

service layer

ServiceFactory

-userservice:UserService -boardservice:BoardService -columnservice:ColumnService -taskservice:TaskService -dataservice:DataService

+ Register(email:string,password:String):json

+ login(email:string,password:string):json

+ logout(email:string):json

+ limitcoulmn(email:string,boardname:string,coulmnOrdinal:int,limit:int):json

AddTask(email:string,boardname:string,title:string,description:string ,dueDate:datetime):json

updateTaskDueDate(email:string,boardname:string,columnOrdinal:int, taskID:int dueDate:DateTime):json

+ updateTaskTiltle(email:string,boardname:string,columnOrdinal:int, taskID:int, title:string(:json

+ updateTaskTDescription(email:string,boardname:string,columnOrdinal:int, taskID:int,description:string):json

+ advanceTask(email:string,boardname:string,columnOrdinal:int,TaskID:int):json

+ AddBoard(email:string,name:String):json

removeBoard(email:string,name:string):json

+ inProgressTasks(email:string).json +LeaveBoard(email:string, boardld:int):json +AssignTask(email:string,boardname:string,columnOrdinal:int , taskID:int, emailAssignee:string):json +JoinBoard(email:string, boardId:int):json

+DeleteData():json

+LoadData():json

+TransferOwnership(currentOwnerEmail:string ,newOwnerEmail:string , boardName:string):json

DataBaseController

+updateTaskDescription(taskID:int, description:string):void

+updateTaskDueDate(taskid:int, DateTime:duedate):void

User

instance:DataBaseController

getInstance():DataBaseController

+removeBoard(boardID:int):void

+removeTask(taskID:int):void

+advanceTask(taskID:int):void

+getBoardIDCounter():int

+getTaskIDCounter():int

<loadUsers():LinkedList<User

+addUser(userDTO:UserDTO):void

+loadBoards():void

+addBoard(boardDTO:BoardDTO):void

+addTask(taskDTO:TaskDTO, boardID,int):void

+addTask(taskDTO:TaskDTO,boardID:int):void

+updateTaskTitle(taskID:int, title:string):void

+AssignTask(taskId:int,email:string):void

GradingService

-serviceFactory:ServiceFactory + Register(email:string,password:String):json

+ login(email:string,password:string):json

+ logout(email:string):json

+ limitcoulmn(email:string,boardname:string,coulmnOrdinal:int,limit:int):json

+ getColumnLimit(email:string,boardname:string,coulmnOrdinal:int):json

+ AddTask(email:string,boardname:string,title:string,description:string ,dueDate:datetime):json

+ getColumnName(email:string,boardname:string,coulmnOrdinal:int):json

+ updateTaskDueDate(email:string,boardname:string,columnOrdinal:int, taskID:int

+ updateTaskTiltle(email:string,boardname:string,columnOrdinal:int, taskID:int,

title:string(:json

+ updateTaskTDescription(email:string,boardname:string,columnOrdinal:int,

taskID:int,description:string):json

+ advanceTask(email:string,boardname:string,columnOrdinal:int,TaskID:int):json

+ getColumn(email:string,boardname:string,columnOrdinal:int):json + AddBoard(email:string,name:String):json

+ removeBoard(email:string,name:string):json

+ inProgressTasks(email:string).json +GetUserBoards(email:string):json

+GetBoardName(boardId:int):json +LeaveBoard(email:string, boardId:int):json

+AssignTask(email:string,boardname:string,columnOrdinal:int, taskID:int, emailAssignee:string):json +JoinBoard(email:string, boardId:int):json +LoadData():json +DeleteData():json

 $+ Transfer Owner Ship (current Owner Email: string\ , new Owner Email: string\ ,\ board Name: string\): js on$

BoardService

-instance:dataBaseController

--instance:BoardContoller

,string boardName):json

+ AddBoard(email:string,name:string):json

+GetBoardController():BoardController

+ GetBoardController():Boardcontroller +JoinBoard(emailToJion:string, boardID:int):json +LeaveBoard(emailToLeave:string , boardID:int):json +TransferBoard(string email,string transferTo

+ RemoveBoard(email:string,name:string):json

ColumnService

--instance:BoardContoller

+ LimitColumn(email:string,boardname:json ,coulmnOrdinal:int,limit:int):json

+ InprogressTasks(email:String):json

taskService

instance:dataBaseController

--instance:BoardContoller

- AdvanceTask(email:string,boardname:string,columnOrdinal:int,taskid:int):json

+ AssignTask(email:string,boardname:string,columnOrdinal:int,taskid:int,emailassign:string):json

+ UpdateTaskDueDate(email:string,boardname:string,columnOrdinal:int,taskid:in,dueDate:DateTime):json + UpdateTaskTitle(email:string,boardname:string,columnOrdinal:int,taskid:in,title:string):json

+ UpdateTaskDescription(email:string,boardname:string,columnOrdinal:int,taskid:in,Description:string):json + AddTask(email:string,boardname:string,title:string,description:string,dueDate:DateTime):json

UserService

+ Errormessage : string -instance:dataBaseController

+ Register(email:string,password:string):json

+ Login(email:string,password:string):json

+ Logout(email:string):json

+ ReturnValue: string -instance:UserContoller

DataService

Response

-instance:BoardController instance:UserController instance:dataBaseController

+LoadData():json +DeleteData():json

BussinessLayer:

ServiceLayer:

service,taskservice to

layers,so we

layer class

object as ocurred

design pattern.

1. we added new service layers; column

make it more easy and elegant to reach bussiness

made a service layer class for each bussiness

2.we added the class response to build json

4.we added DataService for the database.

3.we added ServiceFactory class to improve the

to be returned in service layer classe

1. we added new controller;BoardController to save all the boards and tasks for each user

2.we added new class; column that has name, limit and contains a list of tasks 3.we added DataBaseController class to connect the bussinessLayer with the DataAccessLayer

> email: string - password:string loggedIn:boolean -userDTO:UserDTO

logIn():string + logout():string

User Controller users: LinkedList<User>

getInstance():UserController

-instance:UserContoller

- addUser(email:string,password:string):string

+ GetUser(email:string):user

+ GetUsers():LinkedList<User>

+loadData():void

+ Logout(email:string):string + Login(email:string,password:string):string

BoardController instance:BoardController -boards:Linkedlist<string, LinkedList<Board>>

bussiness layer

askIDCounter int -bordID:int

getInstance():BoardController

- addUser(email:string):void +AssignTask(email:string,boardname:string, columnOrdinal:int,taskid:int,assignemail:string):string

+ addBoard(email:string,name:string):string

+ limitColumn(email:string,boardname:string,

+ removeBoard(email:string,name:string):string

columnOrdinal:int,limit:int):string

+ AddTask(email:string,boardName:string,title:string, description:string,dueDate:Datetime):string

+ updatetaskDescription(email:string,boardname:string columnOrdinal:int,taskID:int,description:string):string

+ updatetaskDueDate(email:string,boardname:string,columnOrdinal:int, taskID:int,dueDate:DateTime):string

+ updatetasktitle(email:string,boardname:string,columnOrdinal:int, taskid:int,title:string):string

+ advanceTask(email:string,boardname:string,columnOrdinal:int,taskid:int):string

+GetTask(taskID:int):Task

+GetUserBoards(email:string):LinkedList<Board>

+loadData():void

+JoinBoard(email:string,boardId:int):string +LeaveBoard(email:string,boardId:int):string +TransferBoard(email:string,transferemail:string,boardName:string):string

GetInprogressTasks(email:string):LinkedList<Task> + GetBoard(email:string,boardname:string):Board

coulmns:LinkedList<Column> - name: string -boardDTO:BoardDTO

DueDate:DateTime):string

+JoinBoard(string emailToJion):string +LeaveBoard(string emailToLeave):string +TransferBoard(string transferTo):string + Addtask(id:int,title:string,description:string,

-joinedUsers:LinkedList<string>

taskid:int,Descrptiong:string):string - updatetaskDueDate(columnOrdinal:int, taskid:int,DueDate:DateTime):string

+ updatetaskDescription(columnOrdinal:int,

+AssignTask(columnOrdinal:int, taskid:int,emailAssign:string):string + updatetasktitlen(columnOrdinal:int,

taskid:int,title:string):string +isJoined(email:string):bool

+ limitColumns(limit:int,id:int):string getColumnLimit(columnID:int):int

getColumnName(columnID:int):string

-AdvanceTask(columnOrdinal:int;taskID:int):string - getColumn(columnOrdinal:int): LinkedList<Task>

+ GetInprogressTasks():LinkedList<Task> GetTask(columnID;int,taskID:int)Task

Task

creationTime:DateTime dueDate:DateTime title: string - description: string -ID:int

-person:string

-taskDTO:TaskDTO +AssignTask(person:string):string

UpdateTaskDescription(description:string):string

UpdateTaskDueDate(dueDate:datetime):string

+ UpdateTasktitle(title:string):string

+ UpdateTaskDescription(taskid:int,description:string):string

+ removeTask(id:int):boolean

name:string

- Limit:init

- tasks:LinkedList<Tasks>

+ UpdateTaskDueDate(taskid:int,dueDate:DateTime):string

+ AddTask(id:int,title:string,description:string,

DueDate:DateTime,person:string):string

Column

+ UpdateTaskTitle(taskid:int,Title:string):string

+ limitColumns(limit:int):void

+ GetTask(TaskID:int):Task

+AssignTask(taskId:int,email:string):string

DataAccessLayer

BoardDTO

Kanban

-taskDAO:TaskDAO -boardDAO:BoardDAO -taskInboard:TaskInBoardDAO -userDAO:UserDAO -instance:kanban - connection:SQLite Connection

getInstance():Kanban +create():void +close():void +open():void +DeleteData():void + loadBoardsDTO():LinkedList<BoardDTO> -loadTasksInBoards():SQLiteDataReader

connection:SQLite Connection

insert(userDTO:UserDTO):void

loadUsers():LinkedList<UserDTO>

UserDAO

-email:string

UserDTO -passward:string

boardname:string -boardld:string -email:string -owner:string

BoardDAO

- connection:SQLite Connection + boardIDCounter():int +insert(| boardDTO:BoardDTO):void +removeBoard(boardID:int):void loadBoards():LinkedList<BoardDTO> +leaveBoard(id:int,emailtoleave:string):void

+TransferBoard(boardid:int,transferto:string,email:string):void

TaskInBoardDTO

-boardId:int -taskId:int

TaskInBoardDAO connection:SQLite Connection +removeTask(taskID:int):void +insert(taskInBoardDTO:TaskInBoard):void +removeBoard(boardID:int):void

TaskDTO -taskid:int -creationTime:DateTime -duedate:DateTime -title:string -description:string -email:string -columnID:int

TaskDAO

- connection:SQLite Connection

+insert(taskDTO:TaskDTO):void +advanceTask(taskID:int):void +updateTaskDescription(taskID:int,description:string):void +updateTaskDueDate(taskID:int,dueDate:string):void +removeTask(taskID:int):void

+TaskIdCounter+():int +AssignTask(taskid:int,email:string):void +updateTaskTitle(taskID:int,title:string):void

2.we added DAOS classes for saving the DTOS 3.we added Kanban class to improve the design

DataAccessLayer I. we added DTOS classes :userDTO,TaskDTO,TaskInBoardDTO,BoardDTO :userDAO,TaskDAO,TaskInBoardDAO,BoardDAO