# High level requirement analysis

## Functional requirements

We need to create following applications:

1. Web portal to upload and manage a list of medical journals, to find and subscribe to journals
2. Desktop client to list and read subscribed journals
3. WebAPI which will be consumed by desktop client and portal.

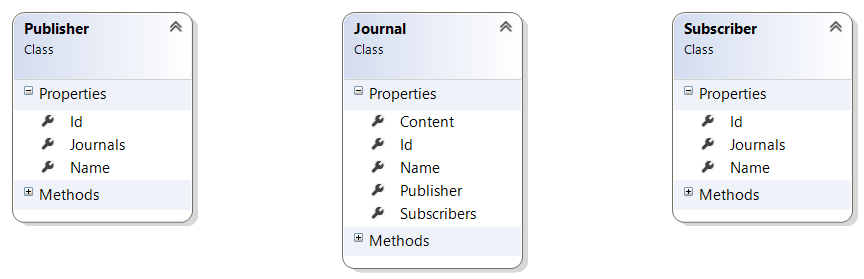
## Security requirements

Desktop client should not allow copying journal's content in any common way

* Not from web/requests/responses
* Not from their file system
* Not from client user interface (copy to clipboard, screenshot, etc)

## Technical requirements

# High level presentation of the data model



Publisher and Subscriber entity has Name and Journals properties. Since Publisher is the owner of journal he has one-to-many relationships with Journal. Subscriber and Journal have many-to-many relationship.

# Architecture

## System



Taking into account functional requirements I decided to use N-tier architecture approach. Each tier and layers of it has it single responsibility. A well-defined contracts allow abstracting from implementation details such as business logic or persistence.

Public user or publisher interacts with WebPortal through web browser. WebPortal in its turn send request to RESTfull WepAPI.

Public users read journals in Desktop client. Desktop client in its turn send request to RESTfull WepAPI too.

## Design patterns

### Enterprise Application Architecture

* Domain model – I defined business entities like Journal, Publisher, Subscriber
* Data Mapper – I used them to map domain objects to database in order to keep domain clean
* Repository – I defined collection-like interface for accessing domain objects from storage
* Service Layer – I encapsulated business logic in services
* MVVM pattern for windows-based applications. Usually it is used by default in WPF application
* Model View Controller – comes as part of asp.net mvc
* Data Transfer Object – comes as part of web api

### Gang of Four

* Factory – see CreateClient in SubscriberServiceProxy

## Multitier



Technologies:

* ASP.NET MVC + Bootstrap – presentation tier of the Web Portal
* WPF + Prism – presentation tier of the Desktop Client
* ASP.NET WepAPI – RESTfull service
* MS SQL Server + Entity framework

## Package diagram



## Deployment



Steps to deploy system on production

* Create fail-safe cluster for MS SQL
* Create network load balancing cluster
* Deplow WepPortal on NLB