

Core

Member/ Provider and Service are all simple Data classes whose single responsibility is to manage

data related to its object. Member **Provider** + name :: Name + name :: Name + address :: Address + address :: Address + ID :: Const Int + ID :: Const Int + Inspection Methods + Inspection Methods **Transaction** + provider :: Provider + member :: Member + service :: Service + service_time :: DateTime + system_time :: DateTime + comments :: String + Inspection Methods Reporter + static DB :: DB + create_report(Transaction) :: Bool record.

Database

+ DB Connector

Service

+ code :: Const Int

+ cost :: Const Int

+ description :: String

+ Inspection Methods

Transaction manages data related to service transaction. It should not be modifiable after creation. Thus, the only functions should be a smart constructor and inspection methods

TODO: Fully define Reporter responsibility

Reporter should compile reports from a transaction record by fetching the member/providers previous transactions from the DB and appending the new transaction to the end of it. This should create a new member/provider_report object that is then passed to the DB so it can update the

TODO: Fully defined database interface

Database will provide a interface to serialize/deserialize member and provider report objects into a **Utility**

DateTime

+ first

+ last :

+ Sma

+ Insp

+ day :: Day (Day, Month, Year, etc. are t

+ month :: Month

+ year :: Year

+ hour :: Hour

+ minute :: minute

+ second :: second

+ DateTime(Day, Month, Year) :: DateTi

+ DateTime(Day, Month, Year, Hour, Mir

+ Inspection Methods

Address

+ street :: String

+ city :: String

+ state :: String

+ zip_code :: String

+ Smart Constructor

+ Inspection Methods

TODO, consider type wrappers for street, city, state, etc.

ype wrappers around an int)

ne

nute, Second) :: DateTime

Name

:: String

: String

rt Constructor

ection Methods

+ get_record(ID) :: Member / Provider
+ update_record(Member_Report)
+ update_record(Provider_Report)