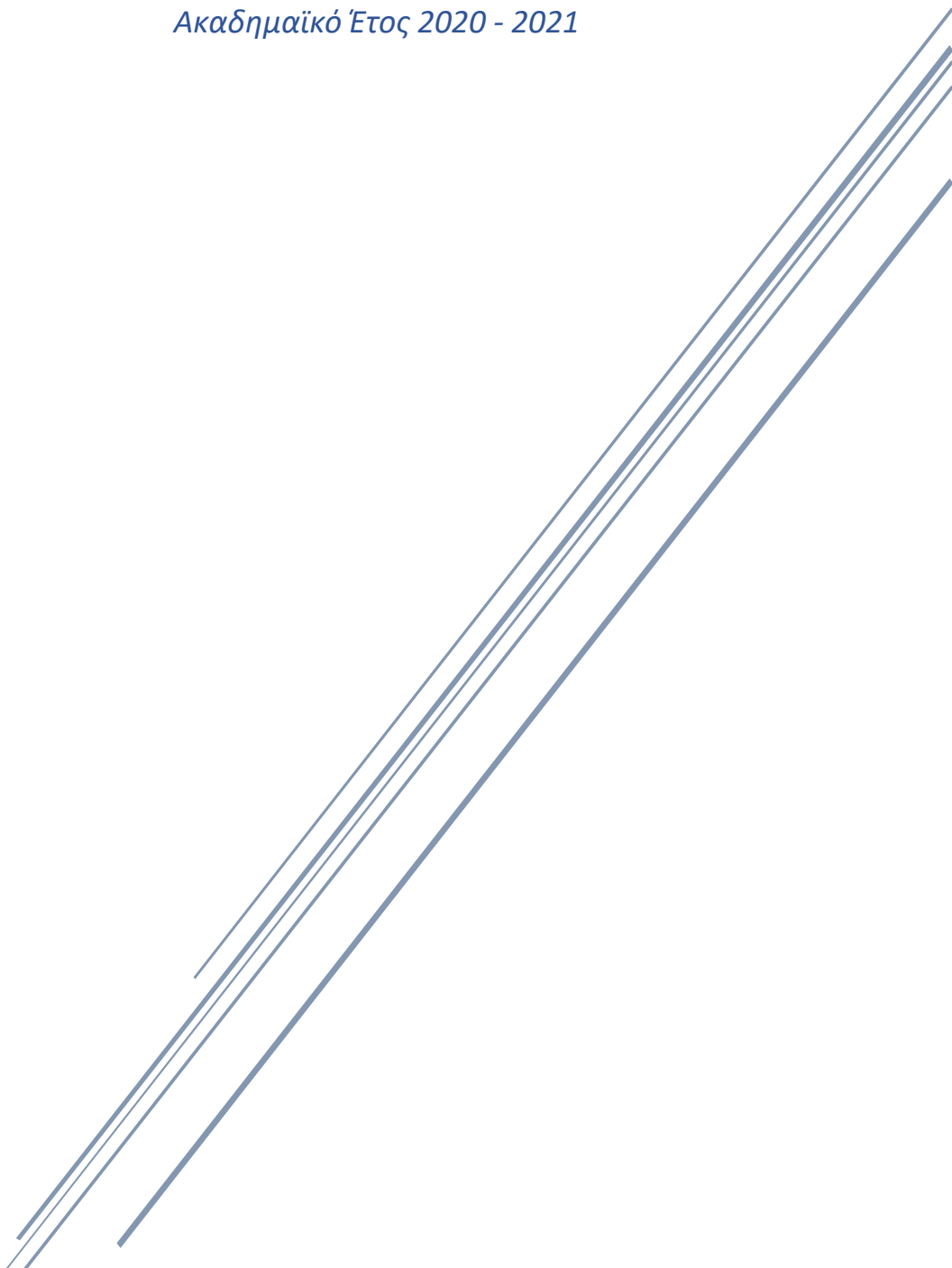


# ΣΥΓΧΡΟΝΑ ΘΕΜΑΤΑ ΤΕΧΝΟΛΟΓΙΑΣ ΛΟΓΙΣΜΙΚΟΥ ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΩΝ ΓΙΑ ΚΙΝΗΤΕΣ ΣΥΣΚΕΥΕΣ

Τεχνικό Εγχειρίδιο

*Ακαδημαϊκό Έτος 2020 - 2021*



ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ  
**UNIVERSITY OF PIRAEUS**

Παναγιώτης Αποστολόπουλος, Π17007  
Δημήτρης Ματσαγγάνης, Π17068

## Περιεχόμενα

Εισαγωγή Εργασίας .....	3
Κεντρική Ιδέα Υλοποίησης .....	7
Microsoft SQL Management Studio .....	8
Visual Studio – ASP. NET WEB Application (.NET Framework).....	8
Ενημερώσεις συστήματος .....	13
Δημιουργία Controllers & Views .....	14
Δημιουργία Αναφορών .....	19
LAYOUT.CSHTML.....	21
METADATA .....	22
Μορφοποίηση .....	24
Βιβλιογραφία.....	25

# Εισαγωγή Εργασίας

## 3<sup>η</sup> Εργασία 2020

ΜΕΓΙΣΤΗ ΒΑΘΜΟΛΟΓΙΑ: 2 ΜΟΝΑΔΕΣ

### Θέμα:

Μια αλυσίδα βιβλιοπωλείων χρησιμοποιεί τη παρακάτω Βάση Δεδομένων σε Access για την οργάνωση των δεδομένων της.

Στα πλαίσια της μηχανογραφικής αναβάθμισης της, χρειάζεται την ανάπτυξη μιας δικτυακής εφαρμογής για την καλύτερη αποτελεσματικότητα στην καταχώρηση και στη διαχείριση των δεδομένων της.

Να αναπτύξετε μια διαδικτυακή εφαρμογή σε Visual C# (ASP.NET Web Application) χρησιμοποιώντας την αρχιτεκτονική σχεδίασης Model View Controller στο περιβάλλον ανάπτυξης Visual Studio. Η εφαρμογή θα πρέπει να υλοποιεί τη παρακάτω λειτουργικότητα.

### Διαχείριση των δεδομένων

Τα δεδομένα της εταιρείας θα πρέπει να διαχειρίζονται πλήρως μέσα από τις αντίστοιχες φόρμες.

1. Η εφαρμογή θα πρέπει να διαχειρίζεται πλήρως (εισαγωγή, διαγραφή, επεξεργασία) τα δεδομένα όλων των πινάκων.

### Δημιουργία αναφορών

Πέρα από το διαχειριστικό κομμάτι, μέσω της εφαρμογής θα πρέπει να υπάρχει μια ενότητα με αναφορές, όπου ο χρήστης μπορεί να πάρει πληροφορίες σχετικά με τα δεδομένα της εφαρμογής. Παραδείγματα αναφορών:

- Ανέκτησε τα στοιχεία επικοινωνίας των συγγραφέων που τα βιβλία τους είναι ανάμεσα σε Χ πρώτα σε πωλήσεις για συγκεκριμένο χρονικό διάστημα. Ο χρήστης θα πρέπει να ορίζει τα εξής κριτήρια για την δημιουργία της αναφοράς: Αριθμός Χ, Χρονικό διάστημα Ημερομηνία Από – Έως.

- Δημιούργησε μια αναφορά που θα δείχνει το Id της παραγγελίας (OrderID), το κατάστημα στο οποίο έγινε η παραγγελία και ο τίτλος του βιβλίου που σχετίζεται με την αντίστοιχη παραγγελία. Εμφάνισε εκείνες τις παραγγελίες που έγιναν (ordered) εντός ενός ημερολογιακού διαστήματος. Ο χρήστης θα πρέπει να ορίζει τα εξής κριτήρια για την δημιουργία της αναφοράς: Χρονικό διάστημα παραγγελιών Ημερομηνία Από - Έως, Επωνυμία καταστήματος Από - Έως (χρησιμοποιώντας τον τελεστή LIKE).

Προσοχή: Σε κάθε φόρμα αναφοράς που θα δημιουργήσετε, ο χρήστης μπορεί να αφήσει κενό ένα κριτήριο. Τα κενά κριτήρια δεν λαμβάνονται υπόψιν στα ερωτήματα που θα χρειαστεί να κατασκευάσετε για την κατάρτιση των αναφορών.

Για το GUI της εφαρμογής να χρησιμοποιήσετε κλάσεις του Bootstrap και να δώσετε έμφαση στη καλή εμφάνιση και διεπαφή με τον χρήστη.

Στα παραδοτέα της εργασίας θα πρέπει να συμπεριλάβετε ένα Τεχνικό Εγχειρίδιο όπου θα αναλύονται με παραδείγματα ο τρόπος χρήσης των αντικειμένων της Βάσης Δεδομένων από το μοντέλο MVC (views, models, controllers).

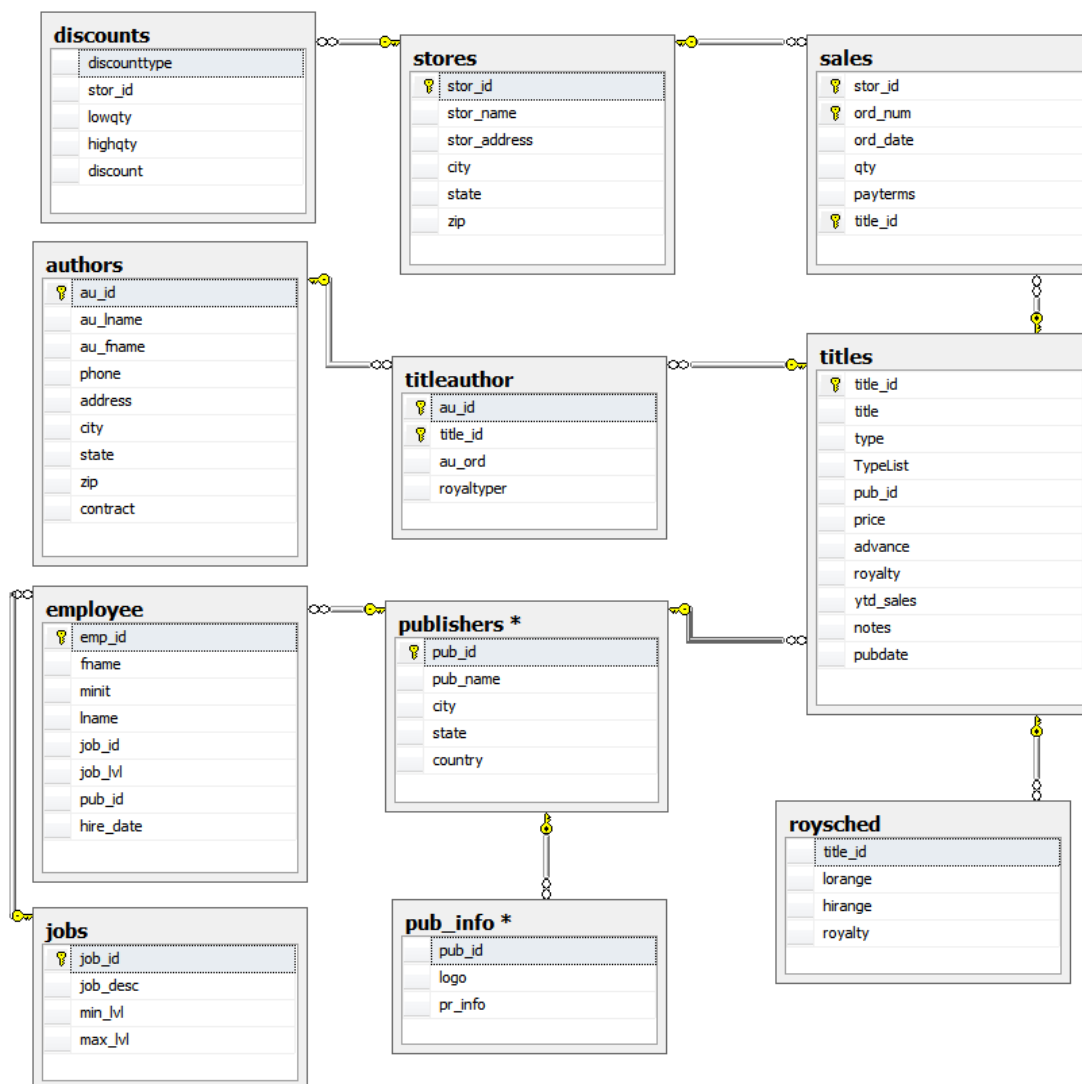
Επίσης θα πρέπει να συμπεριλάβετε ένα Εγχειρίδιο Χρήστη όπου θα παρουσιάζετε την εφαρμογή και τις λειτουργίες της με παραδείγματα και screenshots από την εφαρμογή.

#### Εργαλεία που θα χρειαστείτε για την υλοποίηση της εργασίας

- Visual Studio 2015+ (Asp.NET framework)
- Microsoft SQL Server 2012+

Μέσω του SQL SERVER Management Studio, εκτελέστε το instpubs.sql (που βρίσκεται στο φάκελο της εργασίας) για να δημιουργηθεί η Βάση Δεδομένων Pubs.

Το διάγραμμα της Βάσης Δεδομένων είναι το παρακάτω:



Διάγραμμα Βάσης 1

### Παρατηρήσεις

- Η εργασία είναι δυο ή τριών ατόμων το πολύ.
- Θα πρέπει να παραδοθεί ένα αρχείο AM1-AM2-AM3.zip (AMx ο αριθμός μητρώου), το οποίο θα περιέχει τα αρχεία της εφαρμογής, PDFs αρχεία με τεχνικό εγχειρίδιο & εγχειρίδιο χρήστη.
- Το zip αρχείο να υποβληθεί μέσω eclass (ανεβάστε το αρχείο στην περιοχή «Εργασίες». Εναλλακτικά μπορείτε να ανεβάσετε την εργασία σε dropbox και να υποβάλλετε τον σύνδεσμο που βρίσκεται το zip αρχείο.



- Η εργασία πρέπει να παραδοθεί μέχρι τις 03 Φεβρουαρίου 2021 (μετά την παρέλευση της ημερομηνίας αυτής δεν πρόκειται να γίνουν δεκτές εργασίες).
- Για οποιαδήποτε ερώτηση – απορία στείλε email στο [cman@unipi.gr](mailto:cman@unipi.gr)  
Χρήστος Μανουσόπουλος.

## Κεντρική Ιδέα Υλοποίησης

Η διαδικτυακή εφαρμογή που υλοποιήθηκε, σκοπό έχει την – όσο το δυνατό – πλησιέστερη απεικόνιση μιας αλυσίδας βιβλιοπωλείων, η οποία διαθέτει την τεχνογνωσία και τις υπηρεσίες που ζητήθηκαν από την παραπάνω εκφώνηση – και στα πλαίσια της μηχανογραφικής αναβάθμισης του βιβλιοπωλείου.

Η εφαρμογή σχεδιάστηκε με κύριους κατευθυντήριους άξονες την ευχρηστία της εφαρμογής ακόμα και από έναν άπειρο χρήστη και την συνδεσιμότητα όλων των παραθύρων της μεταξύ τους (HTML σελίδων).

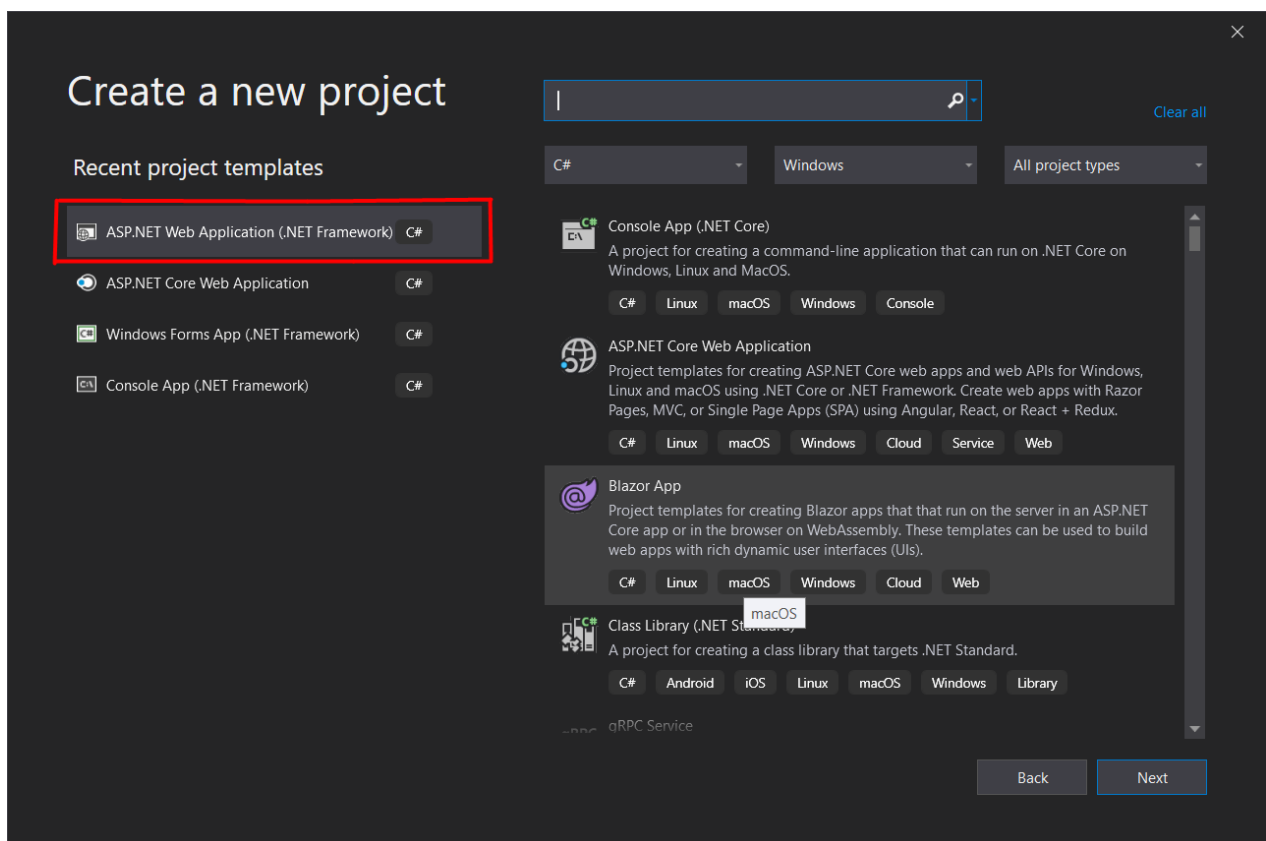
Τέλος, δόθηκε η αρμόζουσα έμφαση στην οργάνωση των δεδομένων της αλυσίδας καθώς και στη πιστή απεικόνιση ενός συστήματος Βάσης Δεδομένων και την αλληλεπίδραση αυτών με τον χρήστη – πελάτη και τις ιστοσελίδες (μέσω partial-view).

## Microsoft SQL Management Studio

Αρχικά, για να μπορέσουμε να διαχειριστούμε τα δεδομένα θα πρέπει να «τρέχουμε» το αρχείο instrubs.sql που μας είχε δοθεί. Έτσι, δημιουργείται η βάση δεδομένων rubs της οποίας το διάγραμμα απεικονίζεται στην εκφώνηση. Βάση του συγκεκριμένου διαγράμματος βασίζεται όλο το project, όπου σε συνδυασμό με το entity framework αυτό υλοποιείται.

## Visual Studio – ASP. NET WEB Application (.NET Framework)

Πρώτο βήμα είναι η δημιουργία ενός καινούργιου project, όπως φαίνεται στο παρακάτω στιγμιότυπο:

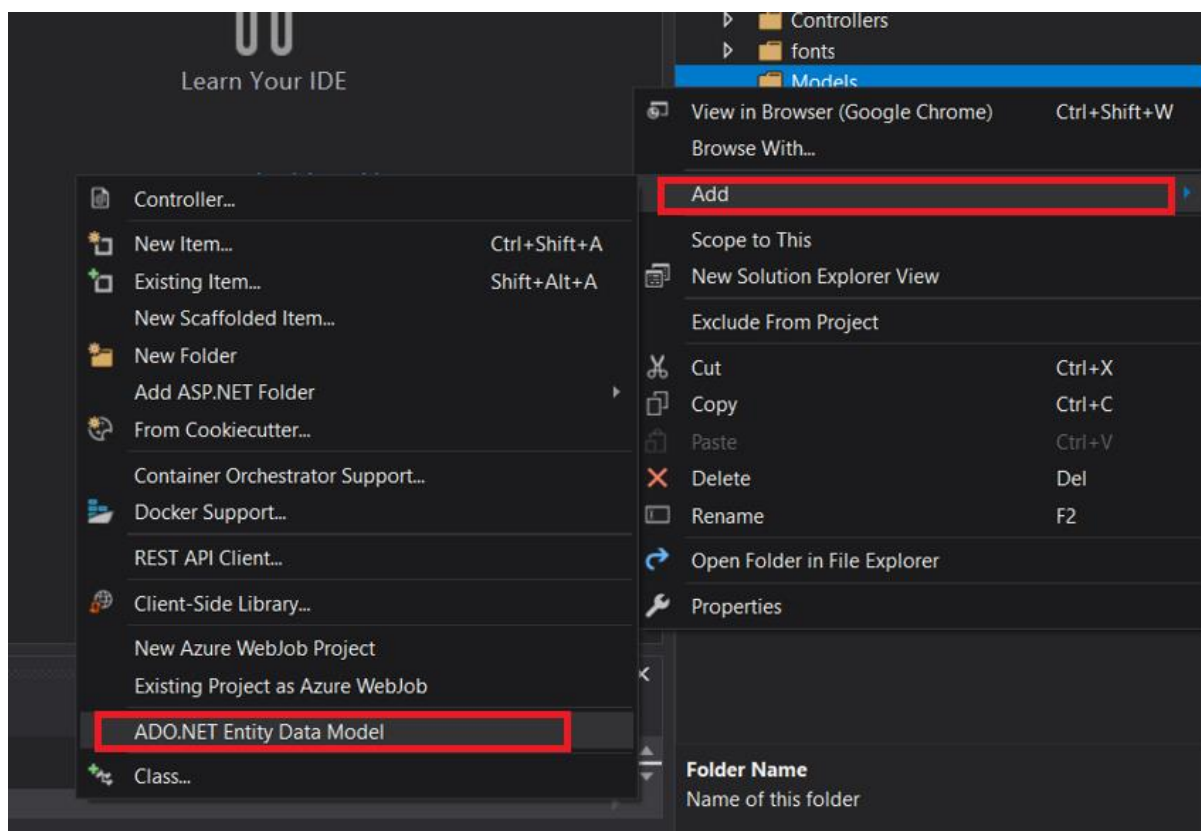


Εικόνα 1 – Δημιουργία project (ASP. NET Web Application)

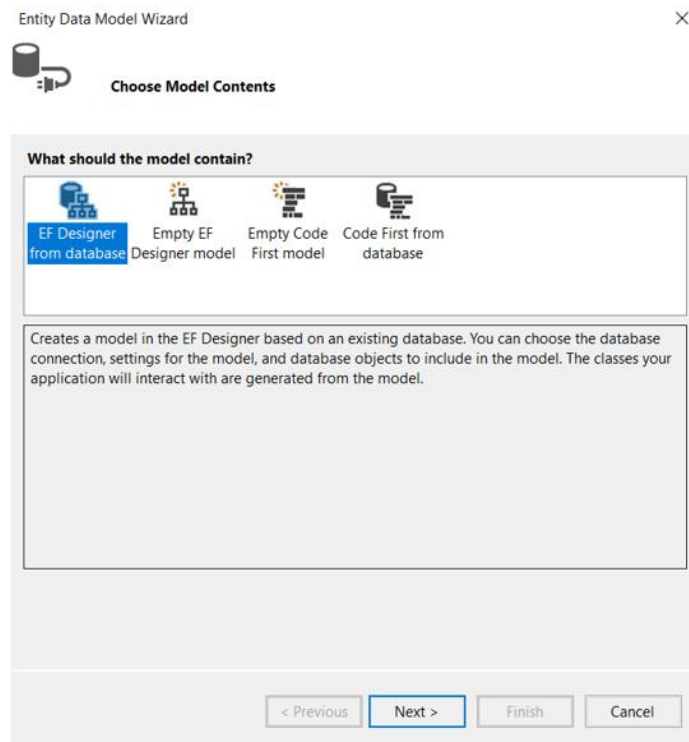


Στη συνέχεια, θα πρέπει να εισάγουμε ένα μοντέλο το οποίο θα προκύψει από την δοθείσα βάση. Να σημειωθεί, ότι μοντέλο είναι το σύνολο των κλάσεων που διαχειρίζονται το σύνολο των δεδομένων από τη βάση. Για να εργαστούμε πάνω σε αυτές τις κλάσεις του μοντέλου θα χρησιμοποιηθεί η τεχνολογία *.NET Framework*, γνωστή και ως **Entity Framework**.

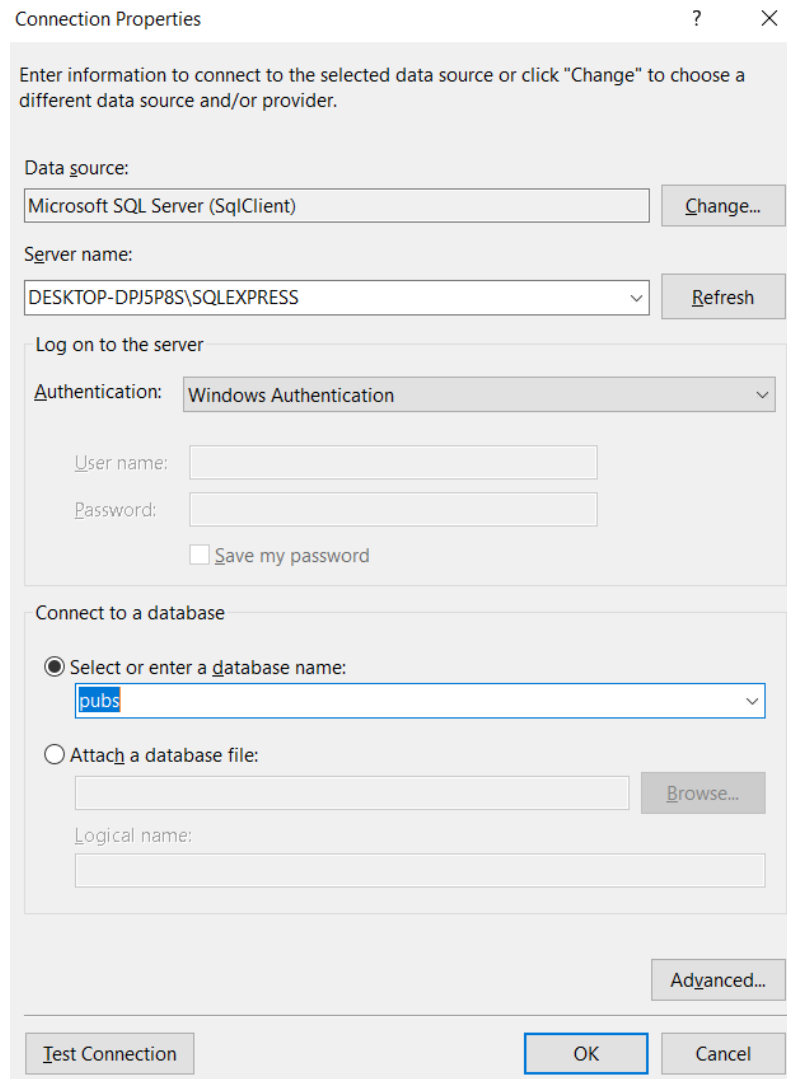
Για την επιτυχή εισαγωγή του μοντέλου θα πρέπει να ακολουθηθεί η παρακάτω αλληλουχία βημάτων, όπως αυτή αποτυπώνεται στις παρακάτω εικόνες.



Εικόνα 2 - Εισαγωγή μοντέλου




Εικόνα 3 - Διαδικασία εισαγωγής μοντέλου






Εικόνα 4 – Διαδικασία εισαγωγής μοντέλου

Στο συγκεκριμένο βήμα (**Εικόνα 4**) δημιουργείται το connection string, ώστε να επιτευχθεί η σύνδεση με τη βάση δεδομένων.

Entity Data Model Wizard ×

 **Choose Your Database Objects and Settings**

**Which database objects do you want to include in your model?**

- > ☒  **Tables**
- > ☐  Views
- > ☐  Stored Procedures and Functions

☐ Pluralize or singularize generated object names

☒ Include foreign key columns in the model

☒ Import selected stored procedures and functions into the entity model

Model Namespace:

< Previous Next > **Finish** Cancel

Εικόνα 5 – Διαδικασία εισαγωγής μοντέλου

Ακολουθώντας, όλα τα παραπάνω βήματα «συνδέουμε» τη βάση με το project και δημιουργείται το model με όνομα «**BookStore.edmx**»

## Ενημερώσεις συστήματος

Στην εφαρμογή που υλοποιήσαμε έχουμε χρησιμοποιήσει και responsive web design, το οποίο επιτυγχάνεται με την χρήση κώδικα και κλάσεων από το Bootstrap. Με τον συγκεκριμένο όρο, εννοούμε ότι όλα τα γραφικά, όπως οι εικόνες, το κείμενο, τα κουμπιά και το navbar, προσαρμόζονται ανάλογα με το μέγεθος της οθόνης και με το resize που μπορεί να κάνει ο χρήστης. Η συγκεκριμένη λειτουργία αποτυπώνεται με πιο παραστατικό τρόπο στο εγχειρίδιο χρήσης, όπου παρατίθενται screenshots της εφαρμογής τόσο από υπολογιστή, όσο και από κινητό, με τη βοήθεια του inspect στο Google Chrome .

Ωστόσο, για να επιτευχθούν όλα τα παραπάνω απαραίτητη προϋπόθεση ήταν η εκτέλεση όλων των διαθέσιμων updates για το Entity Framework, για το jQuery, αλλά και του Bootstrap από την έκδοση 3 στη 4, προκειμένου να αξιοποιηθούν οι λειτουργίες για το responsive web design.

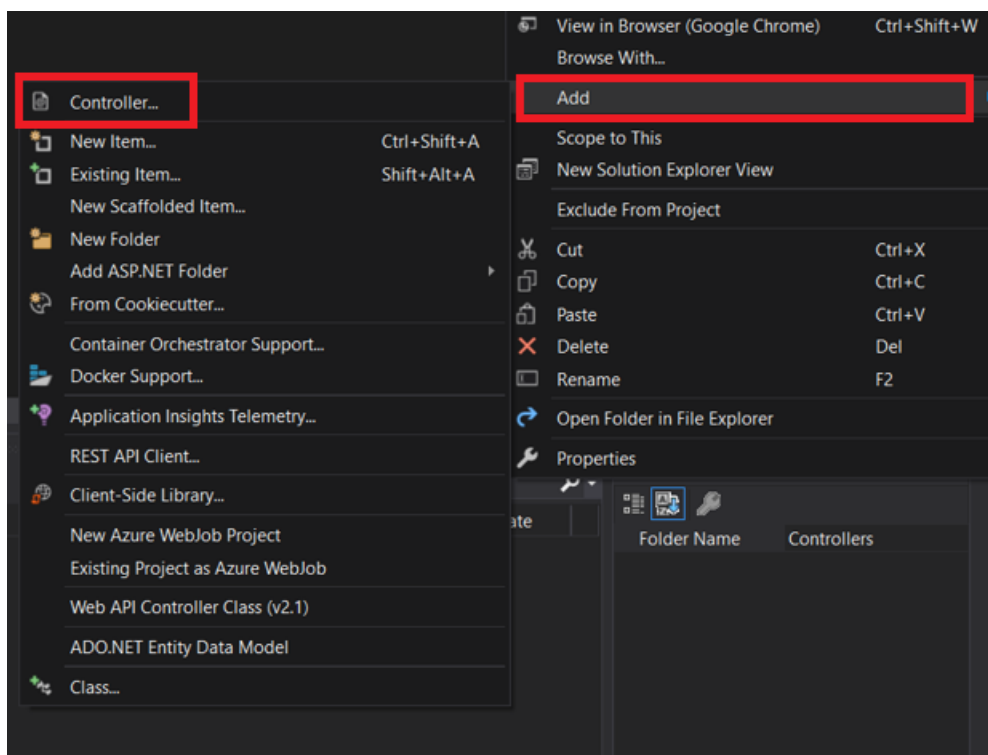
## Δημιουργία Controllers & Views

Αρχικά, **Controller** είναι μια κλάση η οποία διαχειρίζεται τα requests από τον browser, δέχεται δεδομένα από το model και στη συνέχεια καθορίζει συγκεκριμένα view templates τα οποία επιστρέφουν μια «απάντηση» στον browser. Αξιοσημείωτο είναι ότι ο controller στέλνει εντολές στο μοντέλο και να ενημερώνει την κατάσταση του μοντέλου

Τα **Views** είναι templates τα οποία διαχειρίζεται η εφαρμογή μας δυναμικά, ώστε να δημιουργεί HTML responses.

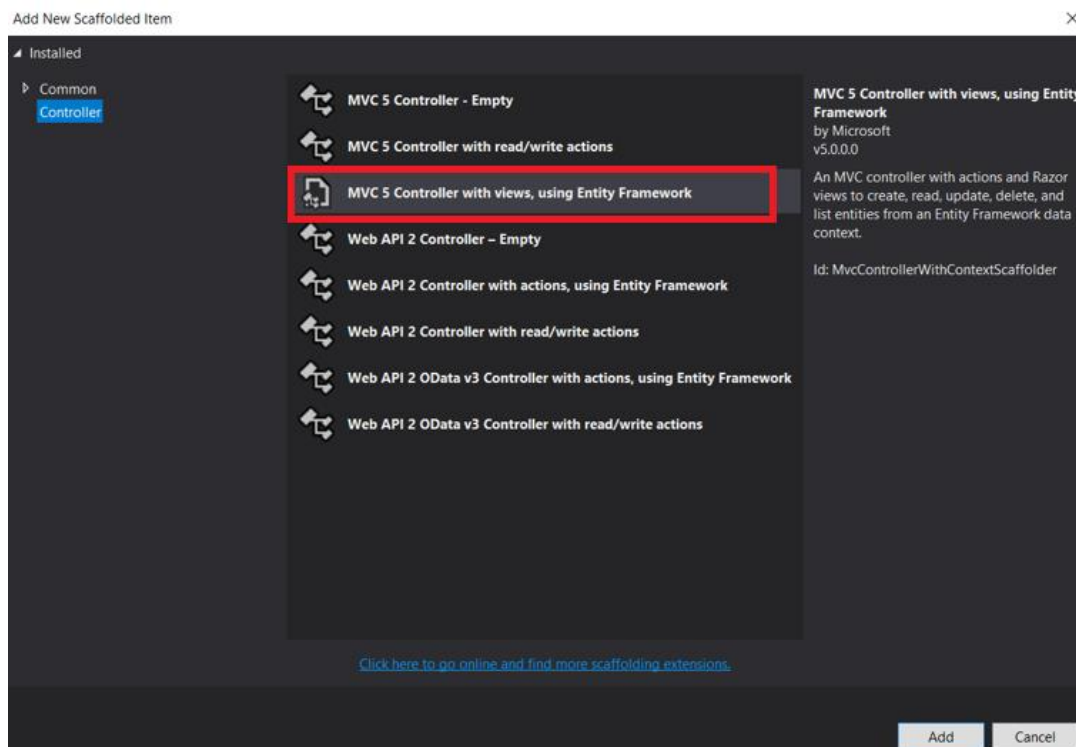
Για τη δημιουργία των controllers και των views ακολουθείται η διαδικασία που φαίνεται παρακάτω:

Η διαδικασία που ακολουθείται για την προσθήκη ενός νέου Controller είναι η ίδια για όλες τις κλάσεις του μοντέλου.



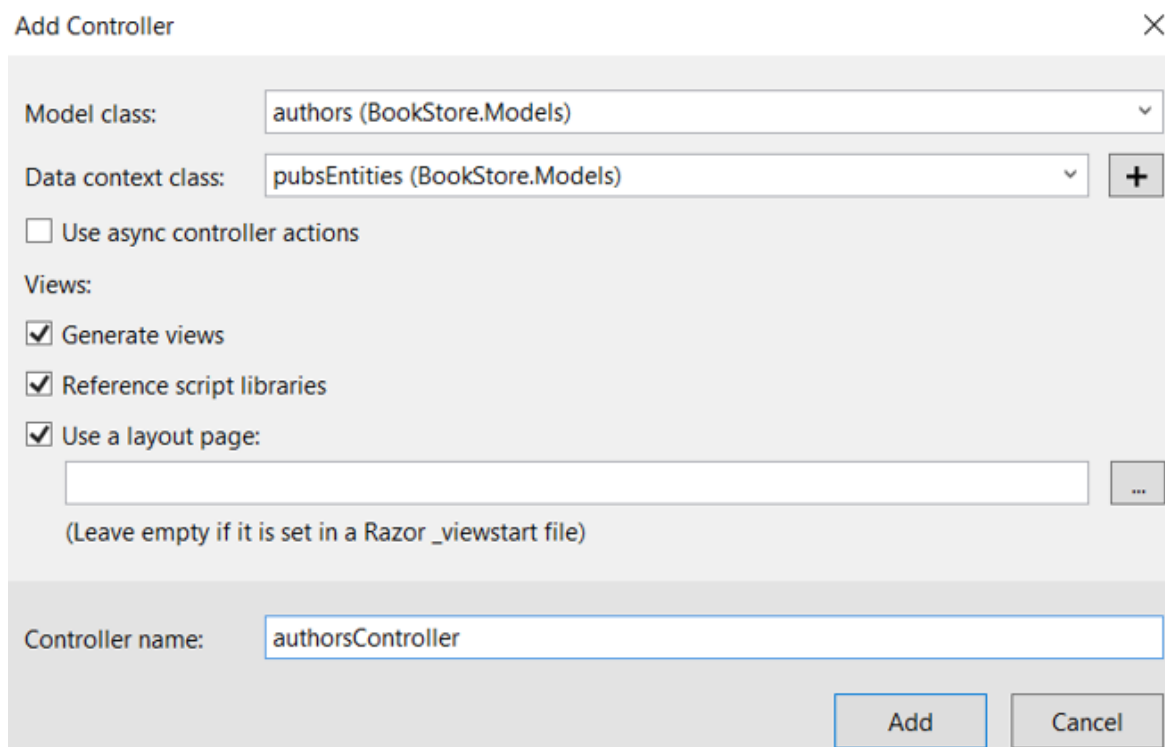
Εικόνα 6 - Διαδικασία δημιουργία controller

Στη συνέχεια, θα πρέπει να επιλέξουμε τη δημιουργία controller με actions και Razor view, ώστε να δημιουργηθούν views για την δημιουργία, επεξεργασία, διαγραφή και προβολής των δεδομένων κάθε πίνακα της βάσης δεδομένων, χρησιμοποιώντας το Entity Framework.



Εικόνα 7 - Διαδικασία δημιουργίας controller

Έπειτα, επιλέγουμε το μοντέλο και την κλάση για την οποία θα δημιουργηθεί και ο αντίστοιχος controller.



Εικόνα 8 - Διαδικασία δημιουργίας controller

Με αυτόν τον τρόπο δημιουργούνται ένας – ένας ο controller και εν συνεχεία τα αντίστοιχα views (*Index, Details, Edit, Delete, Create*). Συνεπώς, για κάθε πίνακα της βάσης δημιουργείται και ένα «σύστημα» controllers-view.

Όσο αναφορά το view, αυτό αναπαριστά με γραφικό τρόπο την πληροφορία που περιέχει το model δημιουργώντας μια γραφική αναπαράσταση στον χρήστη.

Οι controllers για να εμφανίσουν όλες τις πληροφορίες, μέσω του μοντέλου, βασίζονται στο πρωτεύον κλειδί του εκάστοτε πίνακα. Ωστόσο, υπάρχουν και controllers που διαχειρίζονται δύο ξένα κλειδιά ως πρωτεύοντα. Χαρακτηριστικό παράδειγμα είναι ο **titleauthors controller**, ο οποίος έχει σαν πρωτεύον κλειδί το συνδυασμό των δύο ξένων κλειδιών, του *title\_id* και *author\_id*. Επιπρόσθετα, υπήρχαν controllers στους οποίους έπρεπε να προσδιορίσουμε το κλειδί με το οποίο θα γινόταν η αναζήτηση και η εμφάνιση των πληροφοριών, όπως στο **sales controller**, όπου η αναζήτηση γινόταν με βάση τον αριθμό παραγγελίας (*ord\_num*).



Σημαντική προσθήκη για την καλύτερη αλληλεπίδραση του χρήστη με την εφαρμογή ήταν η διαχείριση των εξαιρέσεων (exceptions) που μπορούν να προκύψουν σε κάθε controller.

Στον παρακάτω πίνακα αποτυπώνονται για κάθε controller και οι εξαιρέσεις που μπορεί να προκύψουν.

Controller	Exception	Μήνυμα
<b>Authors</b>	Δημιουργία author με ίδιο id	Το συγκεκριμένο id υπάρχει ήδη, παρακαλώ εισάγετε κάποιο άλλο
	Edit author	Κάποιο σφάλμα προκλήθηκε, παρακαλώ δοκιμάστε ξανά
	Delete author λόγω εξάρτησης της εγγραφής με άλλον πίνακα	Η διαγραφή του συγκεκριμένου συγγραφέα δεν επιτρέπεται λόγω περιορισμών του συστήματος
<b>Employees</b>	Δημιουργία employee με ίδιο id	Το συγκεκριμένο id υπάρχει ήδη, παρακαλώ εισάγετε κάποιο άλλο
	Edit job level, αναντιστοιχία με το απαιτούμενο επίπεδο	Το Επίπεδο Δουλειάς δεν συμβαδίζει με την υπηρεσία του εργαζόμενου
	Delete employee λόγω εξάρτησης της εγγραφής με άλλον πίνακα	Η διαγραφή του συγκεκριμένου υπαλλήλου δεν επιτρέπεται λόγω περιορισμών του συστήματος
<b>Jobs</b>	Δημιουργία νέας Θέσης Εργασίας	Κάποιο σφάλμα προκλήθηκε, παρακαλώ δοκιμάστε ξανά
	Edit Θέσης Εργασίας	Κάποιο σφάλμα προκλήθηκε, παρακαλώ δοκιμάστε ξανά
	Delete job λόγω εξάρτησης της εγγραφής με άλλον πίνακα	Η διαγραφή της συγκεκριμένης δουλειάς δεν επιτρέπεται λόγω περιορισμών του συστήματος
<b>Publishers</b>	Δημιουργία publisher με ίδιο id	Το συγκεκριμένο id υπάρχει ήδη, παρακαλώ εισάγετε κάποιο άλλο
	Edit Publisher	Κάποιο σφάλμα προκλήθηκε, παρακαλώ δοκιμάστε ξανά
	Delete publisher λόγω εξάρτησης της εγγραφής με άλλον πίνακα	Η διαγραφή του συγκεκριμένου εκδοτικού οίκου δεν επιτρέπεται λόγω περιορισμών του συστήματος

<b>Sales</b>	Δημιουργία sale με ίδιο order number, stor_id, title_id	Η παραγγελία υπάρχει ήδη, παρακαλώ εισάγετε κάποιον άλλον αριθμό παραγγελίας
	Edit Sales	Κάποιο σφάλμα προκλήθηκε, παρακαλώ δοκιμάστε ξανά
<b>Stores</b>	Δημιουργία νέου βιβλιοπωλείου με ίδιο id	Το id του βιβλιοπωλείου υπάρχει ήδη, παρακαλώ εισάγετε κάποιο άλλο
	Edit store	Κάποιο σφάλμα προκλήθηκε, παρακαλώ δοκιμάστε ξανά
	Delete store λόγω εξάρτησης της εγγραφής με άλλον πίνακα	Η διαγραφή του συγκεκριμένου βιβλιοπωλείου δεν επιτρέπεται λόγω περιορισμών του συστήματος
<b>TitleAuthors</b>	Δημιουργία Title-Author	Κάποιο σφάλμα προκλήθηκε, παρακαλώ δοκιμάστε ξανά
	Edit Title-Author, σφάλμα με την σειρά του συγγραφέα (au_ord)	Παρακαλώ ελέγξτε το πεδίο Author Order
<b>Titles</b>	Δημιουργία title με ίδιο id	Το συγκεκριμένο id υπάρχει ήδη, παρακαλώ εισάγετε κάποιο άλλο
	Edit title	Κάποιο σφάλμα προκλήθηκε, παρακαλώ δοκιμάστε ξανά
	Delete store λόγω εξάρτησης της εγγραφής με άλλον πίνακα	Η διαγραφή του συγκεκριμένου βιβλίου δεν επιτρέπεται λόγω περιορισμών του συστήματος

## Δημιουργία Αναφορών

Αναφορικά με το ερώτημα για τη δημιουργία αναφορών για τους **κορυφαίους συγγραφείς (top authors)** και την **αναζήτηση παραγγελίας (search order)**, ο κώδικας βρίσκεται στο HomeController.

### Κορυφαίοι Συγγραφείς

Τα κριτήρια, τα οποία υπάρχουν στο View TopAuthorsForm, για την ανάκτηση των κορυφαίων συγγραφέων είναι:

1. Ο αριθμός Χ, το πλήθος των συγγραφέων που επιθυμούμε να εμφανίσουμε.
2. Ημερομηνία Από, η ημερομηνία που μας ενδιαφέρει να ξεκινήσουμε την αναζήτηση
3. Ημερομηνία Έως, η ημερομηνία που μας ενδιαφέρει να τερματίσουμε την αναζήτηση

Ο χρήστης στη συγκεκριμένη φόρμα μπορεί να εισάγει τις τιμές που θέλει, ώστε να περιορίσει το εύρος αναζήτησης. Να σημειωθεί, ότι οποιοδήποτε από τα πεδία μπορούν να είναι κενά κατά την υποβολή του ερωτήματος. Στην περίπτωση που όλα τα πεδία είναι κενά, τότε εμφανίζονται όλα τα στοιχεία επικοινωνίας όλων των συγγραφέων που τα βιβλία τους έχουν τουλάχιστον μία πώληση.

Η μέθοδος που διαχειρίζεται το συγκεκριμένο ερώτημα είναι η **TopAuthors** με την μέθοδο «HttpGet». Το ερώτημα που εκτελείται στη βάση για την εξαγωγή των αποτελεσμάτων βασίζεται στην ακόλουθη λογική. Αρχικά, αφού εκτελέσουμε join ανάμεσα στους πίνακες titleauthors και authors με τη συνθήκη ότι έχουν το ίδιο au\_id, join ανάμεσα στους πίνακες titleauthors και sales με τη συνθήκη ότι έχουν το ίδιο title\_id, αποθηκεύουμε σε ένα Anonymous Object τις στήλες με τα αποτελέσματα που έχουν προκύψει. Στη συνέχεια, οι συγγραφείς ομαδοποιούνται και με τη βοήθεια της συναθροιστικής συνάρτησης “SUM” υπολογίζεται το άθροισμα των ποσοτήτων που έχουν πουληθεί για ένα συγκεκριμένο διάστημα και όπως αυτό έχει οριστεί από τον χρήστη.

## Αναζήτηση Παραγγελίας

Τα κριτήρια , τα οποία υπάρχουν στο View OrdersForm, για την ανάκτηση των παραγγελιών που έχουν γίνει από ένα κατάστημα για ένα συγκεκριμένο χρονικό διάστημα είναι:

1. Όνομα Καταστήματος, εισαγωγή είτε ολόκληρου του ονόματος είτε μέρος αυτού
2. Ημερομηνία Από, η ημερομηνία που μας ενδιαφέρει να ξεκινήσουμε την αναζήτηση
3. Ημερομηνία Έως, η ημερομηνία που μας ενδιαφέρει να τερματίσουμε την αναζήτηση

Όπως και στην δημιουργία της προηγούμενης αναφοράς, ο χρήστης μπορεί να εισάγει τις τιμές που θέλει, ώστε να περιορίσει το εύρος αναζήτησης, αλλά ταυτόχρονα δύναται να αφήσει και κάποιο πεδίο/πεδία κενά. Η μέθοδος που διαχειρίζεται το συγκεκριμένο ερώτημα είναι η **Order** με την μέθοδο «HttpPost». Το ερώτημα που εκτελείται στη βάση για την εξαγωγή των αποτελεσμάτων βασίζεται στην ακόλουθη λογική. Μόλις εκτελέσουμε join ανάμεσα στους πίνακες sales και stores με τη συνθήκη ότι έχουν το ίδιο stor\_id, join ανάμεσα στους πίνακες titles και sales με τη συνθήκη ότι έχουν το ίδιο title\_id, επιλέγουμε τις στήλες που μας ενδιαφέρουν, ώστε αυτό να προβληθεί στον χρήστη. Το τελικό αποτέλεσμα προκύπτει μετά από την εφαρμογή των συνθηκών για το διάστημα που έγινε η παραγγελία και για το όνομα το μαγαζιού με την βοήθεια του τελεστή “LIKE”.

## LAYOUT.CSHTML

Το συγκεκριμένο αρχείο μπορεί να θεωρηθεί ως η βάση για κάθε ένα από τα Views, αφού περιέχεται το «περίγραμμα» των σελίδων. Πιο συγκεκριμένα, με τη βοήθεια ενός Partial View, **\_Menu.cshtml**, έχει προστεθεί ένα navbar το οποίο περιέχει όλες τις σελίδες που μπορεί να περιηγηθεί ο χρήστης. Οι σελίδες αυτές είναι οι εξής: *Αρχική, Επικοινωνία, Σχετικά Με Εμάς, Συγγραφείς, Βιβλιοπωλεία, Εκπτώσεις, Εργαζόμενοι, Εκδοτικοί Οίκοι, Θέσεις Εργασίας, Πωλήσεις, Title/Authors, Τίτλοι Βιβλίων, Royscheds, Publishers Info*. Όλες οι σελίδες που αφορούν τη βάση δεδομένων και τη διαχείριση των δεδομένων (data management) ανήκουν σε ένα Navbar-Item Dropdown.

## METADATA

Τα Metadata<sup>1</sup> αποτελούν τους κανόνες για το μοντέλο μας. Με την χρήση αυτών θέτουμε ορισμένους κανόνες σε κάθε σελίδα μας. Πιο συγκεκριμένα καθορίζουμε ποια από τα πεδία είναι υποχρεωτικά, τι μορφή πρέπει να έχουν, ποιο είναι το ελάχιστο μήκος (length) που πρέπει να έχουν, το εύρος τιμών, αλλά και τι χαρακτήρες θα δέχονται. Σε περίπτωση που κάποιο πεδίο δεν συμπληρωθεί σωστά, τότε εμφανίζεται στο αντίστοιχο πεδίο το κατάλληλο μήνυμα, ώστε να βοηθήσει τον χρήστη να διορθώσει το λάθος. Επιπρόσθετα, μπορούμε να αξιοποιήσουμε τους συγκεκριμένους κανόνες, ώστε να θέσουμε τις επικεφαλίδες στους πίνακες και στα πεδία που εμφανίζονται τα δεδομένα.

Παρακάτω ακολουθεί ένα παράδειγμα για την κλάση του μοντέλου για τους συγγραφείς (authorsMetadata):

```
public class AuthorMetadata
{
    [Display(Name = "Id")]
    [Required(ErrorMessage = "Το πεδίο ID απαιτείται")]
    [RegularExpression("[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9][0-9]", ErrorMessage = "Το Id πρέπει να είναι της μορφής XXX-XX-XXXX")]
    public string au_id;
    [Display(Name = "Επώνυμο")]
    [Required(ErrorMessage = "Το πεδίο Επώνυμο απαιτείται")]
    [StringLength(40, MinimumLength = 2, ErrorMessage = "Το Επώνυμο πρέπει να είναι μεταξύ 3 και 40 χαρακτήρων")]
    public string au_lname;
    [Display(Name = "Όνομα")]
    [Required(ErrorMessage = "Το πεδίο Όνομα απαιτείται")]
    [StringLength(20, MinimumLength = 2, ErrorMessage = "Το Όνομα πρέπει να είναι μεταξύ 3 και 20 χαρακτήρων")]
    public string au_fname;
    [Display(Name = "Τηλέφωνο")]
    [Required(ErrorMessage = "Το πεδίο Τηλέφωνο απαιτείται")]
    [StringLength(12, MinimumLength = 10, ErrorMessage = "Το Τηλέφωνο πρέπει να είναι μεταξύ 10 και 13 χαρακτήρων")]
    public string phone;
    [Display(Name = "Διεύθυνση")]
    [Required(ErrorMessage = "Το πεδίο Διεύθυνση απαιτείται")]
    [StringLength(40, MinimumLength = 3, ErrorMessage = "Η Διεύθυνση πρέπει να είναι μεταξύ 3 και 40 χαρακτήρων")]
    public string address;
    [Display(Name = "Πόλη")]
    [Required(ErrorMessage = "Το πεδίο Πόλη απαιτείται")]
    [StringLength(20, MinimumLength = 3, ErrorMessage = "Η Πόλη πρέπει να είναι μεταξύ 3 και 20 χαρακτήρων")]
    public string city;
    [Display(Name = "Πολιτεία")]
    [Required(ErrorMessage = "Το πεδίο Πολιτεία απαιτείται")]
    [StringLength(2, MinimumLength = 2, ErrorMessage = "Η Πολιτεία πρέπει να είναι 2 χαρακτήρες")]
    public string state;
    [Display(Name = "Τ.Κ.")]
    [Required(ErrorMessage = "Το πεδίο Τ.Κ. απαιτείται")]
    [RegularExpression("[0-9][0-9][0-9][0-9][0-9]", ErrorMessage = "Ο Ταχυδρομικός Κώδικας πρέπει να αποτελείται από 5 ψηφία")]
    public string zip;
    [Display(Name = "Συμβόλαιο")]
    public bool contract;
}
```

Εικόνα 9 – AuthorsMetadata

Για παράδειγμα τα πεδία Id, Επώνυμο, Όνομα, Τηλέφωνο, Διεύθυνση, Πόλη, Πολιτεία και Ταχυδρομικός Κώδικας (Τ.Κ.) είναι υποχρεωτικά, οπότε εάν ο

<sup>1</sup> Τα Metadata βρίσκονται στο αρχείο Class1.cs

χρήστης δεν τα συμπληρώσει δεν μπορεί να ολοκληρώσει την ενέργεια που επιθυμεί, όπως την δημιουργία ή επεξεργασία των στοιχείων ενός συγγραφέα, και εμφανίζεται το σχετικό μήνυμα (π.χ.: «Το πεδίο Id απαιτείται»).

Επιπρόσθετα, τα πεδία Id και T.K. πρέπει να έχουν συγκεκριμένη μορφή. Σε περίπτωση που δεν ικανοποιούνται τα κριτήρια, υποδεικνύεται από το σύστημα η μορφή που πρέπει να έχει το κάθε πεδίο, ώστε να ολοκληρωθεί η διαδικασία. Αντίστοιχα, για τα πεδία Επώνυμο, Όνομα, Τηλέφωνο, Διεύθυνση, Πόλη και Πολιτεία πρέπει να έχουν συγκεκριμένο αριθμό χαρακτήρων.

Συνεπώς, η ορθή συμπλήρωση των πεδίων, σύμφωνα με τους «κανόνες» που ορίζει το αρχείο Class1, ο χρήστης ολοκληρώνει της εκάστοτε διαδικασία.

## Μορφοποίηση

Για την μορφοποίηση των σελίδων, πέρα από κλάσεις του Bootstrap v.4.5 που χρησιμοποιήθηκαν, αξιοποιήθηκαν και ορισμένα αρχεία και φάκελοι για την προσθήκη εικόνων και του λογότυπου της εφαρμογής. Αρχικά, στον φάκελο **Content** υπάρχει το αρχείο *Site.css* όπου έχει τροποποιηθεί προκειμένου να μορφοποιηθούν ορισμένα κουμπιά στις διάφορες φόρμες. Επίσης, στον φάκελο **assets/css** υπάρχει το αρχείο *Contact\_Us.css* που περιλαμβάνει κώδικα για την μορφοποίηση της σελίδας *Επικοινωνία*. Τέλος, στον φάκελο **assets/img** υπάρχουν οι φωτογραφίες και το λογότυπο της εφαρμογής.



## Βιβλιογραφία

Παρακάτω θα αναφερθούν οι βιβλιογραφικές πηγές που χρησιμοποιήσαμε για την εν λόγω υλοποίηση :

1. [Asp.Net -Documentation](#)
2. [Bootstrap Documentation](#)
3. [MVC With LINQ to SQL](#)
4. [MVC Tutorial](#)