SUBSCRIPTIONS/ **DOMAIN**

Subscriptions(DTO)

- + EmployerEmail: String [1] + ProjectName: String [1] + SubscriptionName: String [1] + ProviderName: String [1]
- + SubscriptionDescription: String [1] + Cost: int [1] + TypeSubscription: int + IsEnabled: int [1]
- + SubscriptionDTO(String, String, String, String, String, int, int, int): void

Subscription

- + EmployerEmail: String [1]
- + Employeremail: String [1] + ProjectName: String [1] + SubscriptionName: String [1] + ProviderName: String [1] + SubscriptionDescription: String [1] + Cost: int [1]
- + TypeSubscription: int [1]
- + IsEnabled: int [1] + SubscriptionString, String, String, String, String, int, int, int): void

SubscriptionRepository

- GetAllDeductionsAsync(): Task<IEnumerable<Subscription>> + GetAllBenefictsAsync(): Task<IEnumerable<Subscription>>
- +CreateSubscriptionAsync(Subscription): Task + GetSubscription(string, string): Task<Subscription>?
 + GetDeductionsByProject(string, string): Task<IEnumerable<Subscription>>
 +GetBenefitsByProject(string, string): Task<IEnumerable<Subscription>>

EMPLOYEES /

Employe(DTO)

+ Email: String [1] + EmployeeDTO(String): void

Employee + Email: String [1]

+ Employee(String): void IEmployeeRepository

+ CreateEmployeeAsync(String): Task + GetEmployeeByEmail(string): Task<IEnumerable<Employee>>

ACCOUNTS /

DOMAIN

Account(DTO) + Email: String [1] + UserPassword: String [1] + AccountsDTO(string, string): void

> Account + Email: String [1] + UserPassword: byte[] [1] + IsAuthenticated : bool + Account(String, byte[]): void

IAccountRepository

+InsertAccountData(AccountsDTO): Task + CheckEmail(AccountsDTO): Task<|Enumerable<Account>>
+ CheckPassword(AccountsDTO): Task<|Enumerable<Account>>
+ CheckPassword(AccountsDTO): Task<|Enumerable<Account>>
+ CetAuthenticationState(AccountsDTO, byte): Task< +SendEmail(string, string): void

EMPLOYER /

EmployerDTO + Email: String [1] + EmployersDTO(String): void

Employer + Email:string [1]

+ Employer(String): void

IEmployerRepository

+CreateEmployerAsync(String): Task + GetEmployerAsync(String): Task<Employer>?

PEOPLE / DOMAIN

- Person(DTO) + Email: String [1]
- +Name: String [1] + LastName1: String [1]
- + LastName1: String [1] + LastName2: String [1] + Ssn: int [1] +BankAccount: String [1] +Adress: String [1] +PhoneNumber: String [1..*]
- + PersonDTO(String, String, String, String, int, String, String , String): void

Person

- + Email: String [1] +Name: String [1] + LastName1: String [1] + LastName2: String [1]
- + Ssn: int [1] +BankAccount: String [1] +Adress: String [1] +PhoneNumber: String [1..*]
- + Person(String, String, String, String, int, String, String , String):

IPersontRepository

- + GetProjectEmployees(string): Task<IEnumerable<Person>

+ CreatePersonAsync(Person): Task + GetAllEmployees(): Task<IEnumerable<Person>>

IProjectRepository

GetAllProjectsAsync(): Task<|Enumerable<Project>> CreateProjectAsync(Project): Task GetAllNameProjects(string): Task<|Enumerable<Project>> GetEmployerProyects(string): Task<IEnumerable<Project>> + GetPersonByEmail(string): Task<IEnumerable<Person>> + GetAllInfoEmployer(Person): Task<IEnumerable<Person>> GetProject(string, string): Task<Project>

- Agreements(DTO) + EmployeeEmail: String [1] + EmployeerEmail: String [1]
- + ProjectName: string [1] + ContractDate: string [1] + ContractType: string [1] + MountPerHour: int [1] + Duration: string [1]

+ AgreementDTO(String, string, string, string, string, int, string): void

Agreement

- + EmployeeEmail: String [1] + EmployeerEmail: String [1] + ProjectName: string [1] + ContractDate: string [1]
- + ContractType: string [1] + MountPerHour: int [1] + Duration: string [1]
- + Agreement(String, string, string, string, int, string): void

IAgreementRepository

CreateAgreementAsync(Agreement): Task

DISEÑO DE CAPAS PARA UNA ENTIDAD GENÉRICA EN EL PROYECTO (para cada entidad se cumple el mismo formato a la hora de agregar un nuevo servicio)

PROJECT/ DOMAIN

Project(DTO)

+ EmployerEmail: String [1]

+ ProjectName: String [1]

+ ProjectDescription: String [1] + MaximumBenefitAmount: int [1] + MaximumAmountForBenefits: int [1] + PaymentInterval: String [1]

+ ProjectDTO(String, String, String, int,

Project

+ EmployerEmail: String [1] + ProjectName: String [1] + ProjectDescription: String [1] + MaximumBenefitAmount: int [1]

+ MaximumAmountForBenefits: int [1]

+ PaymentInterval: String [1]

+ Project(String, String, String, int, int, String): void

