**Archetecture Documentation**

* **Business**:
  + *Contains all the managers used to call the repositories.*
* **Data**:
  + *All Data related repositories, models, unit of work and database context.*
  + **Configurations**:
    - Fluent API configuration classes.
  + **Core**:
    - All interfaces and abstract classes used in repositories and unit of work.
  + **Entities**:
    - The EF models.
  + **Repositories**:
    - The concrete repositories used by the managers.
* **API + UI:**
  + *The implementation of the Web API back end and angular front end.*
  + **API:**
    - Controllers, SignalR Hubs, Data Generators.
  + **UI:**
    - Angular application using the SignalR libraries and consuming the API end points.
* **Notes:**
* I used the SignalR approach to avoid unnecessary database hits.
* I used the repository design pattern to separate the database related actions from the application business.
* I used the unit of work design pattern to unify the source of all operations done to the database in one place.
* I used EF code first approach because it gives me the advantage of versioning my database and rolling back to any migration I need.