

DIAGRAMAS DE REFERENCIA Y DIAGRAMAS GENERADOS

ID	Diagramas de Referencia (Humano)		Diagramas Generados (Herramienta)	
	Código en PlantUML	Diagrama en PlantUML	Código en PlantUML	Diagrama en PlantUML
HU-CL01	<pre>@startuml class Member { memberId : Int name : String email : String } class Profile { profileId : Int info : String tags : List<String> } Member "1" -- "1" Profile @enduml</pre>	<pre> classDiagram class Member { memberId : Int name : String email : String } class Profile { profileId : Int info : String tags : List<String> } Member "1" -- "1" Profile </pre>	<pre> @startuml class Member { } class Profile { + updateDescription() } class ProfileField { - name: String - value: String } Member "1" *-- "1" Profile Profile "1" *-- "*" ProfileField @enduml </pre>	<pre> classDiagram class Member class Profile { updateDescription() } class ProfileField { name: String value: String } Member "1" -- "1" Profile Profile "1" *-- "*" ProfileField </pre>

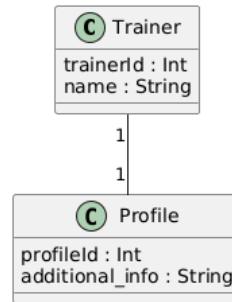
HU-CL02	<pre>@startuml class Practitioner { practitionerId : Int name : String } class Profile { profileId : Int experience : String } Practitioner "1" -- "1" Profile @enduml</pre>	<pre> classDiagram class Practitioner { practitionerId : Int name : String } class Profile { profileId : Int experience : String } Practitioner "1" .. "1" Profile </pre>	<pre>@startuml class Member { } class Practitioner { } class Profile { + addExperience(experience: Experience): void } class Experience { - details: String } Practitioner --> Member Member "1" *-- "1" Profile Profile "1" *-- "*" Experience @enduml</pre>	<pre> classDiagram class Practitioner class Member class Profile { <<addExperience(experience: Experience): void>> } class Experience { <<details: String>> } Practitioner --> Member Member "1" .. "1" Profile Profile "1" .. "*" Experience Experience "*" .. "1" Profile </pre>
---------	--	---	--	---

HU-CL03

```
@startuml
class Trainer{
    trainerId : Int
    name : String
}

class Profile {
    profileId : Int
    additional_info : String
}

Practitioner "1" -- "1" Profile
@enduml
```



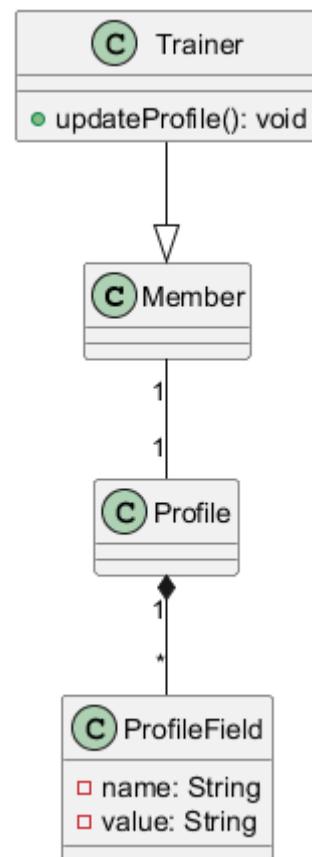
```
@startuml
class Member {
}

class Trainer {
    + updateProfile(): void
}

class Profile {
}

class ProfileField {
    - name: String
    - value: String
}

Trainer --> Member
Member "1" -- "1" Profile
Profile "1" *-- "*" ProfileField
@enduml
```



HU-CL04	<pre>@startuml class Member { memberId : Int name : String email : String } class Profile { profileId : Int summary : String isPrivate : Boolean } class ProfileDirectory Member "1" -- "1" Profile : owns > ProfileDirectory ..> Profile : lists > @enduml</pre>	<pre> classDiagram class Member { memberId : Int name : String email : String } class Profile { profileId : Int summary : String isPrivate : Boolean } class ProfileDirectory Member "1" -- "1" Profile : owns ProfileDirectory ..> Profile : lists @enduml </pre> <p>The diagram shows three classes: Member, Profile, and ProfileDirectory. Member has attributes memberId, name, and email. Profile has attributes profileId, summary, and isPrivate. ProfileDirectory is associated with Profile via a dashed line labeled 'lists'. Member is associated with Profile via a solid line labeled 'owns'.</p>	<pre>@startuml class Member { + viewProfile(profile: Profile) + requestConnection(member: Member) } class Profile {} class Connection {} Member "1" -- "1" Profile (Member, Member) .. Connection @enduml</pre> <p>The code defines Member, Profile, and Connection classes. Member has methods viewProfile and requestConnection. Profile and Connection are empty classes. A note indicates a self-loop association between Member and Profile labeled '1'.</p>	<pre> classDiagram class Member { + viewProfile(profile: Profile) + requestConnection(member: Member) } class Profile {} class Connection {} Member "1" -- "1" Profile Member ..> Connection @enduml </pre> <p>This diagram is identical to the one in the code section, showing Member, Profile, and Connection classes and their associations.</p>
HU-CL05	<pre>@startuml class Member { memberId : Int name : String email : String } class Profile { profileId : Int name : String summary : String isPrivate : Boolean } class SearchCriteria { name : String tag : String location : String } class ProfileSearchService Member "1" -- "1" Profile : owns</pre>	<pre> classDiagram class Member { memberId : Int name : String email : String } class Profile { profileId : Int name : String summary : String isPrivate : Boolean } class SearchCriteria { name : String tag : String location : String } class ProfileSearchService Member "1" -- "1" Profile : owns Profile ..> SearchCriteria : returns matches ProfileSearchService ..> SearchCriteria : uses @enduml </pre> <p>The diagram shows five classes: Member, Profile, SearchCriteria, ProfileSearchService, and a fourth unnamed class. Member has attributes memberId, name, and email. Profile has attributes profileId, name, summary, and isPrivate. SearchCriteria has attributes name, tag, and location. ProfileSearchService is associated with SearchCriteria via a dashed line labeled 'uses'. Member is associated with Profile via a solid line labeled 'owns'. Profile is associated with SearchCriteria via a dashed line labeled 'returns matches'.</p>	<pre>@startuml class Member { + searchForProfiles(filter: SearchFilter): Profile[] + connectWith(member: Member) } class Profile {} class ProfileField { - name: String - value: String } class SearchFilter {} class Connection {} Member "1" -- "1" Profile Profile "1" -- "*" ProfileField @enduml</pre> <p>The code defines Member, Profile, ProfileField, SearchFilter, and Connection classes. Member has methods searchForProfiles and connectWith. Profile and Connection are empty classes. ProfileField is a private class with attributes name and value. SearchFilter is an empty class. A note indicates a self-loop association between Profile and ProfileField labeled '1'.</p>	<pre> classDiagram class Member { + searchForProfiles(filter: SearchFilter): Profile[] + connectWith(member: Member) } class Profile {} class ProfileField { - name: String - value: String } class SearchFilter {} Member "1" -- "1" Profile Member ..> SearchFilter Profile ..> ProfileField : returns matches ProfileField ..> SearchFilter : uses Profile ..> Connection @enduml </pre> <p>This diagram is identical to the one in the code section, showing Member, Profile, ProfileField, SearchFilter, and Connection classes and their associations.</p>

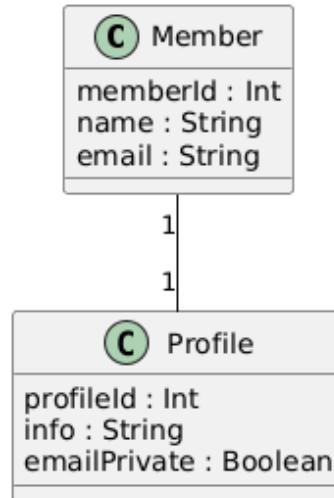
	> ProfileSearchService ..> SearchCriteria : uses > ProfileSearchService ..> Profile : returns matches @enduml		Member ..> SearchFilter Member ..> Profile (Member, Member) .. Connection @enduml	
HU-CL06	@startuml class Member { memberId : Int name : String email : String } class Profile { profileId : Int info: String isPrivate : Boolean } Member "1" -- "1" Profile @enduml	<pre> classDiagram class Member { memberId : Int name : String email : String } class Profile { profileId : Int info: String isPrivate : Boolean } Member "1" .. "1" Profile @enduml </pre>	@startuml class Member { } class Profile { + markAsPrivate(): void } class PrivacySetting { - status: String } Member "1" -- "1" Profile Profile "1" -- "1" PrivacySetting @enduml	<pre> classDiagram class Member class Profile { ● markAsPrivate(): void } class PrivacySetting { □ status: String } Member "1" .. "1" Profile Profile "1" .. "1" PrivacySetting @enduml </pre>

HU-CL07

```
@startuml
class Member {
    memberId : Int
    name : String
    email : String
}
```

```
class Profile {
    profileId : Int
    info : String
    emailPrivate : Boolean
}
```

```
Member "1" -- "1" Profile
@enduml
```



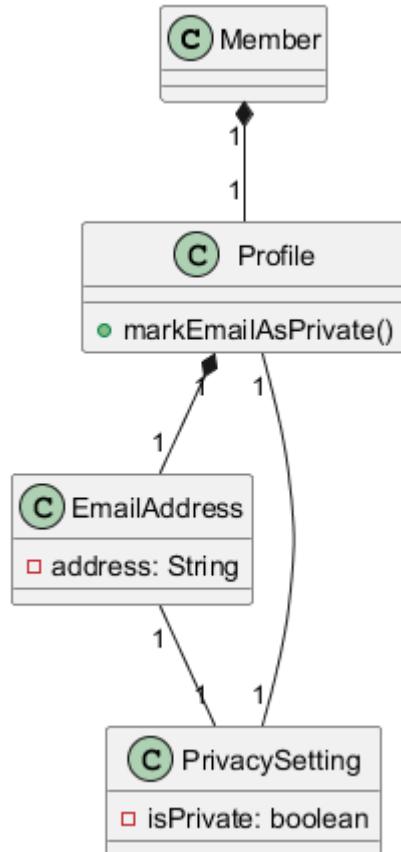
```
@startuml
class Member {
}

class Profile {
    + markEmailAsPrivate()
}

class EmailAddress {
    - address: String
}

class PrivacySetting {
    - isPrivate: boolean
}

Member "1" *-- "1" Profile
Profile "1" *-- "1" EmailAddress
Profile "1" -- "1" PrivacySetting
EmailAddress "1" -- "1" PrivacySetting
@enduml
```



HU-CL08	<pre>@startuml class Trainer { trainerId : Int name : String email : String } class Profile { profileId : Int info : String } class Course { courseId : Int title : String detailsURL : String } Trainer "1" -- "1" Profile Trainer "1" -- "0..*" Course @enduml</pre>	<pre> classDiagram class Trainer { trainerId : Int name : String email : String } class Profile { profileId : Int info : String } class Course { courseId : Int title : String detailsURL : String } Trainer "1" -- "1" Profile Trainer "1" -- "0..*" Course </pre>	<pre>@startuml class Member { } class Trainer { + listCourseInProfile(course: Course): void } class Profile { } class Course { } class CoursePage { - detailedDescription: String } Trainer --> Member Trainer "1" -- "1" Profile Profile "1" -- "*" Course Course "1" -- "1" CoursePage @enduml</pre>	<pre> classDiagram class Trainer { <<listCourseInProfile(course: Course): void>> } class Member { } class Profile { } class Course { } class CoursePage { <<detailedDescription: String>> } Trainer "1" -- "1" Member Trainer "1" -- "1" Profile Profile "1" -- "*" Course Course "1" -- "1" CoursePage </pre>
HU-CL09	<pre>@startuml class CertificationCourse { courseId : Int title : String startDate : Date isOpen : Boolean } class CourseCatalog { CourseCatalog ..> CertificationCourse : lists > } @enduml</pre>	<pre> classDiagram class CourseCatalog { <<lists>> } class CertificationCourse { courseId : Int title : String startDate : Date isOpen : Boolean } CourseCatalog ..> CertificationCourse : lists > </pre>	<pre>@startuml class Visitor { + viewUpcomingCertificationCourses(): List } class Course { - name: String - description: String - schedule: String } class CertificationCourse { } CertificationCourse --> Course Visitor ..> CertificationCourse @enduml</pre>	<pre> classDiagram class Visitor { <<viewUpcomingCertificationCourses(): List>> } class CertificationCourse { } class Course { <<name: String, description: String, schedule: String>> } CertificationCourse --> Course Visitor ..> CertificationCourse </pre>

HU-CL10	<pre>@startuml class Event { eventId : Int name : String eventDate : Date location : String } class EventCatalog EventCatalog ..> Event : lists > @enduml</pre>	<pre> classDiagram class EventCatalog { +Event[] lists } class Event { -eventId : Int -name : String -eventDate : Date -location : String } EventCatalog "1" -- "n" Event : lists </pre>	<pre>@startuml class Visitor { + viewUpcomingEvents() } class Event { - name: String - description: String - date: Date } Visitor ..> Event @enduml</pre>	<pre> classDiagram class Visitor { + viewUpcomingEvents() } class Event { - name: String - description: String - date: Date } Visitor ..> Event : viewUpcomingEvents() </pre>
---------	--	--	--	--