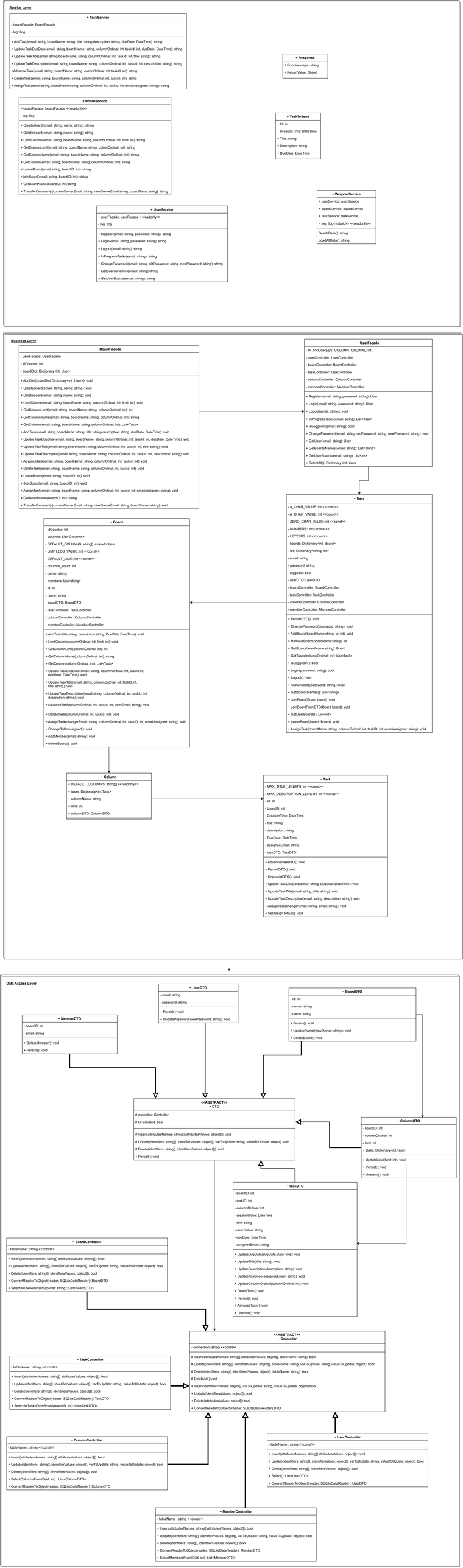
Service Layer
+ TaskService
- boardFacade: BoardFacade
- log: flog
+ AddTask(email: string,boardName: string, title: string,description: string, dueDate: DateTime): string
+ UpdateTaskDueDate(email: string, boardName: string, columnOrdinal: int, taskId: int, dueDate: DateTime): string
+ UpdateTaskTitle(email: string,boardName: string, columnOrdinal: int, taskId: int, title: string): string
+ UpdateTaskDescription(email: string,boardName: string, columnOrdinal: int, taskId: int, description: string): string
+AdvanceTask(email: string, boardName: string, colmnOrdinal: int, taskId: int): string
+ DeleteTask(email: string, boardName: string, columnOrdinal: int, taskId: int): string
+ AssignTask(email string, boardName:string, columnOrdinal: int, taskId: int, emailAssignee: string): string
+ Response
+ ErrorMessage: string
+ ReturnValue: Object
+ BoardService
~ boardFacade: boardFacade <<readonly>>
- log: flog
+ CreateBoard(email: string, name: string): string
+ DeleteBoard(email: string, name: string): string
+ LimitColumn(email: string, boardName: string, columnOrdinal: int, limit: int): string
+ GetColumnLimit(email: string, boardName: string, colmnOrdinal: int): string
+ GetColumnName(email: string, boardName: string, columnOrdinal: int): string
+ GetColumn(email: string, boardName: string, columnOrdinal: int): string
+ LeaveBoard(email:string, boardID: int): string
+JoinBoard(email: string, boardID: int): string
+ GetBoardName(boardID: int):string
+ TransferOwnership(currentOwnerEmail: string, newOwnerEmail:string, boardName:string): string
+ TaskToSend
+ Id: int
+ CreationTime: DateTime
+ Title: string
+ Description: string
+ DueDate: DateTime
+ UserService
~ userFacade: userFacade <<readonly>>
- log: flog
+ Register(email: string, password: string): string
+ Login(email: string, password: string): string
+ Logout(email: string): string
+ InProgressTasks(email: string): string
+ ChangePassword(email: string, oldPassword: string, newPassword: string): string
+ GetBoardsNames(email: string):string
+ GetUserBoards(email: string): string
+ WrapperService
+ userService: userService
+ boardService: boardService
+ taskService: taskService
~ log: flog<<static>> <<readonly>>
DeleteData(): string
LoadAllData(): string
Business Layer
~ BoardFacade
- userFacade: UserFacade
- idCounter: int
- boardDict: Dictionary<int, User>
+ AddDict(boardDict:Dictionary<int, User>): void
+ CreateBoard(email: string, name: string): void
+ DeleteBoard(email: string, name: string): void
+ LimitColumn(email: string, boardName: string, columnOrdinal: int, limit: int): void
+ GetColumnLimit(email: string, boardName: string, columnOrdinal: int): int
+ GetColumnName(email: string, boardName: string, columnOrdinal: int): string
+ GetColumn(email: string, boardName: string, columnOrdinal: int): List<Task>
+ AddTask(email: string,boardName: string, title: string,description: string, dueDate: DateTime): void
+ UpdateTaskDueDate(email: string, boardName: string, columnOrdinal: int, taskId: int, dueDate: DateTime): void
+ UpdateTaskTitle(email: string,boardName: string, columnOrdinal: int, taskId: int, title: string): void
+ UpdateTaskDescription(email: string,boardName: string, columnOrdinal: int, taskId: int, description: string): void
+ AdvanceTask(email: string, boardName: string, colmnOrdinal: int, taskId: int): void
+ DeleteTask(email: string, boardName: string, columnOrdinal: int, taskId: int): void
+ LeaveBoard(email: string, boardID: int): void
+ JoinBoard(email: string, boardID: int): void
+ AssignTask(email: string, boardName: string, columnOrdinal: int, taskId: int, emailAssignee: string): void
+ GetBoardName(boardID: int): string
+ TransferOwnership(currentOwnerEmail: string, newOwnerEmail: string, boardName: string): void
~ UserFacade
- IN_PROGRESS_COLUMN_ORDINAL: int
- userController: UserController
- boardController: BoardController
- taskController: TaskController
- columnController: ColumnController
- memberController: MemberController
+ Register(email: string, password: string): User
+ Login(email: string, password: string): User
+ Logout(email: string): void
+ InProgressTasks(email: string): List<Task>
+ IsLoggedIn(email: string): bool
+ ChangePassword(email: string, oldPassword: string, newPassword: string): void
+ GetUser(email: string): User
+ GetBoardsNames(email: string): List<string>
+ GetUserBoards(email: string): List<int>
+ SelectAll(): Dictionary<int,User>
~ User
- a_CHAR_VALUE: int <<const>>
- A_CHAR_VALUE: int <<const>>
- ZERO_CHAR_VALUE: int <<const>>
- NUMBERS: int <<const>>
- LETTERS: int <<const>>
- boards: Dictionary<int, Board>
- ids: Dictionary<string, int>
- email: string
- password: string
- loggedIn: bool
- userDTO: UserDTO
- boardController: BoardController
- taskController: TaskController
- columnController: ColumnController
- memberController: MemberController
+ PersistDTO(): void
+ ChangePassword(password: string): void
+ AddBoard(boardName:string, id: int): void
+ RemoveBoard(boardName:string): int
+ GetBoard(boardName:string): Board
+ GetTasks(columnOrdinal: int): List<Task>
+ IsLogged(): bool
+ Login(password: string): bool
+ Logout(): void
+ Authenticate(password: string): bool
+ GetBoardsNames(): List<string>
~ JoinBoard(Board board): void
- JoinBoardFromDTO(Board board): void
+ GetUserBoards(): List<int>
+ LeaveBoard(board: Board): void
+ AssignTask(boardName: string, columnOrdinal: int, taskID: int, emailAssignee: string): void
~ Board
- idCounter: int
- columns: List<Column>
- DEFAULT_COLUMNS: string[] <<readonly>>
- LIMITLESS_VALUE: int <<const>>
- DEFAULT_LIMIT: int <<const>>
- columns_count: int
- owner: string
- members: List<string>
- id: int
- name: string
- boardDTO: BoardDTO
- taskController: TaskController
- columnController: ColumnController
- memberController: MemberController
+ AddTask(title:string, description:string, DueDate:DateTime): void
+ LimitColumn(columnOrdinal: int, limit: int): void
+ GetColumnLimit(columnOrdinal: int): int
+ GetColumnName(columnOrdinal: int): string
+ GetColumn(columnOrdinal: int): List<Task>
+ UpdateTaskDueDate(email: string, columnOrdinal: int, taskId:int, dueDate: DateTime): void
+ UpdateTaskTitle(email: string, columnOrdinal: int, taskId:int, title: string): void
+ UpdateTaskDescription(email:string, columnOrdinal: int, taskId: int, description: string): void
+ AdvanceTask(columnOrdinal: int, taskId: int, userEmail: string): void
+ DeleteTask(columnOrdinal: int, taskId: int): void
~ AssignTask(changerEmail: string, columnOrdinal: int, taskID: int, emailAssignee: string): void
~ ChangeToUnassigned(): void
+ AddMember(email: string): void
+ deleteBoard(): void
~ Column
+ DEFAULT_COLUMNS: string[] <<readonly>>
+ tasks: Dictionary<int,Task>
+ columnName: string
+ limit: int
+ columnDTO: ColumnDTO
~ Task
- MAX_TITLE_LENGTH: int <<const>>
- MAX_DESCROPTION_LENGTH: int <<const>>
- Id: int
- boardID: int
- CreationTime: DateTime
- title: string
- description: string
- DueDate: DateTime
- assigneeEmail: string
- taskDTO: TaskDTO
+ AdvanceTaskDTO(): void
+ PersistDTO(): void
+ UnpersistDTO(): void
+ UpdateTaskDueDate(email: string, DueDate:DateTime): void
+ UpdateTaskTitle(email: string, title: string): void
+ UpdateTaskDescription(email: string, description: string): void
+ AssignTask(changerEmail: string, email: string): void
+ SetAssignToNull(): void
Data Access Layer
~ MemberDTO
- boardID: int
- email: string
+ DeleteMember(): void
+ Persist(): void
~ UserDTO
- email: string
- password: string
+ Persist(): void
+ UpdatePassword(newPassword: string): void
~ BoardDTO
- id: int
- owner: string
- name: string
+ Persist(): void
+ UpdateOwner(newOwner: string): void
+ DeleteBoard(): void
<<ABSTRACT>>
~ DTO
# controller: Controller
# IsPersisted: bool
# Insert(attributesNames: string[] attributesValues: object[]): void
# Update(identifiers: string[], identifierValues: object[], varToUpdate: string, valueToUpdate: object): void
# Delete(identifiers: string[], identifiersValues: object[]): void
+ Persist(): void
~ ColumnDTO
- boardID: int
- columnOrdinal: int
- limit: int
+ tasks: Dictionary<int,Task>
+ UpdateLimit(limit: int): void
+ Persist(): void
+ Unersist(): void
~ TaskDTO
- boardID: int
- taskID: int
- columnOrdinal: int
- creationTime: DateTime
- title: string
- description: string
- dueDate: DateTime
- assigneeEmail: string
+ UpdateDueDate(dueDate:DateTime): void
+ UpdateTitle(title: string): void
+ UpdateDescription(description: string): void
+ UpdateAssignee(assigneeEmail: string): void
+ UpdateColumnOrdinal(columnOrdinal: int): void
+ DeleteTask(): void
+ Persist(): void
+ AdvanceTask(): void
+ Unersist(): void
~ BoardController
- tableName : string <<const>>
+ Insert(attributesNames: string[] attributesValues: object[]): bool
+ Update(identifiers: string[], identifierValues: object[], varToUpdate: string, valueToUpdate: object): bool
+ Delete(identifiers: string[], identifiersValues: object[]): bool
+ ConvertReaderToObject(reader: SQLiteDataReader): BoardDTO
+ SelectAllOwnerBoards(owner: string): List<BoardDTO>
<<ABSTRACT>>
~ Controller
- connection: string <<const>>
Insert(attributesNames: string[] attributesValues: object[], tableName: string): bool
Update(identifiers: string[], identifierValues: object[], tableName: string, varToUpdate: string, valueToUpdate: object): bool
Delete(identifiers: string[], identifiersValues: object[], tableName: string): bool
DeleteAll():void
Insert(identifiersValues: object[], varToUpdate: string, valueToUpdate: object):bool
+ Update(identifiersValues: object[]):bool
+ Delete(attributesValues: object[]):bool
+ ConvertReaderToObject(reader: SQLiteDataReader):DTO
~ TaskController
- tableName : string <<const>>
+ Insert(attributesNames: string[] attributesValues: object[]): bool
+ Update(identifiers: string[], identifierValues: object[], varToUpdate: string, valueToUpdate: object): bool
+ Delete(identifiers: string[], identifiersValues: object[]): bool
+ ConvertReaderToObject(reader: SQLiteDataReader): TaskDTO
+ SelectAllTasksForBoard(boardID: int): List<TaskDTO>
~ ColumnController
- tableName : string <<const>>
+ Insert(attributesNames: string[] attributesValues: object[]): bool
+ Update(identifiers: string[], identifierValues: object[], varToUpdate: string, valueToUpdate: object): bool
+ Delete(identifiers: string[], identifiersValues: object[]): bool
+ SelectColumnsFromID(id: int): List<ColumnDTO>
+ ConvertReaderToObject(reader: SQLiteDataReader): ColumnDTO
~ UserController
- tableName : string <<const>>
+ Insert(attributesNames: string[] attributesValues: object[]): bool
+ Update(identifiers: string[], identifierValues: object[], varToUpdate: string, valueToUpdate: object): bool
+ Delete(identifiers: string[], identifiersValues: object[]): bool
+ Select(): List<UserDTO>
+ ConvertReaderToObject(reader: SQLiteDataReader): UserDTO
~ MemberController
- tableName : string <<const>>
+ Insert(attributesNames: string[] attributesValues: object[]): bool
+ Update(identifiers: string[], identifierValues: object[], varToUpdate: string, valueToUpdate: object): bool
+ Delete(identifiers: string[], identifiersValues: object[]): bool
+ ConvertReaderToObject(reader: SQLiteDataReader): MemberDTO
+ SelectMembersFromID(id: int): List<MemberDTO>