Service layer

Service

+ LoadData(): Response
+ DeleteData(): Response
+ Register(userEmail: String, password: String): Response
+ Login(userEmail: String, password: String): Response<SUser>
+ Logout(userEmail: String): Response
+ LimitColumn(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, limit: int): Response
+ GetColumnLimit(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int): Response<int>
+ GetColumnName(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int): Response<String>
+ AddTask(userEmail: String, creatorEmail: String, boardName: String, title: String, description: String, dueDate: DateTime): Reponse<STask>
+ UpdateTaskDueDate(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, dueDate: DateTime): Reponse
+ UpdateTaskTitle(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, title: String): Reponse
+ UpdateTaskDescription(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, description: String): Reponse
+ AdvanceTask(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int): Response
+ GetColumn(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int): Response<IList<Task>>
+ AddBoard(userEmail: String, boardName: String): Response
+ JoinBoard(string userEmail, string creatorEmail, string boardName): Response
+ RemoveBoard(userEmail: String, creatorEmail: String, boardName: String): Response
+ InProgressTasks(userEmail: String): Response<IList<Task>>
+ AssignTask(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, emailAssignee: String): Response
+ GetBoardNames(string userEmail): Response<IList<String>>

Response<T>

+ T value

~ static FromValue(value T): Response<T>
~ static FromError(msg T): Response<T>

Extends

Response

+ ErrorMessage: String
+ ErrorOccured(): bool

STask

+ Id: int
+ CreationTime: DateTime
+ Title: String
+ Description: String
+ DueDate: DateTime
+ emailAssignee: String

SBoard

+ CreatorEmail: String
+ Name: String
+ Columns: IList<String>
+ TaskCounter: int

SUser

+ email: String

SColumn

+ Name: String
+ Limit: int
+ Tasks: IList<STask>

BoardService

~ LoadData(): Response
~ DeleteData(): Response
~ LimitColumn(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, limit: int): Response
~ AddTask(userEmail: String, creatorEmail: String, boardName: creationTime: DateTime, title: String, description: String, DueDate: DateTime): Reponse<STask>
~ UpdateTaskDueDate(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, dueDate: DateTime): Reponse
~ UpdateTaskTitle(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, title: String): Reponse
~ UpdateTaskDescription(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, description: String): Reponse
~ AdvanceTask(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int): Response
~ GetColumn(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int): Response<SColumn>
~ AddBoard(userEmail: String, boardName: String): Response
~ JoinBoard(userEmail: String, creatorEmail: String, boardName: String): Response
~ RemoveBoard(userEmail: String, creatorEmail: String, boardName: String): Response
~ InProgressTasks(userEmail: String): Response<IList<STask>>
~ AssignTask(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, emailAssignee: String): Response
~ GetBoardNames(string userEmail): Response<IList<String>>
- translateList(originalList: IList<BusinessLayer.Task>): IList<STask>

UserService

~ LoadData(): Response
~ DeleteData(): Response
~ Register(userEmail: String, password: String): Response
~ Login(userEmail: String, password: String): Response<User>
~ Logout(userEmail: String): Response

visibility
+ public
- private
~ internal
# protected

Association
Inheritance
Realization / Implementation
Dependency
Aggregation
Composition

From Wikipedia

https://en.wikipedia.org/wiki/Class_diagram

Business layer

BoardController

- userBoards: dict< userEmail: String, HashSet<boardID: String>>

~ LoadData(): void
~ DeleteData(): void
~ GetBoardNames(string userEmail): IList<String>
~ AddBoard(userEmail: String, name: String): void
~ JoinBoard(userEmail: String, creatorEmail: String, boardName: String): void
~ RemoveBoard(userEmail: String, creatorEmail: String, boardName: String): void
~ LimitColumn(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, limit: int): void
~ AddTask(userEmail: String, creatorEmail: String, boardName: String,creationTime: DateTime, title: String, description: String, DueDate: DateTime): BTask
~ AssignTask(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, emailAssignee: String): void
~ UpdateTaskDueDate(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, dueDate: DateTime): void
~ UpdateTaskTitle(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, title: String): void
~ UpdateTaskDescription(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int, description: String): void
~ AdvanceTask(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int, taskId: int): void
~ GetColumn(userEmail: String, creatorEmail: String, boardName: String, columnOrdinal: int): BColumn
~ GetBoard(userEmail: String, creatorEmail: String, boardName: String): BBoard
~ InProgressTasks(userEmail: String): IList<BTask>
- checkColumnOrdinal(creatorEmail: String, boardName: String, columnOrdinal: int): void
- checkBoardExistance(creatorEmail: String, boardName: String, action: String): void
- checkMembership(userEmail: String, creatorEmail: String, boardName: String): void
- validateLogin(userEmail: String, attempt: String): void

User Controller

- const PASS_MIN_LENGTH :int
- const PASS_MAX_LENGTH; int

~ LoadData(): void
~ DeleteData(): void
~ Register(userEmail: String, password: String): void
~ Login(userEmail: String, password: String): BUser
~ Logout(userEmail:String): void
- validateEmail(email: String) :void
- validatePasswordRules(password: String): void

1
0..n

BUser

~ readonly email: String
- password: String

~ validatePassword(password: String): bool

LoginInstance

~ readonly ConnectedEmail: String

~ Login(userEmail: String): void
~ ValidateLogin(userEmail: String): void
~ Logout(userEmail: String

1
0..n

BBoard

- taskIdCounter: int

~ AddMember(memberEmail: String): void
~ LimitColumn(columnOrdinal: int, limit: int): void
~ AddTask(creationTime: DateTime, title: String, description: String, DueDate: DateTime): BTask
~ AssignTask(userEmail: String, columnOrdinal: int, taskId: int, assignee: String): void
~ UpdateTaskDueDate(columnOrdinal: int, taskId: int, dueDate: DateTime): void
~ UpdateTaskTitle(columnOrdinal: int, taskId: int, title: String): void
~ UpdateTaskDescription(columnOrdinal: int, taskId: int, description: String): void
~ AdvanceTask(columnOrdinal: int, taskId: int): void
~ GetColumn(columnOrdinal: int): BColumn
~ GetColumnTasks(columnOrdinal: int): IList<BTask>

BColumn

~ name: String
~ limit: int
- dColumn: DColumn

~ AddTask(taskId: int, creationTime: DateTime, title: String, description: String, DueDate: DateTime): Task
~ AddTask(task: Task): void
~ RemoveTask(taskId: int): Task
~ UpdateTaskDueDate(taskId: int, dueDate: DateTime): void
~ UpdateTaskTitle(taskId: int, title: String): void
~ UpdateTaskDescription(taskId: int, description: String): void
- validateAssignee(userEmail: String, taskId: int): void

1
3

BTask

~ taskId :int
~ creation time: Date
~ dueDate: Date
~ title: String
~ description: String
~ assignee: String
- dTask: DTask
- const MAX_TITLE_LENGTH: int
- const MIN_TITLE_LENGTH: int
- const MAX_DESCRIPTON_LENGTH: int

~ Advance() : void
- validateDescription(description : string): voic
- validateTitle (title :string) :void
- validateDueDate(dueDate : DateTime): void

0..n
1

Data Access Layer

DBoardController

~ override Select(): List<DBoard>
~ override DeleteAll(): void
~ DeleteBoard(creatorEmail:String , boardName: String): void

DBoard

~ readonly CreatorEmail: String
~ BoardName: String
~ Members: HashSet<String>
- const COL_BOARD_NAME: String
- const COL_CREATOR_EMAIL: String

~ numberOfTasks(): int
~ AddMember (memberEmail: String): void

1
3

DColumn

~ CreatorEmail: String
~ BoardName: String
~ readonly Ordinal: int
~ Limit: int
- const COL_CREATOR_EMAIL: String
- const COL_BOARD_NAME: String
- const COL_LIMIT: String
- const COL_ORDINAL: String

1
0..n

DTask

~readonly TaskId: int
~ Title: String
~ Description: String
~ readonly CreationTime: DateTime
~ DueDate: DateTime
~ Assignee: String
~Ordinal: int
~ BoardCreator: String
~ BoardName: String
-const COL_TASK_ID
-const COL_TITLE
-const COL_DESCRIPTION
-const COL_CREATIONTIME
-const COL_DUE_DATE
-const COL_ASSIGNEE
-const COL_ORDINAL
-const COL_CREATOR_EMAIL
-const COL_BOARD_NAME

DUser

- readonly Email: String
- Password: String
- const COL_EMAIL: String
- const COL_PASSWORD: String

DColumnController

- const COL_CREATOR_EMAIL: String
- const COL_BOARD_NAME: String

~ Select(boardCreator: String, boardName: String): List<DColumn>
~ override DeleteAll(): void
~ DeleteBoardColumns(creatorEmail: String , boardName: String): void

BoardMemberController

- readonly_connectionString: String
- const _tableName: String
- const CAL_ID: String
- const CAL_USER: String

~ Select(ID: String): HashSet<String>
~ DeleteAll(): void
~ DeleteBoardMembers(creatorEmail: String, boardName:String): void
~ Insert(ID: String, userEmail: String): void
~ Delete(): void

DUserController

DTaskController

- const COL_ORDINAL: String
- const COL_CREATOR_EMAIL: String
- const COL_BOARD_NAME: String

~ Select(boardCreator: String, boardName: String, ordinal: int): List<DTask>
~ DeleteBoardtask(creatorEmail: String, boardName: String): void

<<abstract>> DalController <T> where T : DTO

readonly _connectionString: String
readonly _tableName: String
const COL_ID: String

~ virtual List<T> Select()
~ virtual void DeleteAll()
# abstract T ConvertReaderToObject(SQLiteDataReader reader);

<<abstract>> DTO

~ ID: String
# const COL_ID: String
# readonly _connectionString: String
# readonly _tableName: String
~ Persist: Boolean

~ Insert(): Boolean
# Update(string attributeName, string attributeValue): Boolean
# Update(string attributeName, long attributeValue): Boolean
# Update(attributeName: String, attributeValue: DateTime): Boolean
# abstract InsertCommand(connection: SQLiteConnection): SQLiteCommand

db