Developing Summary

* Regarding the database:
  + The Id Number is unique and can be used as primary key but I decided to add incremental id(int) as primary key because the performance of inner join with integer is much faster then with string and Id Number has to be string since it can start with zero.
  + Because Id Number(Mispar Teudat Zehut) can’t be repeted, I made its field in database as unique field to prevent entering duplicated Id Numbers
  + I added two tables-ContractType and PackageType, to save contract type names and package type name separately.
* Back end implementation:
  + I decided to use three different repositories(CustomerRepository, ContractRepository and PackageRepository) in order to follow “Single Responsibility” principle(each class should be responsible only for one thing/domain)
  + Caching:
    - I implemented here In-Memory Caching
    - I wanted to avoid code duplication so I built generic method for checking the cache and if the data exists, this method returns the data from cache, otherwise it gets the data from method which comes as parameter and save it to the cache.
    - When put method of CustomerController is called(for updating the customer’s address), the cache should be cleared because cache value is not relevant anymore.
  + I used DTO concept(data transfer object) so I can combine several entities in one object while returning data from web api.
  + I created generic Error Handler to catch exceptions. I would also log those exceptions if it was a real product.
  + For implementing identification that valid for 5 minutes I used JWT token, so after 5 minutes it becomes invalid and the user should loging again.
  + Regarding the caching:
    - I implemented here In-Memory Caching
    - I wanted to avoid code duplication so I built generic method for checking the cache and if the data exists, this method returns the data from cache, otherwise it gets the data from method which comes as parameter and save it to the cache.
    - When put method of CustomerController is called(for updating the customer’s address), the cache should be cleared because cache value is not relevant anymore.
* Front end implementation:
  + I used mat-table from angular material to display contract and package data
  + The “Update Address” button is disabled by default, it becomes enabled after one of the address’s fields is changed. If the change is undone, the button “Update Address” becomes disable again, because there is no reason to enable address update if it has same address’s values.
  + I built navigation url mechanism for convenient use