Presentation Layer Service BoardService uc: UserController + bc: BoardController LoadBoards(bc:boardController): Response Service layer + LoadData(): Response DeleteAll(bc:boardController): Response F DeleteData(): Response + GetUserBoards(bc:boardController, string userEmail): Response<List<Board>> + AddBoard(bc:BoardController, userEmail:string, boardName:string): Response + RemoveBoard(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string): Response Response - id: int {get} - id: int {get} + JoinBoard(bc: BoardController, userEmail:string, creatorEmail:string, boardName:string): Response + ErrorMessage: string - name: string {get} - creationTime: DateTime {get} creatorEmail: string {get} title: string {get} UserService + GetBoardNames(bc: BoardController, userEmail:string): Response<IList<String>> - columnDict: Dictionary<int, Column> {get} - membersList: IList<string> {get} description: string {get} dueDate: DateTime {get} + InProgressTasks(bc:boardController, userEmail:string): Response<IList<Task>> ResponseT - emailAssignee: string {get} + LoadUsers(uc:UserController): Response + LimitColumn(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, limit:int): Response + Value: T User Column + DeleteAll(uc:UserController): Response + GetColumnLimit(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int): Response<int> - email: string {get} name: string {get} + Register(uc:UserController, userEmail:string, password:string): Response - taskLimit: int {get} + GetColumnName(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int): Response<string> - taskList: IList<Task> {get} + LogIn(uc:UserController, userEmail:string, password:string): Response<User> + GetColumn(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int): Response<IList<Task>> + LogOut(uc:UserController, userEmail:string): Response + AddTask(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, title:string, description:string, dueDate:DateTime): Response<Task> + UpdateTaskDueDate(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, taskId:int, dueDate:DateTime): Response + UpdateTaskTitle(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, taskld:int, title:string): Response + UpdateTaskDescription(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, taskId:int, description:string): Response + AdvanceTask(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, taskId:int): Response + AssignTask(bc:BoardController, userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, taskld:int, emailAssignee:string): Response + AddColumn(userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, columnName:string): Response $+ \ Remove Column (user Email: string, \ creator Email: string, \ board Name: string, \ column Ordinal: int): \ Response$ + RenameColumn(userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, newColumnName:string): Response + MoveColumn(userEmail:string, creatorEmail:string, boardName:string, columnOrdinal:int, shiftSize:int): Response boardController boardTop: int {get} - uc: UserController boardDict : dict<id,Board> - idDict : dict<(creatorEmail,boardName),boardId> **Business layer** - userTop: int {get} GetBoard(userEmail:string, creatorEmail:string, boardName:string):Board + GetUser(userEmail:string): User + AddBoard(creatorEmail:string, boardName:string): void + LoadUsers(): void + RemoveBoard(creatorEmail:string, boardName:string, userEmail:string): void + Delete():void + JoinBoard(userEmail: string,creatorEmail:string, boardName:string): void + Register(userEmail:string, password:string): void + GetBoardNames(userEmail:string): IList<String> password: stringCommonPasswords: string[] + Login(userEmail:string, password:string): User + InsertUserToDal(user: DUser): void + validLogin(userEmail:string, funcName:string):bool - ConvertTold(creatorEmail:string, boardName:string);int - IsValidEmail(userEmail:string): void Uses + InProgressTasks(userEmail:string):IList<Task> validLogin(userEmail:string):bool ValidPassword(password:string): bool + LoadBoards(): void LoadBoardMembers():void - CheckMin(password:string): bool CheckMax(password:string): bool + DeleteAllBoards(): void CheckUpperCase(password:string): bool validLogin(userEmail:string , funcName: string): void · CheckLowerCase(password:string): bool - CheckNumber(password:string): bool - CheckCommon(password:string): bool 0:n Boards Board id: int {get}boardName: string {get} creatorEmail: string {get} - taskTop: int - DBoard: DBoard {get} - membersList: List<string> {get} Task name: string {get, set} columns : Dict<int, Column> 0:n Users - id: int {get}- creationTime: DateTime {get} minimumColumns : int taskLimit:int {get, set} -TasksList {get} - dColumn: DColumn {get,set} - title: string {get} + AddUserToList(userEmail:string):void 2:n Columns - description: string {get } + email: string {get} + GetColumn(memberEmail:string, columnOrdinal:int): Column + AddColumn(name:string, columnOrdinal:int): void - dueDate: DateTime {get} + RenameColumn(name:string): void + SetColumOrdinal (ordinal : int): void - assignee: string {get} - maxTitleLength: int loggedIn: bool {get} RemoveColumn(columnOrdinal:int): void id: int {get}DUser : dUser {get}List <int> boardList {get} + MoveColumn(columnOrdinal:int, shiftSize:int): void + GetTaskList(userEmail:string): List<Task> - maxDescriptionLength:int + AddTask(userEmail: string, title:string, description:string, dueDate:DateTime): void + AddTask(newTask :Task ,boardId:int, columnOrdinal:int): void - dTask: Dtask {get,set} + InProgressTasks(userEmail:string):List<Task> + RemoveTask(task:Task): void - Contains(taskld:int): bool - GetTask(taskld:int): Task - LoadTasks(boardld: int , columnOrdinal):void - DeleteTasks():void AdvanceTask(userEmail:string, columnOrdinal:int, taskId:int): void + LogIn(): void - ValidTitle(title:string): bool CheckOrdinal(columnOrdinal:int): bool ValidTaskChangeable(columnOrdinal:int, functionName: string): void ValidMember(userEmail:string): void - ValidDescription(description:string): bool + SetTitle(userEmail:string, newTitle:string): void + LogOut() void + ConfirmPassword(password:string): bool + SetDescription(userEmail:string, newDescription:string): void + SetDueDate(userEmail:string, newDueDate:DateTime): void + AddBoardToList(boardId:int);void + DeleteBoardFromList(boardId:int)void + LoadColumns(): void + SetAssignee(userEmail:string, newAssigneeEmail:string): void + CheckAssignee(assigneeEmail:string): bool + DeleteColumn():void **Data Accses Layer** DTask _ boardId: int {get} DBoard _columnOrdinal: int {get,set} _assignee: string {get,set} _taskTitle: string {get , set} _columnOrdinal: int {get, set} name: string {get}creatorEmail: string {get} - name: string {get,set} DUser __taskDescription: string {get , set} _taskCreationTime: DateTime {get} taskLimit: int {get,set} - taskTop: int {get,set} הורש מהמחלקה האבסטרקתית///// email: string {get} _userEmail: string {get} _taskDueDate: DateTime{get , set} הורש מהמחלקה האבסטרקתית///// - _boardId: int {get} הורש מהמחלקה האבסטרקתית///// -_controller: DalMemberController password: string {get} - id: int controller: DalColumnController הורש מהמחלקה האבסטרקתית///// _controller: DalBoardController - id: int {get} - _controller: DalUserController boardId: int controller: DalTaskControlle + Insert(): bool Insert(): bool + Insert(): bool + Delete():bool + Delete():bool + insertUser():boo + Insert():bool + Delete(): bool + Delete():bool DTO - id: int {get} _controller: DalControlle DalUserController + InsertUser(user:DUser, idColumn: string, DalBoardController DalColumnController DalTaskController emailColumn: string , passwordColumn: string): bool + LoadUsers():IList<DUser> DalMemberController לא ממושב באמת, ממומש במחלקה האבסטרקתית///// + Insert(board:DBoard): bool + LoadBoards():List<DBoard> + Insert(dColumn: DColumn): bool + LoadBoardColumns(boardId:int):List<DColumn> + Insert(task: DTask): bool + DeleteAll(): bool + Insert(DMember):bool + LoadBoardTasks(boardId:int):List<DTask> ConvertReaderToObject(SQLiteDataReader reader): DTO + Update(boardId: int, columnOrdinal: int, columnName:string, value:int): bool + Update(boardId: int, columnOrdinal: int, columnName:string, value:string): bool + LoadBoardMembers(): List<DMember> + Update(Id: int, columnName:string, value:int):bool + LoadColumnTasks(boardId:int , columnOrdinal: int):List<DTask> - ConvertReaderToObject(SQLiteDataReader: reader) + ConvertReaderToObject(SQLiteDataReader reader) : DTO + Update(id: int, boardId: string, columnName:string, value:int): bool - email: string {get} + ConvertReaderToObject(SQLiteDataReader reader) : DTO + Update(id: int, columnName: string, value:string): bool לא ממושב באמת, ממומש במחלקה האבסטרקתית////// לא ממושב באמת, ממומש במחלקה האבסטרקתית////// password: string {get} + DeleteAll(): bool + Delete(DTO Obj):bool לא ממושב באמת, ממומש במחלקה האבסטרקתית////// + ConvertReaderToObject(SQLiteDataReader reader) : DTO הורש מהמחלקה האבסטרקתית///// + Delete(DTO Obj):bool + DeleteAll(): bool + Delete(DTO Obj):bool לא ממושב באמת, ממומש במחלקה האבסטרקתית///// DalController - id: int {get} + DeleteAll(): bool + Delete(DTO Obj):bool controller: DalUserCor + DeleteAll(): bool _connectionString: string _tableName: string + Select(): List<DTO> + DeleteAll(): bool + ConvertReaderToObject(SQLiteDataReader reader) : DTO DB