

Αυτόνομη διαδικτυακή πλατφόρμα για την περιήγηση και εξερεύνηση σημασιολογικών (RDF) δεδομένων

Alexandros Vasilakis  
CSD 3612

Supervisor: Pavlos Fafalios

Professor: Yannis Tzitzikas

Member of the monitoring/evaluation Committee: Dimitris Plexousakis

Ηράκλειο, Ιούλιος 2024

Table of Contents

[Περίληψη 6](#_Toc170078474)

[Βιβλιοθήκες που χρησιμοποιήθηκαν: 6](#_Toc170078475)

[Λειτουργίες συστήματος 7](#_Toc170078476)

# Λίστα εικόνων

[Εικόνα 1: Αρχική σελίδα εφαρμογής 9](#_Toc170082747)

Πίνακας Περιεχομένων

1. Περίληψη
2. Λίστα εικόνων
3. Απαιτήσεις Συστήματος
4. Αντιμετώπιση προβλημάτων
5. Επίλογος
6. Αναφορές / Πηγές
7. Παραρτήματα
8. Αναφορά pom.xml file

# Περίληψη

Ο σκοπός της παρούσας διπλωματικής εργασίας ήταν η δημιουργία μιας αυτόνομης διαδικτυακής πλατφόρμας για την εύκολη και γρήγορη περιήγηση και εξερεύνηση σημασιολογικών δεδομένων. Η προτεινόμενη μέθοδος αποθήκευσης και μεταφοράς δεδομένων είναι μια τεχνολογία του Σημασιολογικού Ιστού, το RDF. Σύμφωνα με το RDF(Resource Description Framework), τα δεδομένα αποθηκεύονται με την μορφή triples(subject-predicate-object)(τριπλέτες(Υποκείμενο-Κατηγόρημα-Αντικείμενο)).

Η SPARQL αποτελεί την βασική γλώσσα για την υποβολή ερωτημάτων και χρησιμοποιείται για την επεξεργασία μιας RDF βάσης δεδομένων.

Προκειμένου να γίνει η φόρτωση και η διαχείριση των RDF δεδομένων ήταν απαραίτητο να χρησιμοποιηθεί ένας εξυπηρετητής δικτύου (web server). Στην συγκεκριμένη εφαρμογή χρησιμοποιήθηκε ο Αpache Tomcat, που αποτελεί έναν open source Semantic Web Framework(πλαίσιο ανοικτού κώδικα Σημασιολογικού Ιστού) που παρέχει υποστήριξη Java.

…

Στα πλαίσια της διπλωματικής εργασίας χρησιμοποιήθηκαν βιβλιοθήκες όπως AG Grid και Chart JS, για την δημιουργία πινάκων και διαγραμμάτων που αφορούσαν τα δεδομένα που που χρειάζονταν να διαχειριστούν

Η παρούσα εφαρμογή αποτελεί την δημιουργία ενός Java Web Application και χρήση της βιβλιοθήκης Apache Jena(server) για διάβασμα των RDF αρχείων που υπάρχουν σε ένα φάκελο

Εμφάνιση στην εφαρμογή των αποτελεσμάτων των SPARQL ερωτημάτων.

Δημιουργήθηκε ένα configuration αρχειου (config.properties) στο οποίο έχουμε ορίσει διάφορες παραμέτρους του συστήματος καθώς και λεπτομέρειες που αφορούν τα δεδομένα, όπως τις κατηγορίες οντοτήτων που μας ενδιαφέρει να εμφανίζονται στην εφαρμογή.

Χρησιμοποιήθηκαν δεδομένα του έργου SeaLit(στον τομέα της Ναυτικής Ιστορίας)

# Αποκωδικοποίηση RDF αρχείων

Το RDF αποτελεί ένα μοντέλο δεδομένων για το διαδίκτυο. Χρησιμοποιείται για την αναπαράσταση πληροφοριών που αφορούν πηγές του διαδικτύου, χωρίς να είναι απαραίτητο ότι τις παίρνουμε απευθείας από αυτό

(Μπορεί να υπάρχουν σε ένα βιβλίο, ή να αναφέρονται από κάποιον άνθρωπο).

Το RDF προορίζεται για καταστάσεις όπου οι πληροφορίες υποβάλλονται σε επεξεργασία από εφαρμογές. Αυτές οι εφαρμογές θα δώσουν το τελικό αποτέλεσμα που θα εμφανιστεί στον χρήστη.

Υπάρχουν διάφορες μορφές RDF δεδομένων:

* TRIG
* RDF/XML
* N-Triples
* N-Quads
* Turtle

Αυτή η μέθοδος χρησιμοποιείται για την καλύτερη σύνδεση, εύρεση και κοινοποίηση δεδομένων.

Ουσιαστικά υποδηλώνει τον Σημασιολογικό Ιστό(Semantic Web/Network)

Ο βασικός τρόπος αναπαράστασης των δεδομένων είναι triples(τριπλέτες)  
(subject-predicate-object)(Υποκείμενο-Κατηγόρημα-Αντικείμενο), όπου:

* Υποκείμενο: αποτελεί το «στοιχείο» που ενεργεί
* Κατηγόρημα: η ιδιότητα του υποκειμένου που μας ενδιαφέρει
* Αντικείμενο: Η τιμή ή η παραπομπή σε άλλο υποκείμενο(για να βρεθεί η τιμή) στην οποία γίνεται η ενέργεια του κατηγορήματος

# SPARQL

# Βιβλιοθήκες που χρησιμοποιήθηκαν:

## AG Grid:

Για την δημιουργία του πίνακα, χρησιμοποιήθηκε η παρούσα βιβλιοθήκη μιας και οι λειτουργίες που παρέχει είναι πολυάριθμες.

Ο πίνακας οριοθετεί με κατάλληλο τρόπο τα columns του  
(Τα columns προκύπτουν με βάση τα queries που υπάρχουν στο configuration file)

Με τα δεδομένα που χρησιμοποιήθηκαν εμφανίζει:

* + Ανάλογα με την οντότητα που θα επιλεγεί

Persons

Locations

Voyages

Ships

* Οι βασικές πληροφορίες για κάθε καταχώρηση, το οποίο προκύπτει από ένα SPARQL Query
* Επιπλέον στήλη που παραπέμπει σε σελίδα με περισσότερες πληροφορίες για κάθε καταχώρηση
* Στο παρακάτω παράδειγμα εμφανίζεται αυτό στο οποίο αναφερόμαστε:

Έστω ότι επιλέγουμε την κατηγορία Ships -> εμφανίζονται όλες οι καταχωρήσεις που έχουμε για αυτή την κατηγορία με κάποιες βασικές πληροφορίες(columns). Στην προκειμένη περίπτωση είναι:

* URI
* Label
* ShipType

 -> Αφού για μια καταχώρηση(record)(row) επιλέξουμε να δούμε περισσότερες πληροφορίες πατώντας το “More Info”, σε νέα σελίδα εμφανίζονται (αν υπάρχουν)(εδώ έχει προστεθεί η δυνατότητα OPTIONAL, σε περιπτώσεις που για κάποιο πεδίο δεν είναι γνωστή αυτή η πληροφορία) περισσότερα πεδία. Στην περίπτωση του παραδείγματος είναι:

* Identity
* Registered In
* Owner
* Tonnage
* Constructed By

# Λειτουργίες συστήματος

1. Φίλτρα

Στον πίνακα όπου εμφανίζονται τα δεδομένα κάθε κατηγορίας, για κάθε στήλη(ετικέτα-label) υπάρχουν φίλτρα. Αυτά τα φίλτρα είναι από τις πιο χρήσιμες λειτουργίες της εφαρμογής, καθώς οι χρήστες μπορούν με άμεσο και γρήγορο τρόπο να βρουν οποιαδήποτε πληροφορία χρειάζεται.

Το φίλτρο που θα έχει κάθε στήλη προκύπτει από το περιεχόμενο τους:

* Αν περιλαμβάνουν κείμενο, τότε ο χρήστης μπορεί να ψάξει με βάση κάποια αλφαβητικό χαρακτήρα
* Αν περιλαμβάνουν ψηφία, τότε ο χρήστης μπορεί να ψάξει με βάση κάποιο αριθμητικό χαρακτήρα
* Αν περιλαμβάνουν ημερομηνίες, τότε ο χρήστης μπορεί να ψάξει τοποθετώντας συγκεκριμένα την χρήση ενός συγκεκριμένου template που αφορά ημερομηνίες

1. Διαγράμματα(Charts)

Τα διαγράμματα προκύπτουν από τα δεδομένα κάθε στήλης και σχετίζονται με τον αριθμό εμφάνισης των δεδομένων που περιέχουν.  
Ανάλογα την περίπτωση μπορούν να περιέχουν σημαντική πληροφορία στον χρήστη, ανάλογα με την στήλη κάποιας οντότητας στην οποία βρίσκετε εκείνη την στιγμή.

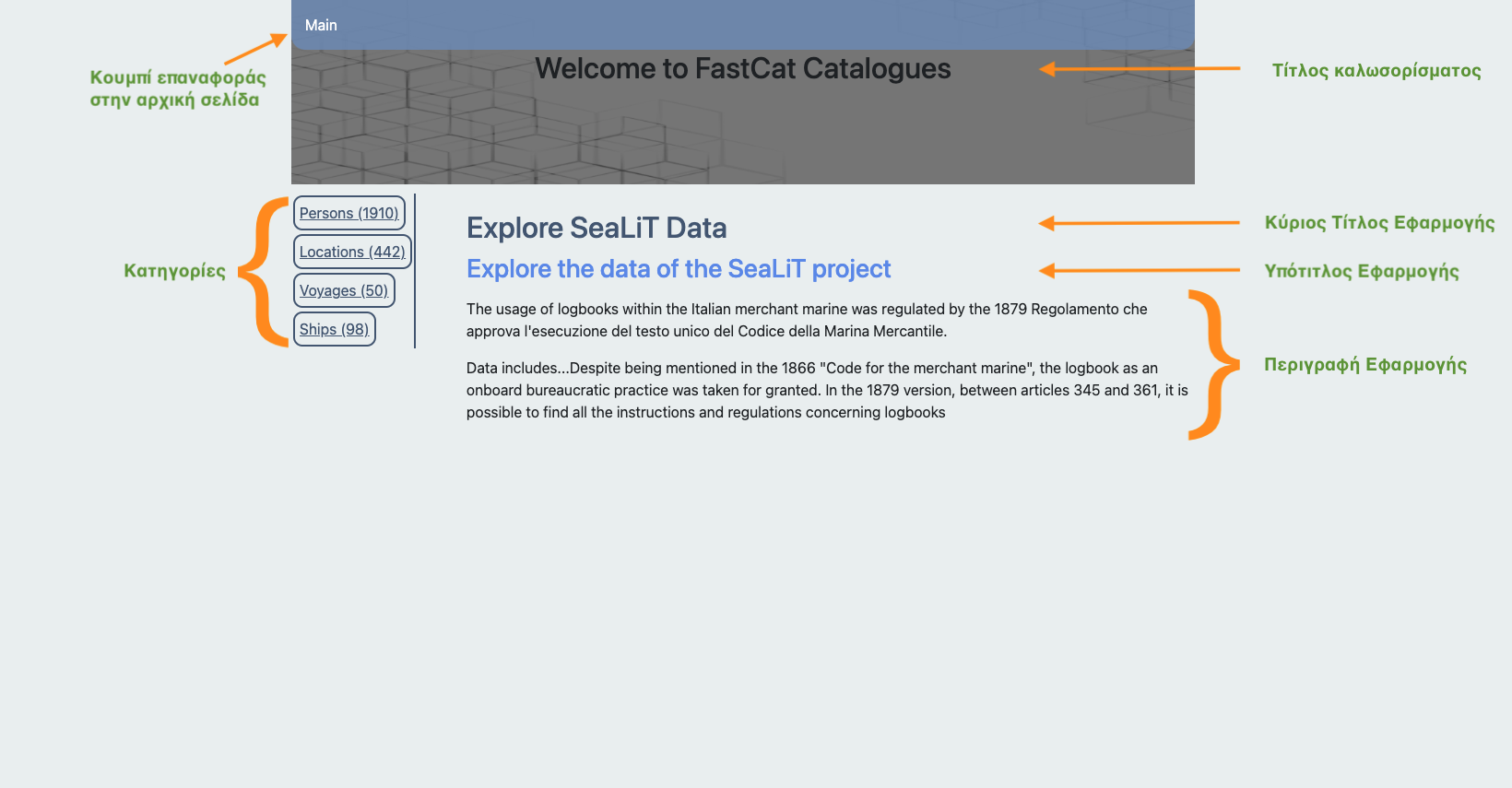
# Δεδομένα εφαρμογής

Τα δεδομένα που χρησιμοποιήθηκαν στην παρούσα εφαρμογή αφορούσαν πληροφορίες από την Ναυτική Ιστορία. Πιο συγκεκριμένα:

* Τοποθεσίες
* Οργανισμοί
* Ονόματα
* Πλοία
* Εγγραφές (Το αρχείο με την περισσότερη και πιο ουσιαστική πληροφορία)
* Λεξιλόγια

# Πλοήγηση εφαρμογής

Κατά την είσοδο στην εφαρμογή βλέπουμε την αρχική σελίδα:

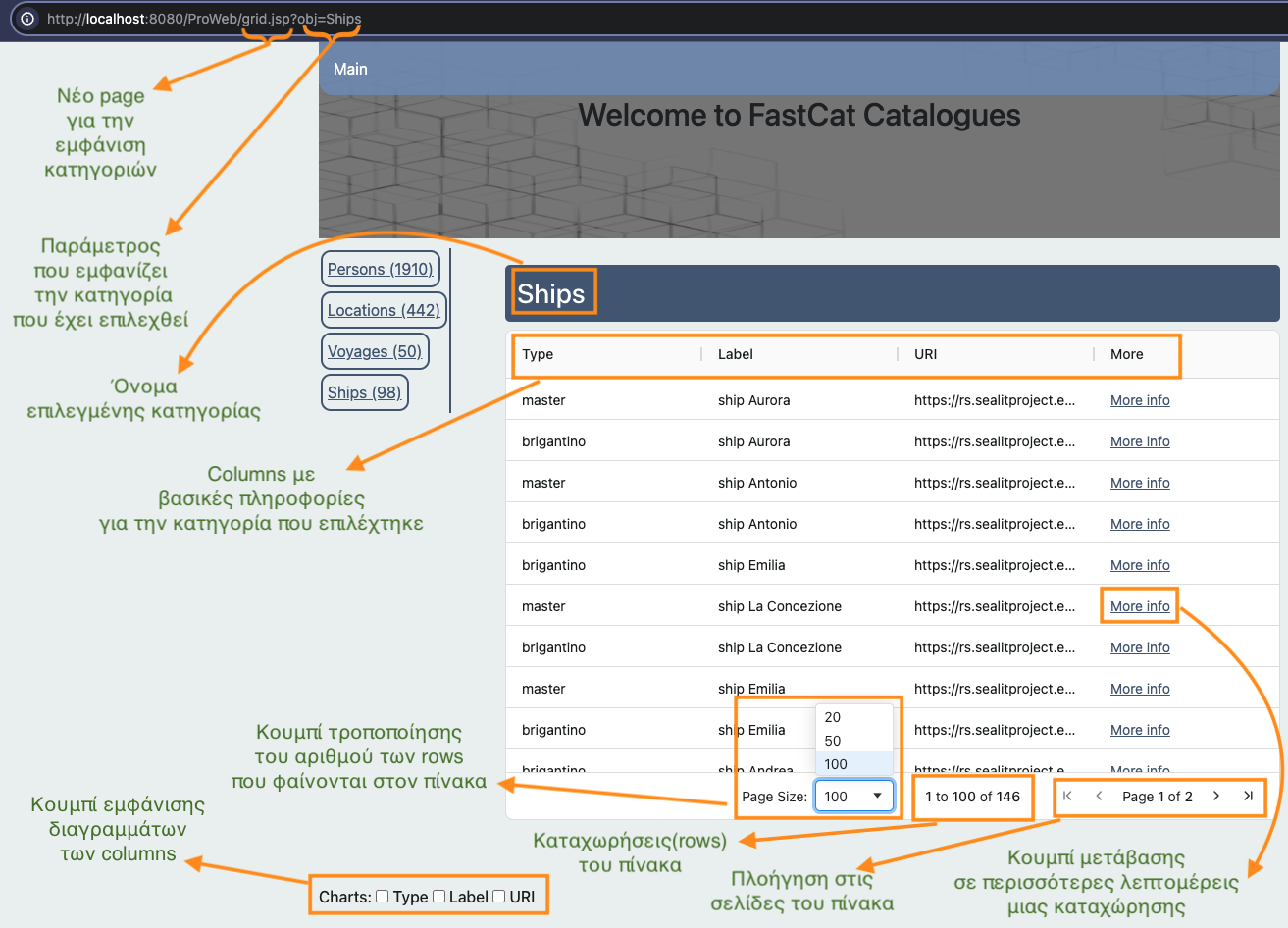


Εικόνα 1: Αρχική σελίδα εφαρμογής

Υπάρχουν:

* + Κουμπί επαναφορά στην αρχική σελίδα
  + Κατηγορίες εφαρμογής
  + Τίτλος Καλωσορίσματος
  + Κύριος Τίτλος Εφαρμογής
  + Υπότιτλος Εφαρμογής
  + Περιγραφή Εφαρμογής

Αφού επιλεχθεί μια κατηγορία εμφανίζεται η παρακάτω σελίδα:



Εικόνα 2: Σελίδα εμφάνισης μιας κατηγορίας

Υπάρχουν:

* + Στο URL:
    - grid.jsp: Το νέο page που δημιουργήθηκε για την αναπαράσταση των δεδομένων κάθε κατηγορίας
    - obj: Αποτελεί την παράμετρο που αναδεικνύει το όνομα της κατηγορίας που έχει επιλεχθεί

Εδώ θα είναι μια από τις:

* + - * Persons
      * Locations
      * Voyages
      * Ships
  + Δεξιά από τις κατηγορίες υπάρχει:
    - Όνομα επιλεγμένης κατηγορίας
    - Πίνακας δεδομένων επιλεγμένης κατηγορίας:
      * Στην πρώτη γραμμή εμφανίζονται τα ονόματα των βασικών columns που επιλέχτηκαν(μέσω του query)
      * Στο κάτω μέρος υπάρχουν οι λειτουργίες του:
        1. Δυνατότητα αλλαγής του μεγέθους των γραμμών της σελίδας
        2. Αριθμός γραμμών στην παρούσα σελίδα και των υπολειπόμενων
        3. Σελιδοποίηση
      * Στην τελευταία στήλη του πίνακα υπάρχει κουμπί μετάβασης σε περισσότερες λεπτομέρειες για την συγκεκριμένη καταχώρηση

Στην κάτω μεριά της προηγούμενης εικόνας υπάρχουν τα διαγράμματα(Charts)



Εικόνα 3: Εμφάνιση διαγραμμάτων columns μιας κατηγορίας

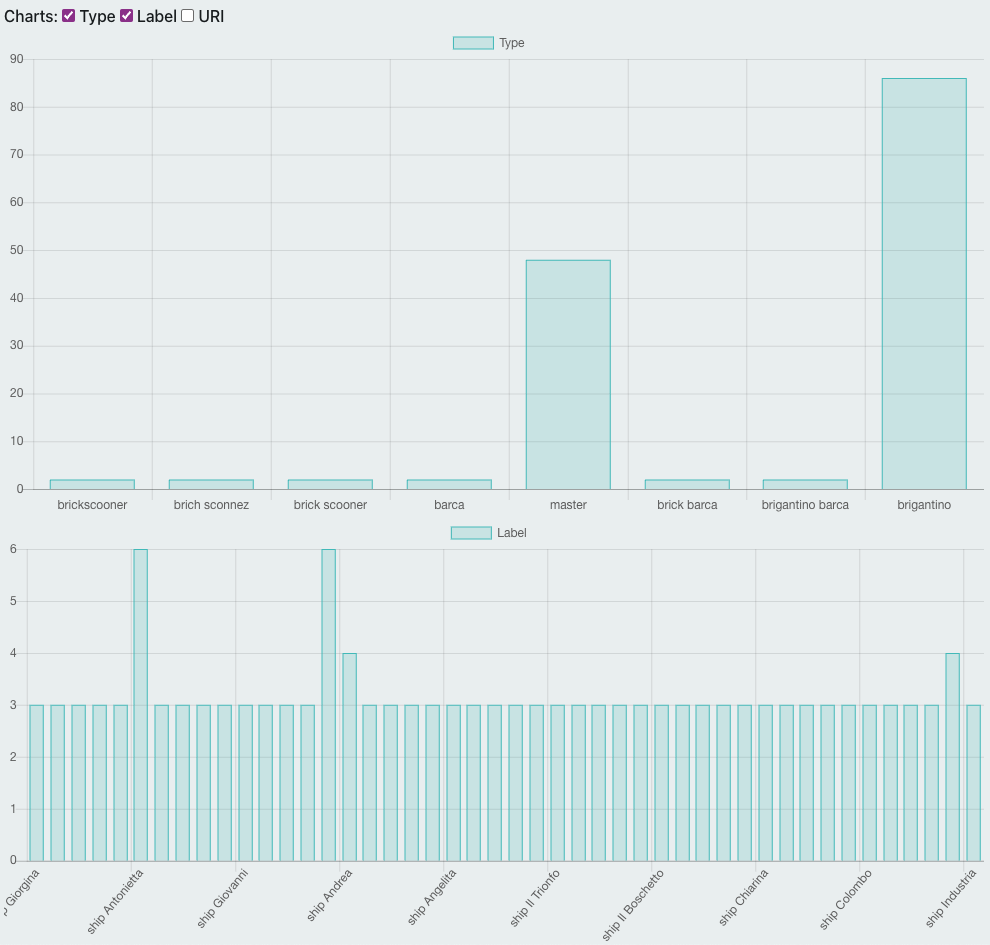
Σε αυτά ο χρήστης μπορεί να επιλέξει να δει το διάγραμμα που επιθυμεί.

Σε καθένα απ’ αυτά υπάρχουν οι εξής πληροφορίες:

* + Στον κάθετο άξονα(στα αριστερά), υπάρχει ο αριθμός εμφάνισης των στοιχείων της επιλεγμένης στήλης του πίνακα.  
    (Το εύρος των τιμών προκύπτει από τα δεδομένα)
  + Ο οριζόντιος άξονας (στην κάτω μεριά), δείχνει τα στοιχεία του επιλεγμένου column
  + Όταν o κέρσορας πηγαίνει σε μια ομάδα στοιχείων του διαγράμματος, εμφανίζονται:
    - Το όνομα της ομάδας
    - Η ποσότητα εμφάνισης τους στον πίνακα

Η επιλογή του παραπάνω διαγράμματος έγινε επειδή αποτελεί ένα πολύ καλό παράδειγμα μιας και δίνει χρήσιμα αποτελέσματα.

Επιπλέον μπορούν να εμφανιστούν και πολλαπλά διαγράμματα την ίδια στιγμή



Εικόνα 4: Δυνατότητα εμφάνισης πολλαπλών διαγραμμάτων την ίδια στιγμή

Για την δημιουργία των charts, είχε επιλεγεί αρχικά το AG Charts, ωστόσο στην συνέχεια, λόγω πιο εύκολης και δωρεάν χρήση της, επιλέχτηκε η βιβλιοθήκη Chart JS.

Through .trig files, I found more information about a single record(That’s how I add them)

# Configuration properties file

Στα αρχεία που έχουν δημιουργηθεί στο παρόν project, υπάρχει εκείνο που αφορά την διαμόρφωση των ρυθμίσεων(config.properties).

Σε αυτό μπορούμε να οριστεί (σε ένα preprocessing βήμα) διάφορες παραμέτροι του συστήματος, καθώς και λεπτομέρειες που αφορούν τα δεδομένα, όπως τις κατηγορίες οντοτήτων που μας ενδιαφέρει να εμφανίζονται στην εφαρμογή.

Αποτελεί το βασικό αρχείο, η βάση δεδομένων, που χρησιμοποιεί η εφαρμογή όπου:  
- από την μια, με τα τωρινά δεδομένα, υπάρχει το ενδεχόμενο-δυνατότητα να δεχτεί αλλαγές με την πάροδο του χρόνου

- από την άλλη, αν αλλάξουν τα δεδομένα, τότε αλλάζουν κι αυτά που παρουσιάζονται στην πλατφόρμα.

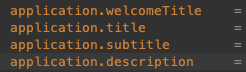
Σε κάθε περίπτωση, εφόσον η αλλαγή γίνει προσεκτικά και στοχευμένα, η πλατφόρμα θα παρουσιάσει οτιδήποτε περιέχει το παρόν αρχείο, χωρίς να δημιουργηθεί το οποιοδήποτε πρόβλημα.

Το configuration file χρησιμοποιείται για δοκιμή, επιβεβαίωση, διόρθωση και βελτιστοποίηση των λειτουργιών της εφαρμογής που δημιουργήθηκε.

Έχει δομηθεί με τον εξής τρόπο:  
Περιλαμβάνει βασικά στοιχεία, αποκλειστικά για την εφαρμογή, τα οποία είναι:

* 1. Τίτλος Καλωσορίσματος
  2. Κύριος Τίτλος Εφαρμογής
  3. Υπότιτλος Εφαρμογής
  4. Περιγραφή Εφαρμογής

(Με τον τρόπο που φαίνονται στο πρώτο μέρος της [Πλοήγησης Εφαρμογής](#_Πλοήγηση_εφαρμογής))

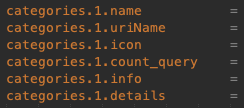


Ύστερα υπάρχει :

* + το πεδίο που περιλαμβάνει τον αριθμό των κατηγοριών των οντοτήτων που υπάρχουν και στην συνέχεια,
  + στοιχεία που αφορούν κάθε μια κατηγορία( π.χ. categories.X.name (όπου “X”, ο αριθμός για κάθε κατηγορία)



* + - Όνομα
    - ΌνομαURI
    - Εικονίδιο
    - Αριθμό Εμφανίσεων (Αφορά τον αριθμό που ένα query βρίσκει μια κατηγορία με το όνομα που μας ενδιαφέρει)
    - Πληροφορίες (Μέσω ενός query, παίρνουμε τα βασικά columns που θέλουμε να εμφανίζονται σαν πρώτη εικόνα στον πίνακα κάθε κατηγορίας)
    - Λεπτομέρειες (Μέσω ενός query, παρέχονται περισσότερες λεπτομέρειες για το row, που ο χρήστης επέλεξε να δει)



Στο παρόν project, όπως προαναφέρθηκε, έχουν χρησιμοποιηθεί δεδομένα του έργου SeaLit(στον τομέα της Ναυτικής Ιστορίας).

Κάποια από τα σημαντικότερα στοιχεία που αναφέρονται είναι:

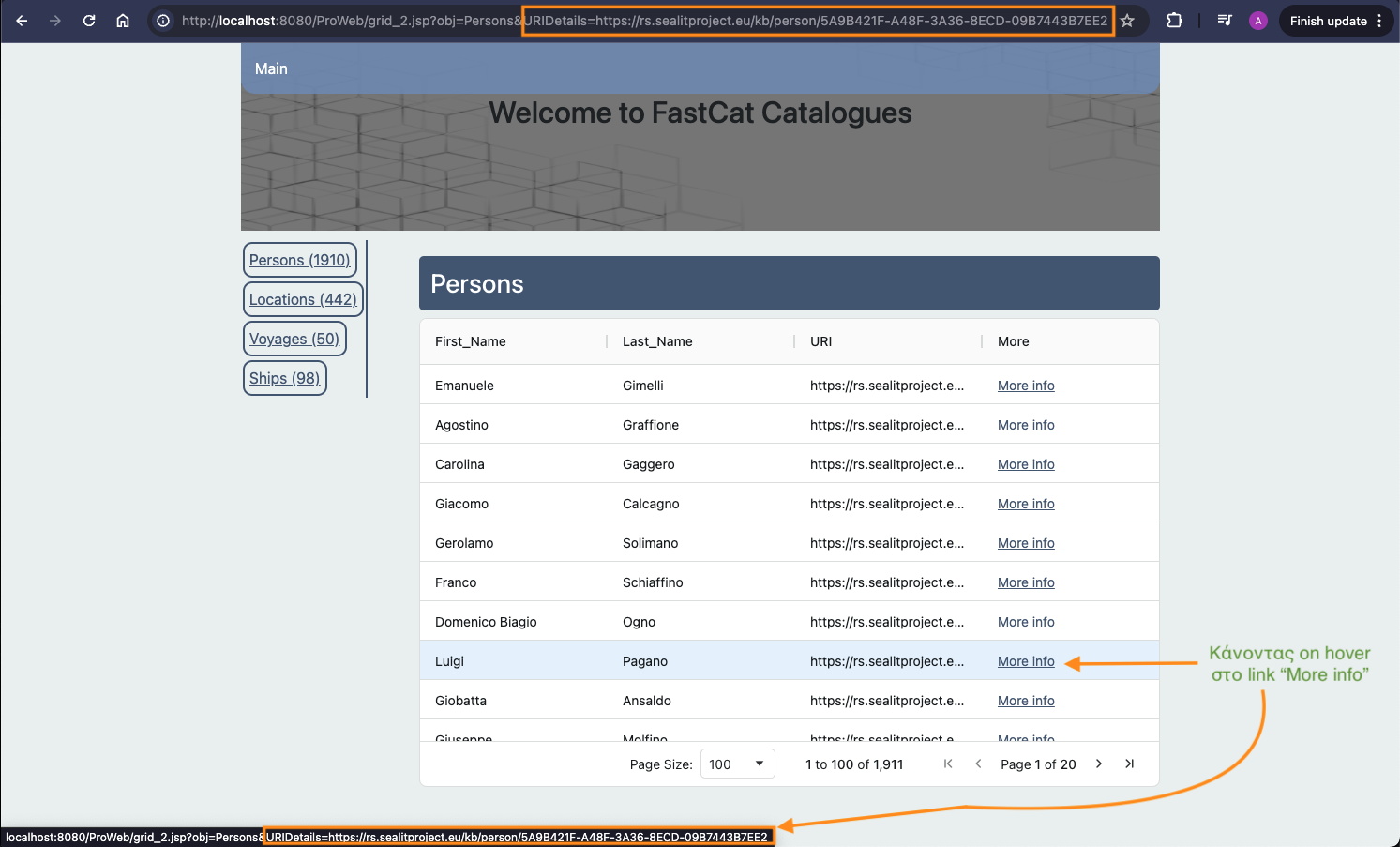
* + Πρόσωπα (Persons)
  + Τοποθεσίες (Locations)
  + Ταξίδια(Voyages)
  + Πλοία (Ships)
  + Ημερομηνίες (Dates)
  + Χρονικά διαστήματα (Spans)
  + Αναγνωριστικά (Identifiers)

**Προβλήματα** που προέκυψαν:  
Υπάρχει η πιθανότητα κάποιο column του query, λόγω έλλειψης δεδομένων, να μην έχει τιμή.

**Επίλυση**:  
Σε αυτή την περίπτωση, έχει μπει σε κάθε τέτοιο column η επιπλέον παράμετρος OPTIONAL, προκειμένου να μην δημιουργείται πρόβλημα στην εμφάνιση των αποτελεσμάτων.

**Χωρίς την παράμετρο OPTIONAL:**

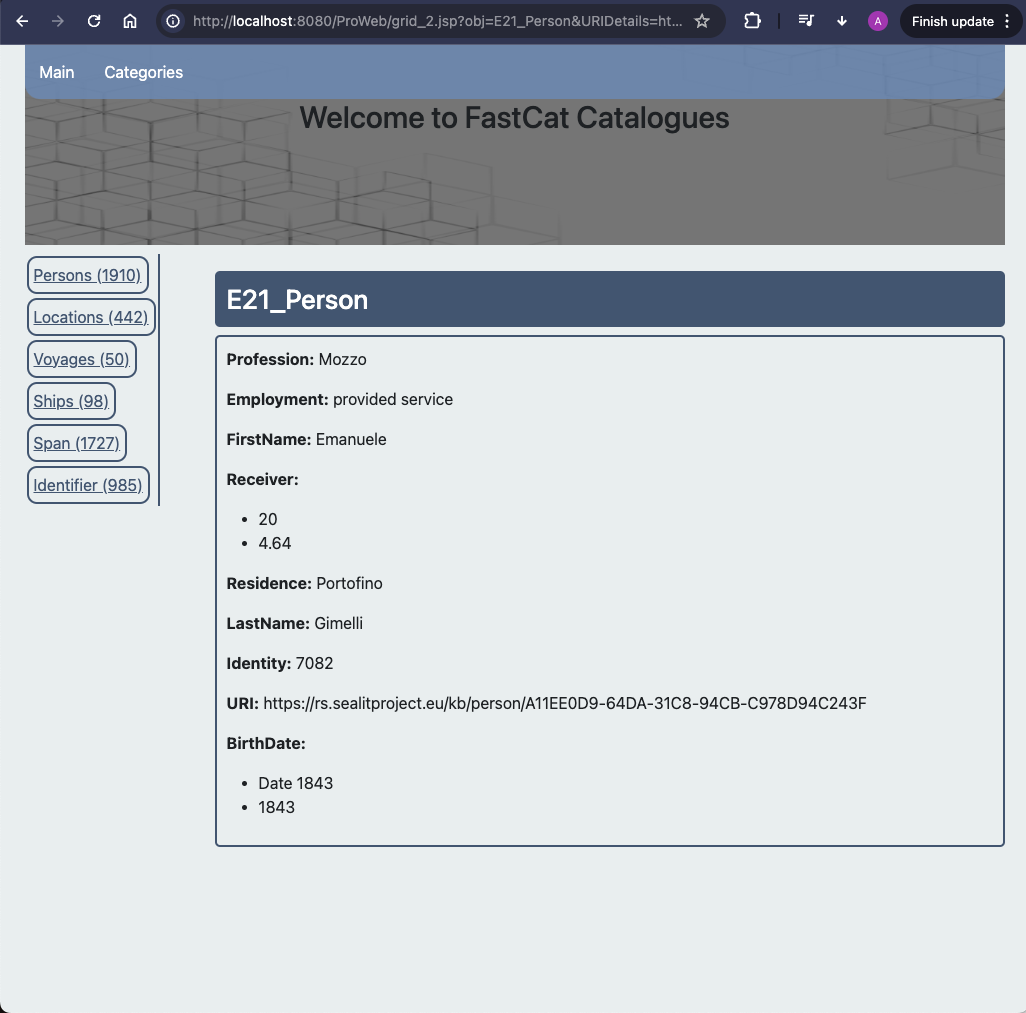
Όπως φαίνεται στην παρακάτω εικόνα, παίρνουμε την ίδια σελίδα με εκείνη του πίνακα, μαζί με την παράμετρο στο URL που περιλαμβάνει το URI του record(row) που επιλέχτηκε.



**Με την χρήση της παραμέτρου OPTIONAL:**

Επιλύεται το πρόβλημα και έχουμε τα επιθυμητά αποτελέσματα.

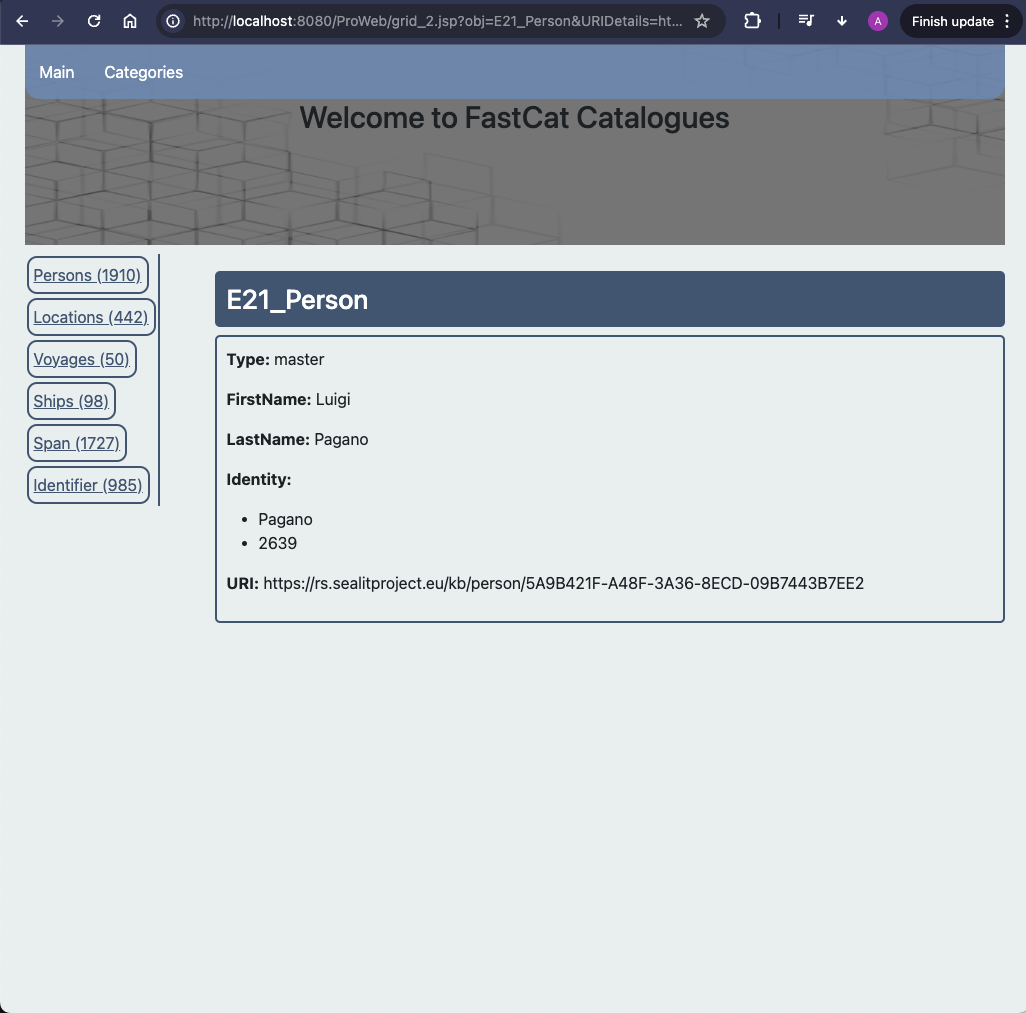
1η περίπτωση:



**ΠΡΕΠΕΙ ΝΑ διορθώσω το First\_Name**

Όπως φαίνεται στην εικόνα παραπάνω, υπάρχουν αρκετές πληροφορίες για την συγκεκριμένη καταχώρηση και όλες οι μεταβλητές έχουν τιμή.

2η περίπτωση:



Από την άλλη, σε αυτή την εικόνα, πεδία όπως BirthDate, Employment, Receiver, δεν έχουν τιμή, χωρίς ωστόσο να δημιουργείται το παραμικρό πρόβλημα, αφού όλα τα υπόλοιπα πεδία εμφανίζονται κανονικά.

# Απαιτήσεις Συστήματος

Το μηχάνημα στο οποίο έτρεχε η εφαρμογή, είχε τα παρακάτω χαρακτηριστικά:

|  |  |  |
| --- | --- | --- |
| Λειτουργικό Σύστημα | MacOS Sonoma 14.2.1 |  |
| Επεξεργαστής | Apple M1 3.2GHz |  |
| Μνήμη | 16 GB RAM |  |

# Troubleshooting

* + Read .trig extension files
  + Follow path to find the exact information about a column(label)
  + Put “OPTIONAL” (parameter??) in queries, so I didn’t lose any information  
    Some records may hadn’t some label.  
    In that case, without the OPTIONAL, the user couldn’t get all the other useful information
  + Πως να επιλέξουμε την γραμμή που θα εμφανίσει τα περισσότερα αποτελέσματα για κάθε row
  + Decoding - Encoding

# Επίλογος

## App Optimization - Improvements

* Δυνατότητα αποθήκευσης των δεδομένων που παρουσιάζονται
* Μετάβαση από μια κατηγορία σε μία άλλη, επιλέγοντας ένα συγκεκριμένο χαρακτηριστικό της συγκεκριμένης γραμμής(record)(row)

# Αναφορές / Πηγές

* Πληροφορίες από τις διαλέξεις του μαθήματος «ΗΥ-561 Διαχείριση Δεδομένων στον Παγκόσμιο Ιστό» μέσω της πλατφόρμας elearn.uoc <https://elearn.uoc.gr/user/view.php?id=13674&course=4510>
* <https://jena.apache.org/tutorials/rdf_api.html>

# Παραρτήματα

1. <https://www.ag-grid.com/javascript-data-grid/reference/>
2. <https://www.chartjs.org/docs/latest/getting-started/>
3. <https://getbootstrap.com/docs/5.3/getting-started/download/>