Object Design Document (ODD)

MeerKat

Object Design Document

Versione 0.1.4

Coordinatore del progetto:

Nome	Matricola

Revision History

Data	Versione	Descrizione	Autore
28/11/2024	0.1	Scrittura indice.	Stefano Nicolò , Francesco Giuseppe Trotta, Giuseppe Ballacchino, Gabriel Tabasco
2/12/2024	0.1.1	Introduzione	Gabriel Tabasco
10/12/2024	0.1.2	Sezione Pacchetti	Giuseppe Ballacchino,Gabriel Tabasco
11/12/2024	0.1.3	Interfacce delle classi	Francesco Giuseppe Trotta, Stefano Nicolò Zito
14/02/20256	0.1.4	Final revision	Stefano Nicolò , Francesco Giuseppe Trotta, Giuseppe Ballacchino, Gabriel Tabasco

Indice

2.3 Organizzazione dei file

3. Interfacce delle classi

4. Conclusione

1. Introduzione

1.1 Trade-off nel design degli oggetti

- Descrizione delle decisioni principali prese nel design:
  - Memoria vs Tempo di risposta: Si punta a metodi di accesso che prioritizzino velocità di risposta rispetto alla quantità di memoria utilizzata.
  - Costruire vs Comprare: *Utilizzo di strutture dati e framework esistenti dove possibile.*

1.2 Linee guida per la documentazione delle interfacce

- Convenzioni:
  - **Nomi delle classi:** Singolare (es. `User`, `ChatRoom`).
  - **Nomi dei metodi:** Frasi verbali (es. `getNome`, `inviaMessaggio`).
  - **Eccezioni:** Restituite tramite meccanismi dedicati.

1.3 Definizioni e abbreviazioni

- **UUID:** Identificatore univoco universale.
- **Byte array:** Rappresentazione binaria di dati (es. immagini).
- **DateTime:** Tipo di dato per rappresentare date e orari.

1.4 Riferimenti

- Specifiche dei requisiti (RAD).
- Diagrammi UML correlati.

---

2. Pacchetti

2.1 Descrizione generale

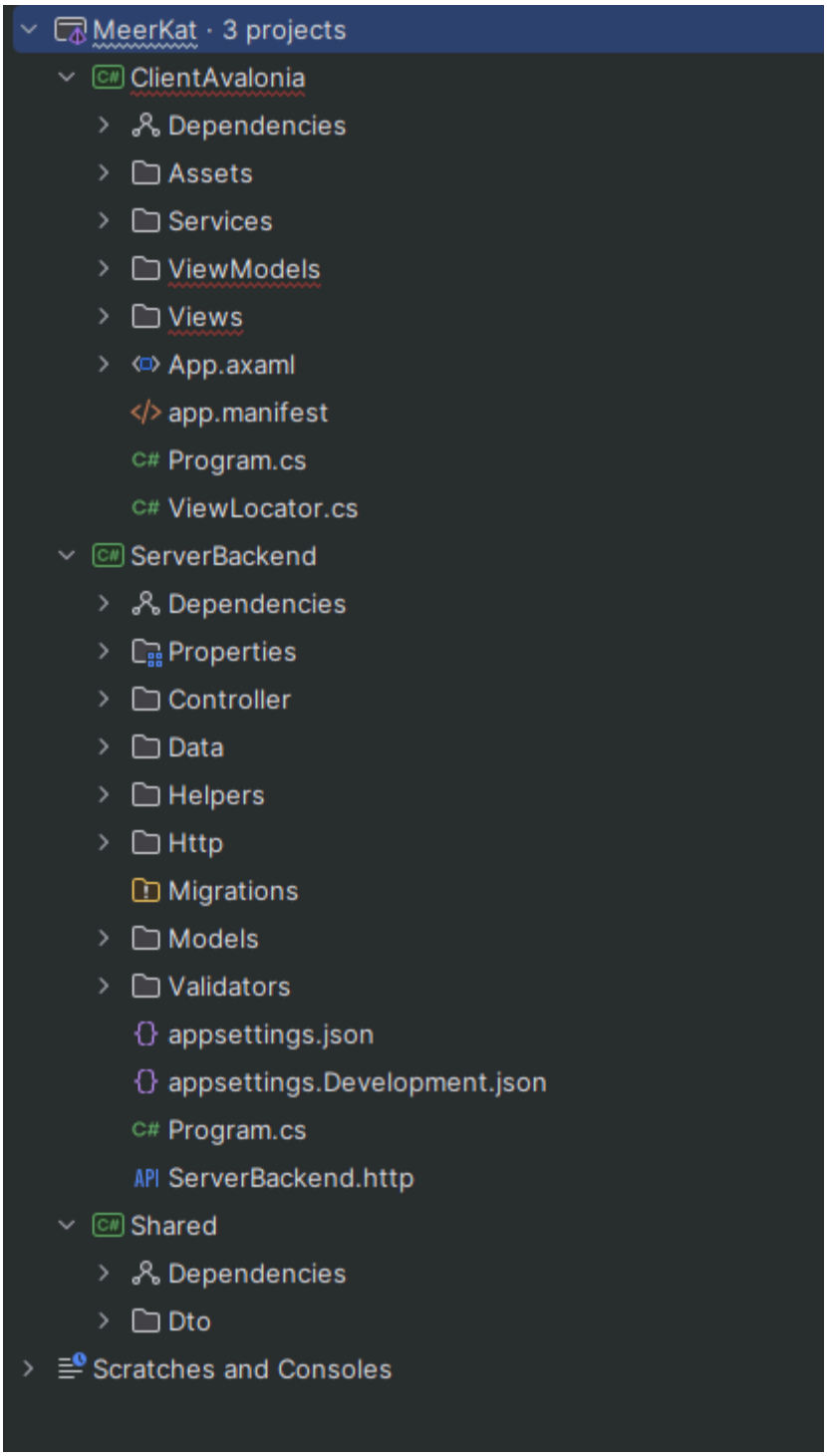
Nel progetto sono presenti tre pacchetti inerenti al pattern MVVM utilizzato per la creazione del Sistema Software. I pacchetti sono:

- **Models:** Models è l'implementazione del modello di dominio, che comprende la logica di business e di validazione (e.g. DTO).
- **ViewModels:** Intermerdiario tra **Views** e **Models**, è responsabile della gestione della logica delle interfacce grafiche, effettuando il binding dei dati gli eventi della View e i metodi del Model.
- **Views:** Responsabile della definizione della struttura, il layout e l'aspetto di ciò che l'utente vede sullo schermo.

Invece i package generali del progetto sono:

- **ClientAvalonia**
  - Componenti dell'interfaccia grafica, UI e gestione della UI
- **ServerBackend**
  - Componenti della gestione del backend del progetto
- **Shared**
  - Componenti di comunicazione con il database

2.3 Struttura dei pacchetti



3. Interfacce delle classi

3.1 Class: `UserDto`

- **Attributi:**
  - `+id: Guid`
  - `+name: String`
  - `+surname: String`
  - `+email: String`
  - `+password: String`
  - `+DateOfBirth: DateOnly`
  - `+immage: byte[]`
- **Metodi:**
  - Getter:
    - `+getName(): String`
    - `+getSurname(): String`
    - `+getEmail(): String`
    - `+getPassword(): String`
    - `+getDateofBirth(): DateOnly`
    - `+getId(): Guid`
    - `+getImmage(): byte[]`
  - Setter:
    - `+setNome(String name): void`
    - `+setCognome(String surname): void`
    - `+setEmail(String email): void`
    - `+setPassword(String password): void`
    - `+setDataNascita(DateOnly DateOfBirth): void`
    - `+setId(Guid id): void`
    - `+setImmagine(byte[] immage):void`

3.2 Class: `UserViewModel`

- **Attributi:**
  - `-user : UserDto`
- **Metodi:**
  - `+ToggleEditName() : void`
  - `+ToggleEditSurname() : void`
  - `+ToggleEditMail() : void`
  - `+ToggleEditDate() : void`
  - `+SaveEdit() : void`

3.3 Class: `UserControl`

- **Metodi:**
  - `+NestedTypes(bool nested): IQueryable<User>`
  - `+GetUsers([FromQuery] bool nested = false): Task<ActionResult<IEnumerable<UserDto>>>`
  - `+GetUserById(Guid id, [FromQuery] bool nested = false): Task<ActionResult<UserDto>>`
  - `+GetUserSelf([FromQuery] bool nested = false):  Task<ActionResult<UserDto>>`
  - `+CreateUser([FromBody] User user):  Task<ActionResult<UserDto>>`
  - `+UpdateUser( User user): Task<IActionResult>`
  - `+DeleteUser(Guid id): Task<IActionResult>`
  - `+UserLogin( LoginRequestDto request): Task<IActionResult>`

3.4 Class: TeamsViewModel

- **Attributi:**
- `userService: UserService`
- **Metodi:**
  - `TeamsViewModel(): void`
  - `LoadUserAsync(): Task`

- `getTeamList(): ObservableCollection<TeamTemplate>`
- `setTeamList(teamList: ObservableCollection<TeamTemplate>): void`
- `getTeamList2(): ObservableCollection<TeamTemplate>`
- `setTeamList2(teamList2: ObservableCollection<TeamTemplate>): void`

3.5 Class: TeamsViewModel

- **Metodi:**
  - T`eamController(meerkatContext: MeerkatContext): void`
  - `NestedTypes(nested: bool): IQueryable<Team>`
  - `GetTeams(nested: bool = false): Task<ActionResult<IEnumerable<TeamDto>>>`
  - `GetTeamById(id: Guid, nested: bool = false): Task<ActionResult<TeamDto>>`
  - `CreateTeam(team: Team): Task<ActionResult<TeamDto>>`
  - `UpdateTeam(team: Team): Task<IActionResult>`
  - `DeleteTeam(id: Guid): Task<IActionResult>`

3.6 Class: `TeamDto`

- **Attributi:**
  - `+Id : Guid`
  - `+Name: string`
  - `+Description: string`
  - `+DeadLine: DateTime`
  - `+Image = byte[]`
  - `+ManagerId: Guid`
  - `+Manager: User`
  - `+Members: ICollection<User>`
  - `+TaskList: ICollection<TaskList>`
- Metodi:

- Getter:
  - `+getDescription(): string`
  - `+getName(): string`
  - `+getDeadLine( ): DateTime`
  - `+getImage(): byte[]`
  - `+getManagerId(): Guid`
  - `+getManager(): User`
  - `+getMembers(): ICollection<User>`
  - `+getTaskList(): ICollection<TaskList>`
- Setter:
  - `+setDescription(): string`
  - `+getName(): string`
  - `+getDeadLine( ): DateTime`
  - `+getImage(): byte[]`
  - `+getManagerId(): Guid`
  - `+getImage(): byte[]`
  - `+getManager(): User`
  - `+getMembers(): ICollection<User>`
  - `+getTaskList(): ICollection<TaskList>`

3.7 Class: `TaskDto`

- Attributi:
  - `+Id: Guid`
  - `+Name: string`
  - `+Description: string`
  - `+TaskListId: Guid`

- `+TaskList: TaskList`
- `+DeadLine: DateOnly`

- Metodi:
  - Getter:
    - `+getName() : string`
    - `+getDescription(): string`
    - `+getTaskListId(): Guid`
    - `+getTaskList(): TaskList`
    - `+getDeadLine(): DateOnly`
  - Setter:
    - `+setName() : string`
    - `+setDescription(): string`
    - `+setTaskListId(): Guid`
    - `+setTaskList(): TaskList`
    - `s+etDeadLine(): DateOnly`

3.8 Class: `TaskListDto`

- **Attributi:**
  - `id: Guid`
  - `Name: String`
  - `Description: String`
  - `Tasks: ICollection<TaskDto>`
  - `TeamId: Guid`
  - `Team: TeamDto`

- **Metodi:**

- **Getter:**
  - `getName(): String`
  - `getId(): Guid`
  - `getDescription(): String`
  - `getTasks(): ICollection<TaskDto>`
  - `getTeamId(): Guid`
  - `getTeam(): TeamDto`
- **Setter:**
  - `setId(id: Guid): void`
  - `setTeam(team: TeamDto): void`
  - `setName(name: String): void`
  - `setDescription(description: String): void`
  - `setTasks(tasks: ICollection<TaskDto>): void`
  - `setTeamId(teamId: Guid): void`

3.9 Class: `Server`

Attributi:

- `+ isRunning: bool`
- `+ tcpListener: TcpListener`
- `+ clients:ConcurrentDictionary<string, TcpClient>`

Metodi:

- `+ GetServerHost(): String`
- `+ GetServerPort(): Integer`
- `+ StartServer(ipAddress: String, port: Integer): Task`
- `BroadcastMessageAsync(sender: String, message: String): Task`
- `HandleClient(client: TcpClient): Task`

- `+ StopServer(): void`

3.10 Class: `ChatClient`

Attributi:

- `_tcpClient: TcpClient`
- `_reader: StreamReader`
- `_writer: StreamWriter`
- `_mainWindow: MainWindow`
- `_users: List<User>`

Metodi:

- `+ ConnectAsync(host: String, port: Integer): Task`
- `ReadMessagesAsync(): Task`
- `+ SendMessageAsync(userName: String, message: String): Task`
- `+ AddUser(userName: User): void`
- `+ Disconnect(): void`

Class: `ChatGroup`

Attributi:

- `+Users: List<User>`
- `_chatMessages: List<MessageChat>`
- `+AdminGroup: User`
- `+GroupName: string`

Metodi:

- `+AddUser(user: User): bool`
- `+AddMessage(message: MessageChat): void`
- `+GetMessages(): IEnumerable<MessageChat>`
- `+ChangeGroupName(name: string): void`

- `+ToString(): string`
- `+GetChatUsers(): List<User>`

3.11 Class: `MessageChat`

Attributi:

- `+Sender: User`
- `+Receiver: User`
- `+Content: string`

Metodi:

- `+ MessageChat(sender: User, content: String): void`
- `+ ToString(): String`

3.12 Class: `ChatGroupViewModel`

Attributi:

- `+ Messages: ObservableCollection<MessageChat>`
- `+ CurrentGroup: ChatGroup`

Metodi:

- `+ ChatGroupViewModel(): void`
- `+ OnGroupSelected(group: ChatGroup): void`
- `+ SendMessage(content: String): void`

3.13 Class: `MainWindowViewModel`

Attributi:

- `+ GroupName: String`
- `+ SelectedGroup: ChatGroup`

Metodi:

- `# OnPropertyChanged(propertyName: String): void`
- `+ DisplayGroups: IEnumerable<ChatGroup>`

- `+ AddGroup(groupName: String): void`
- `+ AddMessage(username: String, message: String): void`
- `+ OpenGroupChatWindow(selectedGroup: ChatGroup): void`

---

4. Conclusione

- Questo documento deve essere mantenuto aggiornato durante lo sviluppo.
- Usare come riferimento durante la fase di test e integrazione.