

## Icon Recruitment Test

### Framework and Tools

.NET Core 3.1: <https://dotnet.microsoft.com/download/dotnet/3.1>

Autofac: <https://www.nuget.org/packages/Autofac/>

Bootstrap 5.0.1: <https://www.nuget.org/packages/bootstrap/>

log4net: [NuGet Gallery](#) | [log4net 2.0.12](#)

MSTest.TestFramework: <https://www.nuget.org/packages/MSTest.TestFramework/>

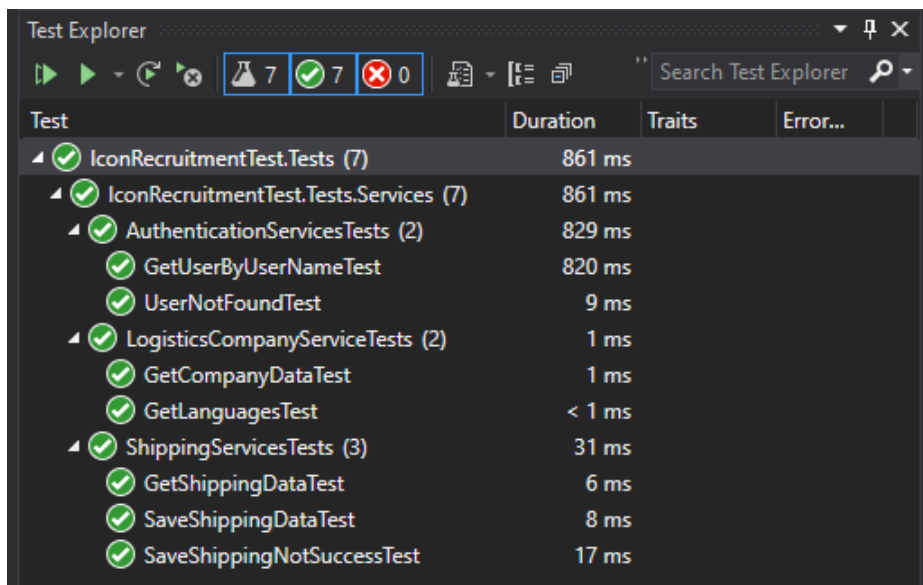
Solution: IconRecruitmentTest

Details regarding directories in the attached solution:

- IconRecruitmentTest.Common
  - Common -> Implements common functions across application (enumeration class)
  - Extensions -> Method extensions for getting description of enumeration
  - Logging-> Implemented log4net Interface methods to log errors across application (Implemented factory method for logging errors based on project)
  - Message-> Used to pass parameters from service to JavaScript.
  - Models-> Specific data and application logic and data model for creating tables
  - Resources-> GlobalStrings for creating multiple languages resources
- IconRecruitmentTest. Data
  - Data -> Populating user table with an initial set of data.
  - IconDbContext->Specified collection of all entities that can be queried from the database
  - Migrations->History of database creating/updating table etc.
- IconRecruitmentTest. Services
  - Authentication -> Interfaces for implementing methods to verify user
  - LogisticsCompany-> Interfaces for implementing methods to get suppliers, supported languages
  - Shipping-> Interfaces for implementing methods to save shipping data and to get shipping data for dashboard analysis.
  - Translate->Created a singleton implementation for getting resources for specific language.

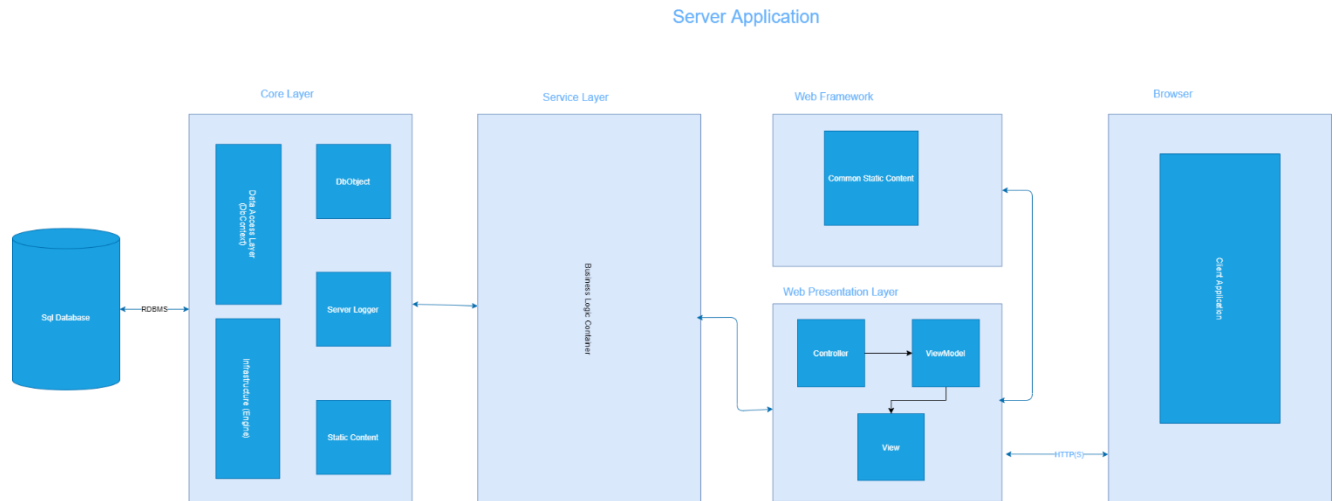
- BaseService-> Base Service to implement base methods for logging interface
- IconRecruitmentTest.Web
  - wwwroot-> Implemented custom style (**css/site.css**) and custom JavaScript function (**js/App**). Create Sprite for images and access across application with css class.
  - App\_Data-> Created a folder from configuration of log4net to store LOG files based on configuration.
  - Common-> Implemented common function for web solutions
  - Controllers-> Implements action result for handling request.
  - Extensions->Methods extension for HtmlContent and Json.
  - Infrastructure->Dependency registrations of services with Autofac, loads data in application startup.
  - ViewModel-> Created view model to use in view and have a separate logic from model
  - Views->Created views for frontend layout (Account,Dashboards,Login,Home,Languages)
  - Appsettings.json->Configuration file for application, added configuration as connection strings, settings for validation and prices based on suppliers.
  - log4net.config -> Configuration file for logs, patterns for creating and naming log file.
- IconRecruitmentTest.Tests
  - Base-> Implements a Base class methods for initializing Interface
  - Configuration-> Configuration used in testing
  - Extensions-> Assert Extensions for objects
  - Helper-> Configures and Initializes classes
  - Services-> Contains all tests for methods developed in Application

I have created unit test methods only for services as below (but we can create also for controllers)



Test	Duration	Traits	Error...
IconRecruitmentTest.Tests (7)	861 ms		
IconRecruitmentTest.Tests.Services (7)	861 ms		
AuthenticationServicesTests (2)	829 ms		
GetUserByUserNameTest	820 ms		
UserNotFoundTest	9 ms		
LogisticsCompanyServiceTests (2)	1 ms		
GetCompanyDataTest	1 ms		
GetLanguagesTest	< 1 ms		
ShippingServicesTests (3)	31 ms		
GetShippingDataTest	6 ms		
SaveShippingDataTest	8 ms		
SaveShippingNotSuccessTest	17 ms		

## Server Application Diagram:



As database I have created 2 simple tables as below:

1. **Users** -> To store created users that have permission to login and see dashboards or other information.
2. **ShippingData** -> To store orders for calculated shipping

ShippingData	
Id	
width	
height	
depth	
weight	
totalVolume	
totalPrice	
CreationTime	
LastModificationTime	
companyType	

Users	
Id	
Username	
Password	
Email	
IsEnabled	
CreationTime	
LastModificationTime	

### The task:

Design and develop a web application that has the following minimum features:

1. A customer can input the dimensions of the package (width, height, depth) as well as the weight.
2. The price for shipping the package is calculated based on dimensions and weight. Calculations are based on the logic described in the table above.
3. A customer can input the dimensions of the package (width, height, depth) as well as the weight.
4. The details of the package are captured by the web application so it can be analysed later.
5. The package price is displayed to the user. The user is then encouraged to make the order

- Identification and suggestion of fixes for any requirement shortcomings
  - Any parcel > 30Kg costs €43.99 plus €0.41 for every Kg over 25Kg (I believe that here is a misleading information and In code I have calculated for 1kg over)
  - **ShipFaster supplier**  
Validation: Any parcel >= 10Kg  
Price: Any parcel > 10Kg and <= 20Kg costs €16.99  
I believe that in order to work correct: We can change either the validation ore price.  
In code I have considered: Any parcel > =10Kg and <= 20Kg costs €16.99 in order to pass validation and get price.

**Note:** Before starting the application please make sure you have changed the ConnectionStrings located at appsettings.json .

At Package Manage Console please create the database with following commands: As Default project choose **Libraries\IconRecruitmentTest.Data** since migration and DbContext is created at this project.

- **Enable-Migrations** (if migration are not enabled)


- **Update-Database**

```
Package Manager Console
Package source: All | Default project: Libraries\IconRecruitmentTest.Data
Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages. Some packages may include dependencies which are governed by additional licenses. Follow the package source (feed) URL to determine any dependencies.
Package Manager Console Host Version 5.9.1.0
Type 'get-help NuGet' to see all available NuGet commands.
PM> Enable-Migrations
Enable-Migrations is obsolete. Use Add-Migration to start using Migrations.
PM> Update-Database
Build started...
Build succeeded.
No migrations were applied. The database is already up to date.
Done.
PM>
```

When user opens the main page, this is what he will see. By default, Cargo4You will be selected and user can put the values for the other attributes.

Both inputs for dimensions and weight have an info (personalized by supplier type) box appended to them where user can see additional info about each of them while hovering the mouse.

By default, the 'Send shipping' button is disabled until the data for Volume and Weight are populated correctly.

 Home

LoginEnglish ▾

Supplier

Cargo4You ▾

**Set the dimensions:** ⓘ

Width:0

Height0

Depth0

**Set the weight:** ⓘ

Weight:0

**Total calculated:**

Volume: 0 Cm3

Weight: 0 Kg

Cost 0 €

Send shipping

When user puts the right data on the fields, Cost is automatically updated.

Supplier

Cargo4You

**Set the dimensions:** ⓘ

Width:10

Height10

Depth10

**Set the weight:** ⓘ

Weight:10

**Total calculated:**

Volume: 1000 Cm3

Weight: 10 Kg

Cost 18 €

Send shipping

The 'Send shipping' button is now enabled and once clicked; the input data is saved in Database. A small popup is displayed with a success message. UI sample:

If user wants to calculate the data for another supplier, when he changes the supplier dropdown choice, the data below won't reset, so that it is easier for the user to distinguish which one is the best choice for them. Sample below:

Supplier

ShipFaster ▾

**Set the dimensions:** ⓘ

Width: 10

Height: 10

Depth: 10

**Set the weight:** ⓘ

Weight: 10

**Total calculated:**

Volume: 1000 Cm3

Weight: 10 Kg

Cost 11.99 €

Send shipping

For every invalid input, there is a warning icon displayed, which when hovering there, user will get more details on the issue. Cost is set to €0 when data is not valid.

Home Login English ▾

Supplier

Cargo4You ▾

**Set the dimensions:** ⓘ

Width: 12

Height: 12

Depth: 40

**Set the weight:** ⓘ

Weight: 23 ⚠

**Total calculated:**

Volume: 0 Cm3

Weight: 12 Kg

Cost 0 €


Send shipping

**Validation**

⚠ Total volume should be less than 2000cm3  
Calculated width x height x depth

On the top right of the page, there is a login page where users who are saved in database can put their credentials and login(Default user : Username->**Admin** , Password->**Admin123** ).

Login page:

 Home

Login English ▾

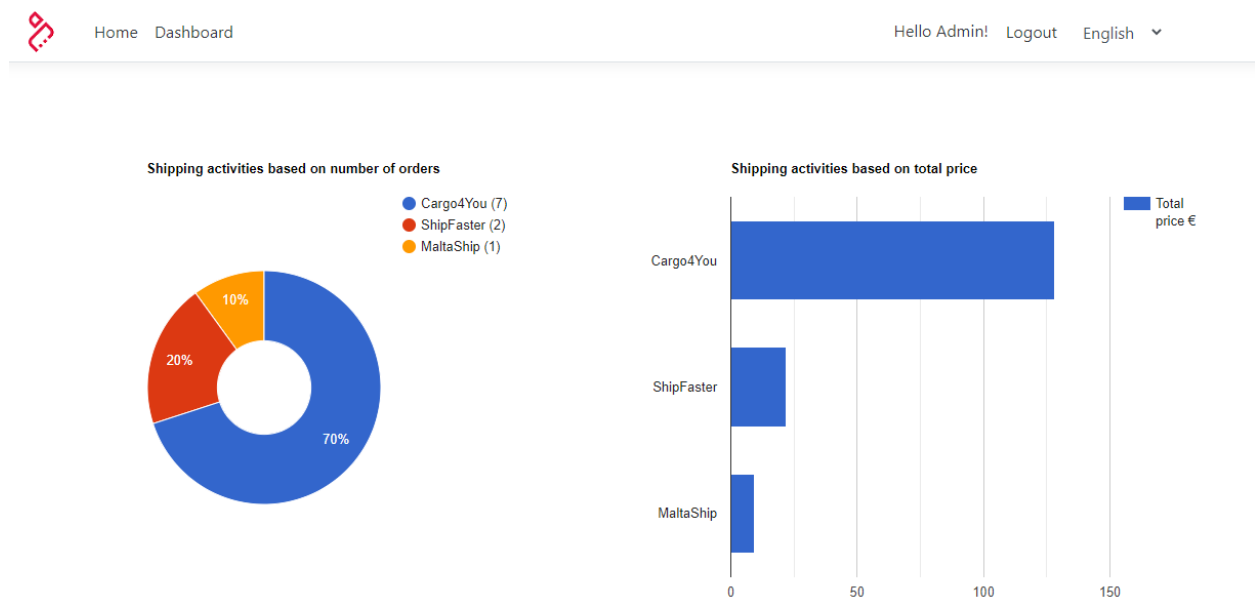
Use a local account to log in.

Username

Password

Login

There is another tab on the top left of the page named 'Dashboard' which has two pre-built dashboards that analyses data which are sent for shipping. All the data is stored in database and dashboard uses them. This page is only visible for Admin panel, so by default it will be hidden for non-logged in uses.



This page can be customized further to display more dashboards based on business needs.

There is also a possibility to change the language, for now the default language is English, but is user switches to e.g. Italian, every content in the page will be translated in Italian.





Fornitore

Cargo4You

**Imposta le dimensioni:**

Larghezza:

0

Altezza

0

Profondità

0

**Imposta il peso:**

Il peso:

0

**Totale calcolato:**

Volume: 0 Cm3

Il peso: 0 Kg

Costo 0 €

[Invia spedizione](#)