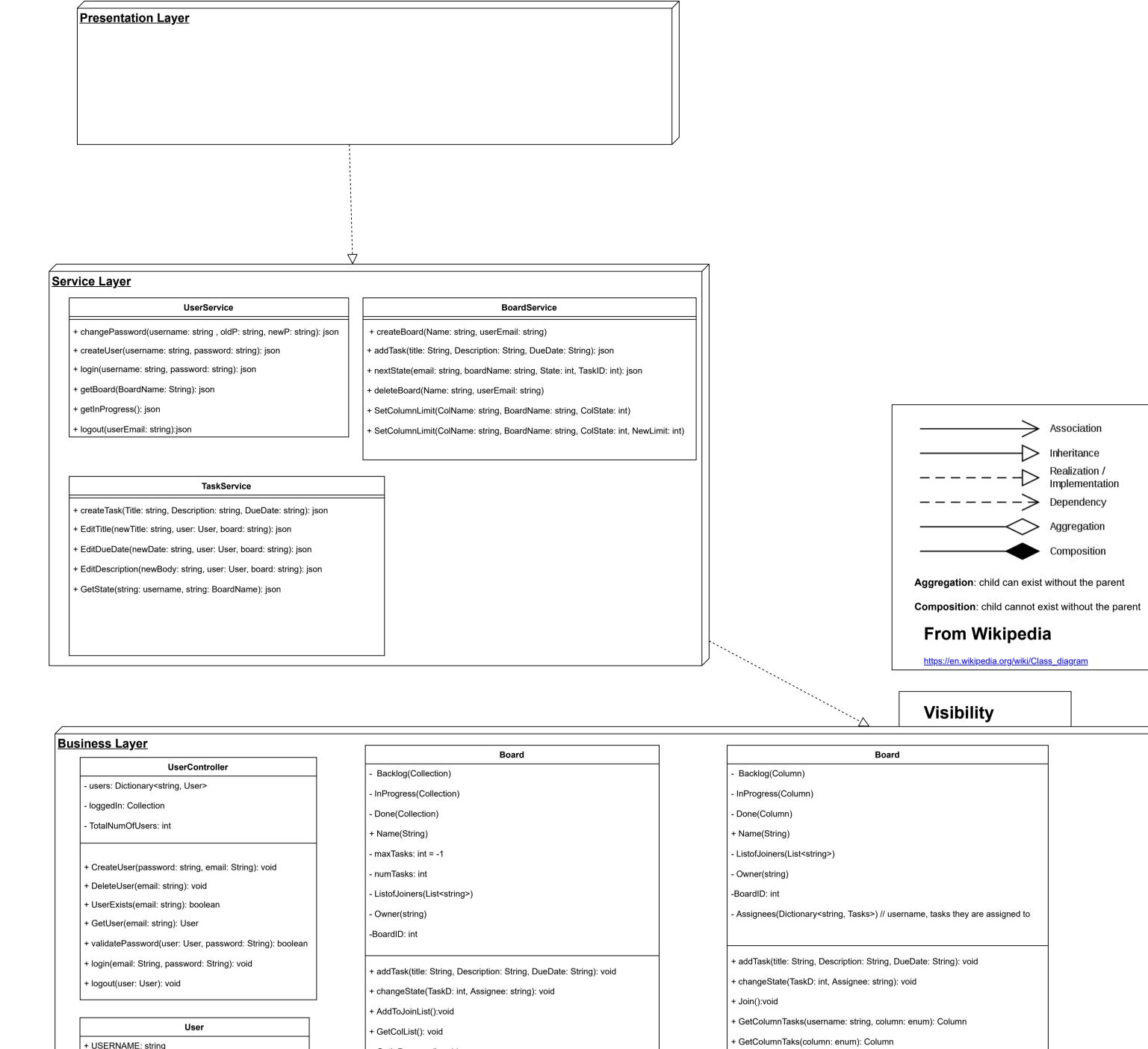
BoardController - Title(String) Dictionary<string(username), <string(boardname), Board> > - DueDate(String) - Dictionary<board_id(int), Board> + CREATION DATE(DateTime) - TotalNumOfBoards: int Description(newBody: string): void + Exists(Name: String): Boolean - State(int) + createBoard(Name: String): void + TaskID(int) + deleteBoard(Name: String): void - Assignee(string) + getBoard(Name: String): void + changeMax(newMax: Int): void + DeleteBoard(owner: string, BoardName: string): void + JoinBoard(Joiner: username, BoardOwner: string, BoardName: string): void + LeaveBoard(Leaver: string, BoardName: string): void + EditTitle(newTitle: string): void + IsOwner(BoardID: int, (OR) BoardName: string): bool + EditDueDate(newDate: String): void + ChangeOwner(NewOwner: string, OldOwner: string,) + EditDescription(newBody: string): void + EditAssignee(newAssignee: string):void + Getters + Setters



+ DeleteBoard(): void



Visibility
+ public
- private
protected



Changes Made after submitting Milestone 1:

HasAnyBoard- Check if a BoardController is empty - for exception

GetColList - in order to get all tasks in a given column in a board

GetNameOrdinal - Returns the name of a column given its id in

A User object no longer holds a collection of their boards and

The class user is therefore connected only to the UserConroller,

UserController.IsLoggedIn(). This allows a better (lower) duality

Setting the max number of tasks in now done from the Board

Changes Made before starting Milestone 2

INew methods for all new functional requirements. Data Access

it is an attribute of a board rather than a task. It can be said that

The checks in order to get a user's board and tasks are preformed

GetAllInProgress - See design changes for explanation

HasThisBoard- Checks if a BoardController has a board - for

IsLegalPassword- in order to validate the password

IsValidEmail- To validate the email address at registration

Created:

IsHebrew - to validate email

exception handling purposes

Added getters and setters in Board

order to fulfill GradingService requirements

handling purposes

Design changes:

which limits our duality.

between the classes.

Created:

Layer DTO's and Mappers.

Design changes:

through calling an instance of

(rather than Task). This is done because

initially we made a design mistake.

AddTask - same as max tasks.

