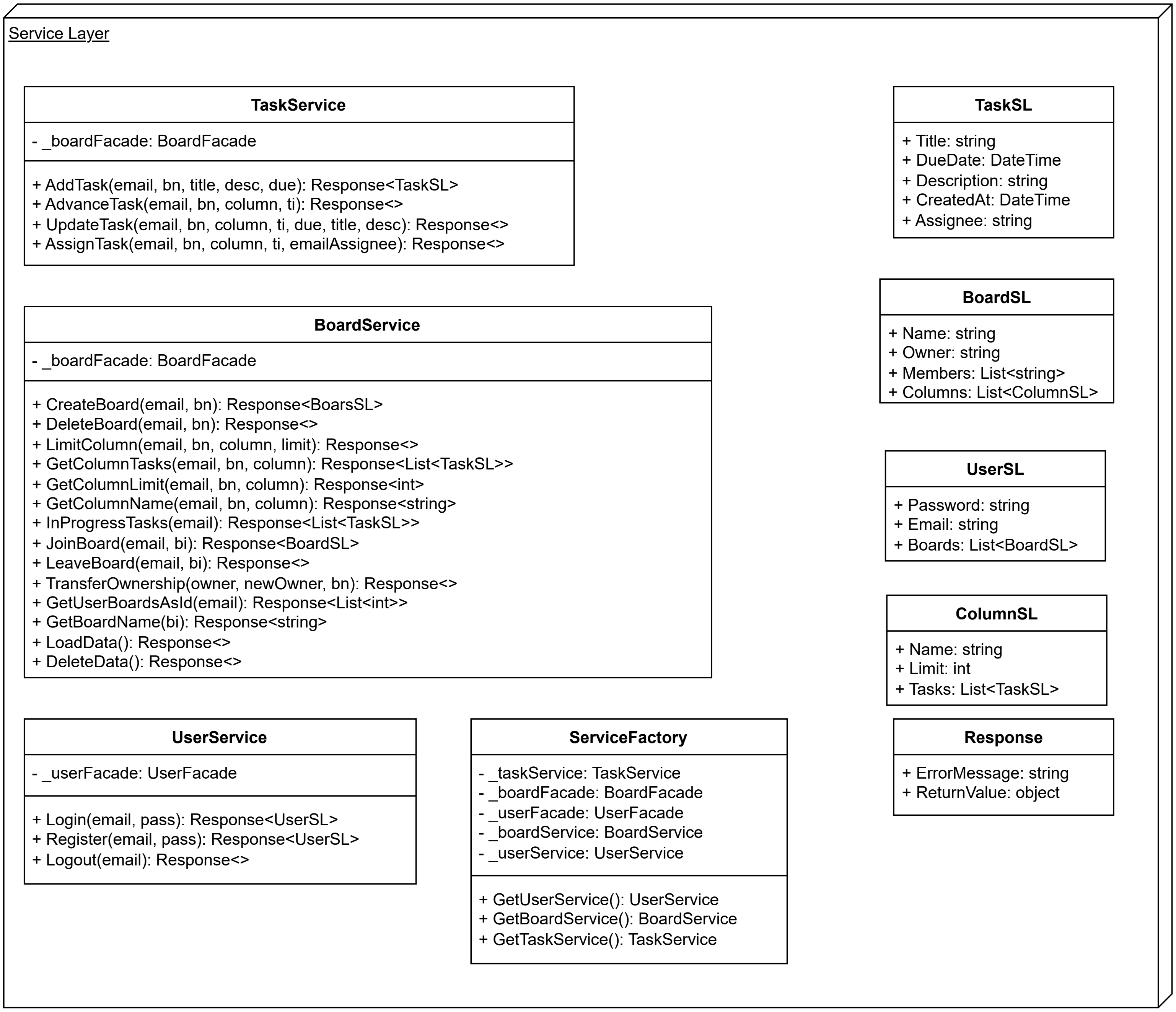


TaskService
```
- _boardFacade: BoardFacade

+ AddTask(email, bn, title, desc, due): Response<TaskSL>
+ AdvanceTask(email, bn, column, ti): Response<>
+ UpdateTask(email, bn, column, ti, due, title, desc): Response<>
+ AssignTask(email, bn, column, ti, emailAssignee): Response<>
```

BoardService
```
- _boardFacade: BoardFacade

+ CreateBoard(email, bn): Response<BoarsSL>
+ DeleteBoard(email, bn): Response<>
+ LimitColumn(email, bn, column, limit): Response<>
+ GetColumnTasks(email, bn, column): Response<List<TaskSL>>
+ GetColumnLimit(email, bn, column): Response<int>
+ GetColumnName(email, bn, column): Response<string>
+ InProgressTasks(email): Response<List<TaskSL>>
+ JoinBoard(email, bi): Response<BoardSL>
+ LeaveBoard(email, bi): Response<>
+ TransferOwnership(owner, newOwner, bn): Response<>
+ GetUserBoardsAsId(email): Response<List<int>>
+ GetBoardName(bi): Response<string>
+ LoadData(): Response<>
+ DeleteData(): Response<>
```

UserService
```
- _userFacade: UserFacade

+ Login(email, pass): Response<UserSL>
+ Register(email, pass): Response<UserSL>
+ Logout(email): Response<>
```

ServiceFactory
```
- _taskService: TaskService
- _boardFacade: BoardFacade
- _userFacade: UserFacade
- _boardService: BoardService
- _userService: UserService

+ GetUserService(): UserService
+ GetBoardService(): BoardService
+ GetTaskService(): TaskService
```

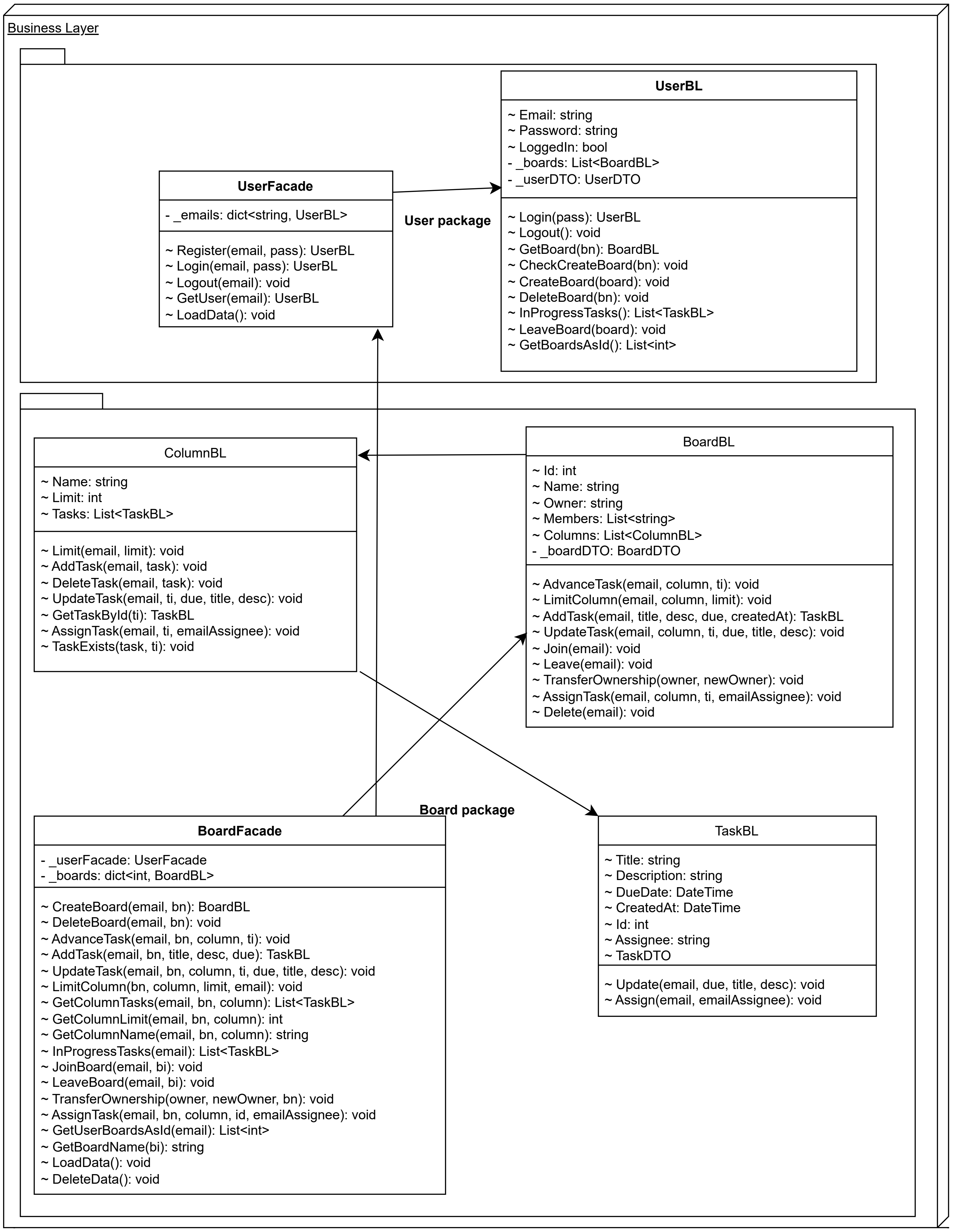
TaskSL
```
+ Title: string
+ DueDate: DateTime
+ Description: string
+ CreatedAt: DateTime
+ Assignee: string
```

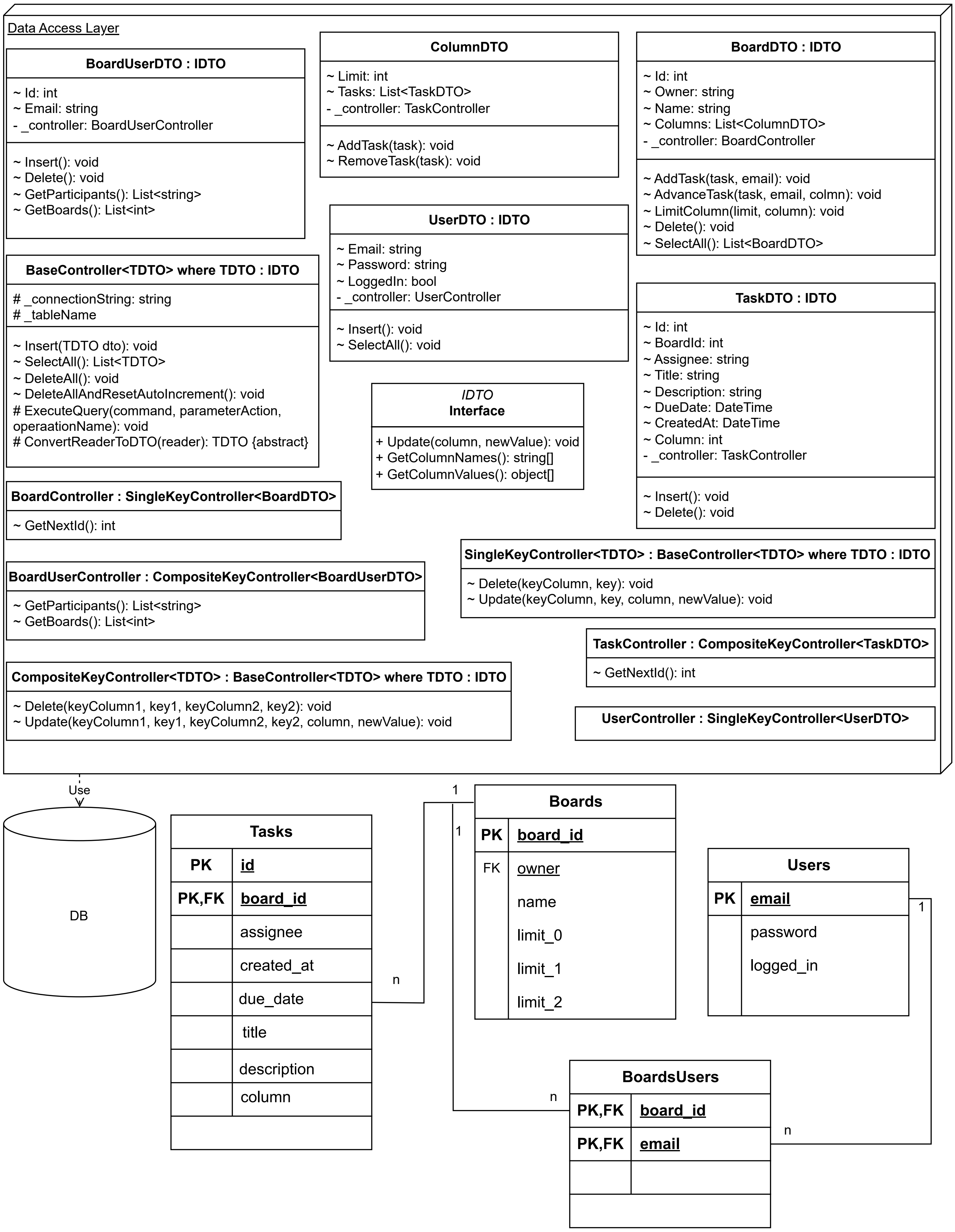
BoardSL
```
+ Name: string
+ Owner: string
+ Members: List<string>
+ Columns: List<ColumnSL>
```

UserSL
```
+ Password: string
+ Email: string
+ Boards: List<BoardSL>
```

ColumnSL
```
+ Name: string
+ Limit: int
+ Tasks: List<TaskSL>
```

Response
```
+ ErrorMessage: string
+ ReturnValue: object
```


Business Layer
User package
UserFacade
- _emails: dict<string, UserBL>
~ Register(email, pass): UserBL
~ Login(email, pass): UserBL
~ Logout(email): void
~ GetUser(email): UserBL
~ LoadData(): void
UserBL
~ Email: string
~ Password: string
~ LoggedIn: bool
- _boards: List<BoardBL>
- _userDTO: UserDTO
~ Login(pass): UserBL
~ Logout(): void
~ GetBoard(bn): BoardBL
~ CheckCreateBoard(bn): void
~ CreateBoard(board): void
~ DeleteBoard(bn): void
~ InProgressTasks(): List<TaskBL>
~ LeaveBoard(board): void
~ GetBoardsAsId(): List<int>
Board package
ColumnBL
~ Name: string
~ Limit: int
~ Tasks: List<TaskBL>
~ Limit(email, limit): void
~ AddTask(email, task): void
~ DeleteTask(email, task): void
~ UpdateTask(email, ti, due, title, desc): void
~ GetTaskById(ti): TaskBL
~ AssignTask(email, ti, emailAssignee): void
~ TaskExists(task, ti): void
BoardBL
~ Id: int
~ Name: string
~ Owner: string
~ Members: List<string>
~ Columns: List<ColumnBL>
- _boardDTO: BoardDTO
~ AdvanceTask(email, column, ti): void
~ LimitColumn(email, column, limit): void
~ AddTask(email, title, desc, due, createdAt): TaskBL
~ UpdateTask(email, column, ti, due, title, desc): void
~ Join(email): void
~ Leave(email): void
~ TransferOwnership(owner, newOwner): void
~ AssignTask(email, column, ti, emailAssignee): void
~ Delete(email): void
BoardFacade
- _userFacade: UserFacade
- _boards: dict<int, BoardBL>
~ CreateBoard(email, bn): BoardBL
~ DeleteBoard(email, bn): void
~ AdvanceTask(email, bn, column, ti): void
~ AddTask(email, bn, title, desc, due): TaskBL
~ UpdateTask(email, bn, column, ti, due, title, desc): void
~ LimitColumn(bn, column, limit, email): void
~ GetColumnTasks(email, bn, column): List<TaskBL>
~ GetColumnLimit(email, bn, column): int
~ GetColumnName(email, bn, column): string
~ InProgressTasks(email): List<TaskBL>
~ JoinBoard(email, bi): void
~ LeaveBoard(email, bi): void
~ TransferOwnership(owner, newOwner, bn): void
~ AssignTask(email, bn, column, id, emailAssignee): void
~ GetUserBoardsAsId(email): List<int>
~ GetBoardName(bi): string
~ LoadData(): void
~ DeleteData(): void
TaskBL
~ Title: string
~ Description: string
~ DueDate: DateTime
~ CreatedAt: DateTime
~ Id: int
~ Assignee: string
~ TaskDTO
~ Update(email, due, title, desc): void
~ Assign(email, emailAssignee): void


Data Access Layer
BoardUserDTO : IDTO
~ Id: int
~ Email: string
- _controller: BoardUserController
~ Insert(): void
~ Delete(): void
~ GetParticipants(): List<string>
~ GetBoards(): List<int>
ColumnDTO
~ Limit: int
~ Tasks: List<TaskDTO>
- _controller: TaskController
~ AddTask(task): void
~ RemoveTask(task): void
BoardDTO : IDTO
~ Id: int
~ Owner: string
~ Name: string
~ Columns: List<ColumnDTO>
- _controller: BoardController
~ AddTask(task, email): void
~ AdvanceTask(task, email, colmn): void
~ LimitColumn(limit, column): void
~ Delete(): void
~ SelectAll(): List<BoardDTO>
UserDTO : IDTO
~ Email: string
~ Password: string
~ LoggedIn: bool
- _controller: UserController
~ Insert(): void
~ SelectAll(): void
BaseController<TDTO> where TDTO : IDTO
# _connectionString: string
# _tableName
~ Insert(TDTO dto): void
~ SelectAll(): List<TDTO>
~ DeleteAll(): void
~ DeleteAllAndResetAutoIncrement(): void
# ExecuteQuery(command, parameterAction, operaationName): void
# ConvertReaderToDTO(reader): TDTO {abstract}
IDTO
Interface
+ Update(column, newValue): void
+ GetColumnNames(): string[]
+ GetColumnValues(): object[]
TaskDTO : IDTO
~ Id: int
~ BoardId: int
~ Assignee: string
~ Title: string
~ Description: string
~ DueDate: DateTime
~ CreatedAt: DateTime
~ Column: int
- _controller: TaskController
~ Insert(): void
~ Delete(): void
BoardController : SingleKeyController<BoardDTO>
~ GetNextId(): int
SingleKeyController<TDTO> : BaseController<TDTO> where TDTO : IDTO
~ Delete(keyColumn, key): void
~ Update(keyColumn, key, column, newValue): void
BoardUserController : CompositeKeyController<BoardUserDTO>
~ GetParticipants(): List<string>
~ GetBoards(): List<int>
TaskController : CompositeKeyController<TaskDTO>
~ GetNextId(): int
CompositeKeyController<TDTO> : BaseController<TDTO> where TDTO : IDTO
~ Delete(keyColumn1, key1, keyColumn2, key2): void
~ Update(keyColumn1, key1, keyColumn2, key2, column, newValue): void
UserController : SingleKeyController<UserDTO>
Use
DB
Tasks
PK
id
PK,FK
board_id
assignee
created_at
due_date
title
description
column
n
1
Boards
PK
board_id
FK
owner
name
limit_0
limit_1
limit_2
1
Users
PK
email
password
logged_in
1
n
BoardsUsers
PK,FK
board_id
PK,FK
email
n