

Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Προπτυχιακών Σπουδών

«Πληροφοριακά Συστήματα»

**Προπτυχιακή Εργασία**

|  |  |
| --- | --- |
| Τίτλος Διατριβής | (Ελληνικά)  Δημιουργεία Υπηρεσιοστρεφούς εφαρμογής (SOA) client-server σε .ΝΕΤ  (Αγγλικά)  Implementation of a Service-Oriented Architecture client-server using .NET |
| Ονοματεπώνυμο Φοιτητή | ΣΤΑΜΑΤΙΟΣ ΛΙΑΤΣΟΣ |
| Πατρώνυμο | ΔΗΜΗΤΡΙΟΣ |
| Αριθμός Μητρώου | Π/06198 |
| Επιβλέπων | Χρήστος Δουληγέρης, Καθηγητής |

|  |  |
| --- | --- |
| Σαββάτο 10 | Φεβρουαρίου 2024 |

|  |
| --- |
| Τριμελής Εξεταστική Επιτροπή |

|  |  |  |
| --- | --- | --- |
| (υπογραφή) | (υπογραφή) | (υπογραφή) |
| Όνομα Επώνυμο Βαθμίδα | Όνομα Επώνυμο Βαθμίδα | Όνομα Επώνυμο Βαθμίδα |

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα διπλωματική εργασία εκπονήθηκε στα πλαίσια της ολοκλήρωσης των σπουδών μου στο Προπτυχιακό Πρόγραμμα Σπουδών του Τμήματος Πληροφορικής του Πανεπιστημίου Πειραιώς στα Πληροφοριακά Συστήματα.

Θα ήθελα να ευχαριστήσω τον καθηγητή μου κ. Χρήστο Δουληγέρη για το θέμα της παρούσας διπλωματικής εργασίας, την οποία μου προσέφεραν την δυνατότητα να εκπωνήσω.

Θα ήθελα επίσης να ευχαριστήσω τον καθηγητή Απόστολο Καράλη, του οποίου η βοηθεία ήταν σημαντική καθόλη την διάρκεια της ανάπτυξης και εκπόνησης της συγγεκριμένης εργασίας.

Τέλος, ευχαριστείες θα ήθελα να αποδώσω στους γονείς μου και στην γυναίκα μου, που με βοήθησαν με τον δικό τους τρόπο ο καθένας τους καθόλη την διάρκεια της εκπόνησης της εργασίας αυτής.

Περίληψη

Ο σκοπός της συγκεκριμένης διπλωματικής εργασίας είναι η ανάπτυξη ενός πληροφοριακού συστήματος, το οποίο ως στόχο έχει την διασύνδεση προμηθευτών οχημάτων (importers) με τις διάφορες αντιπροσωπείες αυτοκινήτων με σκοπό την παραγγελία νέων οχημάτων.

Κατά την υλοποίηση χρησιμοποιήθηκε το service oriented architecture (SOA) για την υλοποίηση των διάφορων υπηρεσιών (services), που το καθένα με την σειρά του έχει δημιουργηθεί αναλόγως με τον τομέα (domain) πάνω στο οποίο επιδρά. Ως εκ τουτου, με βάση το πεδίο στο οποίο επικεντρώθηκε η πτυχιακή (παραγγελίες οχημάτων) δημιουργήθηκαν τρεία διαφορετικές υπηρεσίες για την διαχείρηση των πελάτων/χρηστών, των οχημάτων/μοντέλων και των παραγγελιών.

Η επικοινωνία μεταξύ των υπηρεσιών πραγματοποιείτε ασύγχρονα, έτσι ώστε να υπάρχει μια χαλαρή σύνδεση (loosely coupled) μεταξύ των υπηρεσιών. Η διασύνδεση του γραφικού περιβάλλοντος (Graphical User Interface) με τις υπηρεσίες γίνεται σύγχρονα με την χρήση GRPC (Google Remote Procedure Calls).

Για την δημιουργεία του User Interface (UI) χρησιμοποιήθηκε η βιβλιοθήκη Blazor.NET. Το γραφικό περιβάλλον περιλαμβάνει το διαχηριστικό μέρος καθώς και το περιβάλλον διασύνδεσης του χρήστη με το σύστημα.

Abstract

The purpose of this specific thesis is the development of an information system, which aims to connect vehicle importers with various car dealerships for the purpose of ordering new vehicles.

During the implementation, service-oriented architecture (SOA) was used to implement various services, each created according to the specific domain it influences. Therefore, based on the focus of the thesis (vehicle orders), three different services were created for managing customers/users, vehicles/models, and orders.

The communication between services is asynchronous to ensure a loosely coupled connection between them. The graphical user interface (GUI) is connected to the services synchronously using GRPC (Google Remote Procedure Calls).

For the creation of the User Interface (UI), the Blazor.NET library was used. The graphical environment includes the management part as well as the user interface for interacting with the system.

Table of Contents

Type chapter title (level 1)1

Type chapter title (level 2)2

Type chapter title (level 3)3

Type chapter title (level 1)4

Type chapter title (level 2)5

Type chapter title (level 3)6

Εισαγωγή

Αντικείμενο Διπλωματικής

Στον κόσμο της σύγχρονης ανάπτυξης λογισμικού, η υιοθέτηση της Αρχιτεκτονικής Προσανατολισμένης προς τις Υπηρεσίες (Service-Oriented Architecture - SOA) έχει αναδειχθεί ως η προσέγγιση που ενισχύει την αρθρωτότητα (modularity), την επεκτασιμότητα (scalability) και τη βελτιωμένη ευελιξία του συστήματος (flexilibity). Αυτή η διατριβή εξετάζει την εφαρμογή των αρχών της SOA στη δημιουργία ενός πλήρους πληροφοριακού συστήματος. Κεντρικό σημείο αυτής της προσπάθειας είναι η δημιουργία τριών διακριτικών μικρουπηρεσιών - των μικρουπηρεσιών Οχήματος, Πελάτη και Παραγγελιών - σχεδιασμένων να αλληλεπιδρούν απροβλημάτιστα μέσα σε ένα δυναμικό οικοσύστημα.

Χρησιμοποιώντας το Kafka για ασύγχρονη επικοινωνία εξασφαλίζουμε την αποτελεσματική και ανθεκτική ανταλλαγή δεδομένων. Αυτή η αρχιτεκτονική όχι μόνο προάγει την αυτονομία των μεμονωμένων υπηρεσιών, αλλά θεμελιώνει επίσης μια βάση για την επεκτασιμότητα και την προσαρμοστικότητα σε αντίδραση στις εξελισσόμενες επιχειρηματικές ανάγκες.

Το μέρος του χρήστη (frontend) του συστήματος δημιουργήθηκε χρησιμοποιώντας το Blazor, ένα πλαίσιο που διευκολύνει την ανάπτυξη διαδραστικών και δυναμικών εφαρμογών ιστού. Αυτή η διεπαφή χρήστη όχι μόνο περιλαμβάνει τις διαχειριστικές πτυχές αλλά παρέχει επίσης ένα φιλικό προς τον χρήστη περιβάλλον για την αλληλεπίδραση με το σύστημα.

Επιπλέον, αυτή η διατριβή εξερευνά την ολοκλήρωση του αναπτυγμένου συστήματος με το Microsoft Business Central ERP. Συγκεκριμένα, δημιουργείται μια ευέλικτη σύνδεση για τη διαχείριση των παραγγελιών που προκύπτουν από τη μικρουπηρεσία Παραγγελιών. Αυτή η συνεργασία με το Microsoft Business Central ERP ενισχύει τη συνολική αποτελεσματικότητα και συνοχή της διαχείρισης παραγγελιών, προσθέτοντας ένα επίπεδο εξελιγμένης λειτουργικότητας στο πληροφοριακό σύστημα.

Καθώς πλοηγούμαστε μέσα από τις λεπτομέρειες αυτών των αλληλοσυνδεδεμένων συστατικών, αυτή η διατριβή στοχεύει στον εκτυπωσιακό εξηγηματισμό των αρχιτεκτονικών αποφάσεων, των προκλήσεων υλοποίησης και των πρακτικών οφελών που αποκομίζονται από την συνετή εφαρμογή της Αρχιτεκτονικής Προσανατολισμένης προς τις Υπηρεσίες σε αυτό το πολυπλοκό σύστημα.

Οργάνωση Κειμένου

1. Ανάλυση Απαιτήσεων Συστήματος

Παρακάτω θα ακολουθήσει μια περιγραφή της αρχιτεκτονικής του συστήματος, ενώ στην συνέχεια θα προχωρήσουμε στην ανάλυση απαιτήσεων για τις λειτουργίες τις οποίες το σύστημα αυτό παρέχει.

1. Αρχιτεκτονική

Το σύστημα αυτό θα έχει ώς στόχο να επιδείξει την διασύνδεση μεταξύ διαφόρων υπηρεσιών χρησιμοποιώντας την Αρχιτεκτονική Προσανατολισμένη προς τις Υπηρεσίες (Service – Oriented Architecture) με ασφαλή διασύνδεση μεταξύ των συστημάτων. Για τον λόγο αυτό κατασκευάστηκαν κάποιες υπηρεσίες οι οποίες οι οποίες αλληλεπιδρούν είτε σύγχρονα είτε ασύγχρονα ανάλογα με το πόσο γρήγορος θέλουμε να είναι ο χρόνος απόκρισης κάθε υπηρεσίας.

Η πρώτη υπηρεσία έχει να κάνει με την διαχείρηση των χρηστών του συστήματος καθώς και των πελατών. Υπάρχει μία σχέση εξάρτησης μεταξύ χρήστη και πελάτη. Ο διαχειρηστής του συστήματος μπορεί να δημιουργήσει έναν χρήστη και στην συνέχεια απο αυτόν τον χρήστη να δημιουργήσει έναν πελάτη. Ο χρήστης, κατα την ώρα της δημιουργείας του, δημιουργείται και σαν εγγραφή μέσα στο σύστημα που χρησιμοποιούμε για την αυθεντικοποίηση και την εξουσιοδότηση των χρηστών.

Η δεύτερη υπηρεσία αναλαμβάνει να διαχειρηστεί τις οντότητες οι οποιές είναι σχετικές με τις πληροφορίες των οχημάτων που θέλουμε να αποθηκεύσουμε στην βάση μας. Σε αυτό το σημείο ο διαχειριστής του συστήματος μπορεί να δημιουργήσει τα μοντέλα και τις μάρκες των αυτοκινήτων που θα χρησιμοποιηθούν σε δεύτερο χρόνο κατα την δημιουργεία και εξυπηρέτηση των παραγγελιών απο τους χρήστες.

Η τρίτη υπηρεσία σχετίζεται με την διαχείρηση των παραγγελιών. Ο χρήστης μπορεί να δημιουργήσει μια νέα παραγγελία, η οποία με την σειρά της αποθηκεύται μέσα στο σύστημα ουράς μηνυμάτων. Αυτό το μήνυμα παραλαμβάνεται απο το ERP στο οποίο και δημιουργείται μία προσφορά. Ο χρήστης του ERP παραλαμβάνει αυτόματα την προσφορά στο σύστημα μέσω μιας περιοδικής διεργασίας που εκτελείται στο παρασκήνιο. Έπειτα μπορεί να μετατρέψει την προσφορά αυτή σε παραγγελία και έπειτα να τιμολογήσει την παραγγελία αυτή. Κατα την τιμολόγηση ο μήνυμα αποστέλλεται πάλι στην ουρά μηνυμάτων και παραλαμβάνεται απο την υπηρεσία διαχείρησης παραγγελιών. Όπου και δημιουργούνται οι αντίστοιχες εγγραφές μέσα στην βάση δεδομένων.

Ο συνδετικός κρίκος όλων των παραπάνω υπηρεσιών, είναι το γραφικό περιβάλλον χρήστη το οποίο παρέχει την δυνατότητα τόσο στους διαχειρηστές όσο και στους χρήστες να εκτελέσουν τις διάφορες ενέργειες τις οποίες παρέχει το σύστημα.

Τέλος, τα σύστηματα αυτά πρέπει να παρέχουν ένα βαθμό ασφάλειας έτσι ώστε κάποιος εξωτερικός ή κακόβουλος χρήστης να μην μπορεί να έχει πρόσβαση στα δεδομένα του συστήματος. Για τον λόγο αυτό, όλες οι επικοινωνίες είναι κρυπτογραφημένες χρημοποιώντας το πρωτόκολο TLS (Transport Layer Security). Πέρα όμως απο την ασφαλή επικοινωνία κάθε χρήστης ανάλογο με την βαθμίδα που έχει μέσα στο σύστημα θα πρέπει να μπορεί να εκτελέι συγκεκριμένες λειτουργίες. Αυτό επιτυγχάνεται με την χρήση μιας υπηρεσίας διαχείρησης ταυτότητας και πρόσβασης (Keycloak).

A screenshot of a computer

Description automatically generated

**Εικόνα 1 – Αρχιτεκτονική**

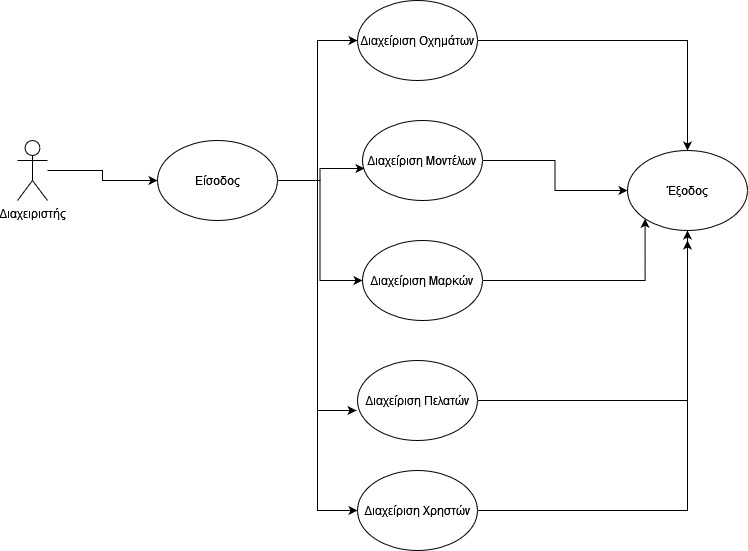
Κάθε χρήστης αφού συνδεθεί με το σύστημα, θα λάβει ένα διαπιστευτήριο (access token) το με το οποίο θα αποφασίζεται απο το σύστημα αν ο συγκεκριμένος χρήστης έχει πρόσβαση στις συγκεκριμένες μεθόδους του συστήματος.

Περιγραφή Λειτουργιών

Στο κεφάλαιο αυτό θα αναφερθούμε στην περιγραφή των λειτουργιών του συστήματος που καλείται να εκτελέσει ώστε να είναι χρήσιμο και ορθά κατασκευασμένο. Όπως ήδη έχουμε αναφέρει το σύστημα χωρίζεται σε τρεία επιμέρους κομμάτια, στην σελίδα που χειρίζεται ο διαχειριστής του συστήματος, την σελίδα με την οποία αλληλεπιδρούν οι χρήστες και το ERP στο οποίο γίνεται η ολοκλήρωση και καταχώρηση των παραγγελιών.

* + 1. Υποσύστημα Διαχειριστή

Στο υποσύστημα του Διαχειριστή, ο χρήστης θα έχει την δυνατότητα να δημιουργεί, να ανανεώνει και να διαγράφει νέους χρήστες, πελάτες, μοντέλα και μάρκες οχημάτων καθώς και τα ίδια τα οχήματα.

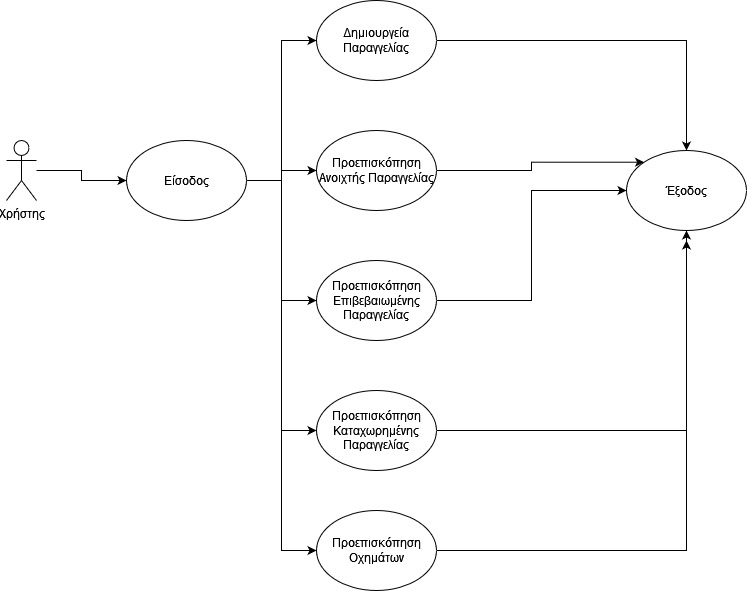


**Εικόνα 2 - Διάγραμμα Περιπτώσεων Χρήσης Διαχειριστή**

Στο παραπάνω διάγραμμα φαίνονται οι δυνατότητες που προσφέρει η σελίδα στον διαχειριστή του συστήματος. Ο διαχειριστής μπορεί να δημιουργήσει, να τροποποιήσει ή να διαγράψει όποια απο τις οντότητες διαχειρίζεται το σύστημά. Η δημιουργεία νέων χρηστών δίνει την δυνατότητα σε νέους χρήστες να εισέλθουν στο σύστημα αλλά μόνο ένας χρήστης με δικαιώματα διαχειριστή μπορεί να δημιουργηθεί και αυτός δημιουργείται αυτόματα κατα την αρχικοποίηση του συστήματος. Πολλές απο τις οντότητες είναι συνδεδεμένες μεταξύ τους, οπότε και ο διαχειριστής δεν μπορεί να προβεί στην δημιουργεία, για παράδειγμα, ενός μοντέλου πριν δημιουργήσει την αντίστοιχη μάρκα.

* + 1. Υποσύστημα Χρήστη

Στο υποσύστημα του χρήστη, ο εκάστωτε χρήστης που έχει συνδεθεί στην σελιδά θα μπορεί να δημιουργήσει νέες παραγγελίες αυτοκινήτων καθώς και να κάνει προεπισκόπηση αυτών.



**Εικόνα 3 - Διάγραμμα Περιπτώσεων Χρήσης Χρήστη**

Στο παραπάνω διάγραμμα χρήσης αποκονίζονται οι διάφορες λειτουργίες στις οποίες μπορεί να προβεί ο χρήστης της εφαρμογής. Έτσι σαν αρχική διαδικασία ο χρήστης έχει την επιλογή να ξεκινήσει μια παραγγελία ενός οχήματος. Στην συνέχεια μπορεί να ανατρέχει στις ανοιχτές παραγγελίες, στις παραγγελίες οι οποίες έχουν επιβεβαιωθεί, καθώς και στις παραγγελίες οι οποίες έχουν φτάσει σε επίπεδο τιμολόγησης και είναι πλέον κατοχυρωμένες. Κάθε κατοχυρωμένη παραγγελία δημιουργεί και ένα όχημα, οπότε ο χρήστης μπορεί να δει και τα οχήματα που έχει στην διάθεση του.

1. Μοντέλο Οντοτήτων Συσχετίσεων

Στο κεφάλαιο αυτό παραθέτουμε το σχήμα της βάσης δεδομένων που χρησιμοποιεί η κάθε υπηρεσία για να αποθηκεύση τα δεδομένα της, η απεικόνηση χρησιμοποιεί το μοντέλο Οντοτήτων Συσχετίσεων για να αποτυπώσει την εξάρτηση και την δομή της βάσης.

**A diagram of a software application

Description automatically generated with medium confidence**

**Εικόνα 4 – Διάγραμμα Υπηρεσίας Οχημάτων**

A screenshot of a computer

Description automatically generated

Εικόνα 5 - Διάγραμμα Υπηρεσίας Χρηστών/Πελάτων

A diagram of a data flow

Description automatically generated with medium confidence

Εικόνα 6 - Διάγραμμα Υπηρεσίας Παραγγελειών

Όπως φαίνεται απο τα παραπάνω διαγράμμα, κάθε υπηρεσία έχει και την δική της βάση, με το δικό της ξεχωριστό σχήμα αναλόγος με τα δεδομένα που καλείται να διαχειριστεί. Υπάρχει μια σύνδεση μεταξύ των δεδομένων των διαφορετικών βάσεων με στόχο να μπορούν να ανακτηθούν τα συσχετιζόμενα δεδομένα και να εμφανιστούν στην γραφική διεπαφή των χρηστών.

1. Επιμέρους Συστήματα

Στην ενότητα αυτή θα αναπτύξουμε την λειτουργικότητα και χρησιμότητα στην υλοποιήση, των επιμέρουν υπηρεσιών που χρησιμοποιήθηκαν κατά την υλοποίηση. Αυτές οι υπηρεσίες είναι τα:

* **Kafka**: Το Apache Kafka είναι μια πλατφόρμα κατανεμημένου συστήματος ροής συμβάντων ανοικτού κώδικα, που χρησιμοποιείται για τη δημιουργία πραγματικού χρόνου αγωγών δεδομένων και εφαρμογών ροής. Αρχικά αναπτύχθηκε από το LinkedIn και αργότερα δόθηκε στη δημοσιότητα ως μέρος του Apache Software Foundation. Το Kafka παρέχει μια ανθεκτική σε βλάβες και επεκτάσιμη λύση για τη διαχείριση μεγάλων όγκων δεδομένων σε πραγματικό χρόνο.
* **Kong**: Το Kong είναι ένα API gateway ανοικτού κώδικα και ένα επίπεδο διαχείρισης των υπηρεσιών. Δρα ως κεντρική πλατφόρμα για τη διαχείριση, την ασφάλεια και την επέκταση των API (Διεπαφών Προγραμματισμού Εφαρμογών) σε διάφορες υπηρεσίες ή εφαρμογές. Το Kong σχεδιάστηκε για να απλοποιήσει τις πολυπλοκότητες που σχετίζονται με τη διαχείριση των API, παρέχοντας χαρακτηριστικά όπως αυθεντικοποίηση, εξουσιοδότηση, περιορισμό ρυθμού, καταγραφή και άλλα.
* **Redis**: Είναι μια online προσωρινή-κρυφή μνήμη την οποία μπορούν να μοιραστούν οι διάφορες υπηρεσίες μεταξύ τους, χωρίς να κάνει η κάθε μια διαχείριση της μνήμης αυτής ξεχωριστά. Αυτό δίνει το πλεονέκτημα ότι έαν μια υπηρεσία χρειαστεί να επανεκκινήσει θα έχει στην διαθεσή της δεδομένα αυτής της μνήμης.
* **Keycloak**: Είναι η υπηρεσία που παρέχει στα προγραμματά μας την δυνατότητα για αυθεντικοποίηση και εξουσιοδότηση. Σαν υπηρεσία είναι επεκτάσιμη και διαθέτει ακόμα και υπηρεσίες αυθεντικοποίησης μέσω facebook/google etc.
* **ElasticSearch**: Το Elasticsearch είναι ένα ανοικτού κώδικα, κατανεμημένο σύστημα αναζήτησης που βασίζεται στο Apache Lucene. Σχεδιάστηκε για να παρέχει μια εμπειρία αναζήτησης κλιμακούμενη και υψηλής απόδοσης για διάφορους τύπους δεδομένων, από κείμενα έως δομημένα δεδομένα και πληροφορίες γεωχωρικής τοποθεσίας. Το Elasticsearch αποτελεί μέρος της Elastic Stack (προηγουμένως ELK Stack), που περιλαμβάνει εργαλεία όπως το Kibana, τα Beats και το Logstash.
* **KafkaConnect**: Το Kafka Connect είναι ένα σύνολο διεπαφών εντός του Apache Kafka που απλοποιεί την ενσωμάτωση του Kafka με εξωτερικά συστήματα. Παρέχει έναν κλιμακούμενο και αξιόπιστο τρόπο σύνδεσης του Kafka με διάφορες πηγές και απορροφητές δεδομένων, επιτρέποντας την άρρηκτη μεταφορά δεδομένων μεταξύ θεμάτων Kafka και εξωτερικών συστημάτων.
* Kibana: Το Kibana είναι ένα εργαλείο ανοικτού κώδικα για την απεικόνιση και την εξερεύνηση δεδομένων, σχεδιασμένο να λειτουργεί με την πλατφόρμα Elasticsearch. Αποτελεί μέρος της Elastic Stack (προηγουμένως ELK Stack), η οποία περιλαμβάνει επίσης το Elasticsearch και το Logstash. Το Kibana επιτρέπει στους χρήστες να αλληλεπιδρούν με και να απεικονίζουν δεδομένα που αποθηκεύονται στο Elasticsearch, καθιστώντας πιο εύκολο τον κατανοητό και αναλυτικό χειρισμό μεγάλων όγκων δεδομένων.

Σύγκριση gRPC με REST

Μια «Διεπαφή Προγραμματισμού Εφαρμογών» αποτελείται απο ένα σύνολο κανόνων και προτοκόλων τα οποία επιτρέπουν σε ένα σύνολο εφαρμογών να αλληλεπιδρούν και να ανταλλάσουν δεδομένα.

Υπάρχουν πολλές διαφορετικές αρχιτεκτονικές για την δημιουργεία μιας τέτοια διεπαφής και κάθε μία απο αυτές έχει τα δικά της πλεονεκτήματα και μειονεκτήματα. Δύο από αυτές τις αρχιτεκτονικές είναι το REST και το gRPC. Το REST ορίζει ένα σύνολο από κανόνες για την δημιουργεία εφαρμογών και βασίζεται στην απλότητα, την μη διατήρηση κατάστασης (stateless) καθώς και στην επικοινωνία με βάση τους πόρους (resources). Απο την άλλη, το gRPC, είναι ένα σύστημα υψηλής απόδοσης για απομακρησμένη κλήση διαδικασιών (RPC) το οποίο χρησιμοποιεί τα Protocol Buffers και το HTTP/2 για την αποτελεσματική επικοινωνία σε κατανεμημένα συστήματα.

Το REST είναι ο πιο διαδεδομένος τρόπος για την δημιουργεία «Διεπαφών Προγραμματισμού Εφαρμογών» αλλά, με την σειρά του, το gRPC έχει σημειώσει σημαντική άνοδο λόγο της υποστήριξής του για ροές δεδομένων διπλή κατεύθυνσης (bidirectional streaming), της ισχυρούς αποτύπωσης των τύπων δεδομένων (strong data typing) και της δημιουργίας κώδικα ανεξαρτήτους γλώσσας προγραμματισμού.

Τι είναι το REST

Το REST (Representation State Transfer), είναι ο πιο διαδεδομένος τρόπος για την δημιουργεία «Διεπαφών Προγραμματισμού Εφαρμογών». Σε μία τέτοια εφαρμογή, οι πόροι προς διαχείρηση , αναγνωρίζονται απο το «Εννιαίο Αναγνωριστικό Πόρων» (Uniform Resource Identifier – URI) και οι διάφορες λειτουργίες προς τους πόρους αυτούς γίνονται χρησιμοποιώντας ένα σύνολο μεθόδων του HTTP προτοκόλου. Οι πόροι αναπαρίστανται σε μορφή JSON, η οποία μεταφέρεται μεταξύ του πελάτη και του διακομιστή στα σώματα των αιτημάτων και των απαντήσεων του HTTP.

Τι είναι το gRPC

Το gRPC είναι ένα ανοιχτού κώδικα, υψηλής απόδοσης πλαίσιο το οποίο παρέχει την δύνατότητα σε κατανεμημένα συστήματα να επικοινωνούν μεταξύ τους με τον βέλτιστο τρόπο. Παρέχει μια υλοποίηση του RPC πρωτοκόλου, το οποίο δίνει την δυνατότητα σε εφαρμογές να καλούν απομακρισμένες μεθόδους σαν να ήταν μέρος της υλοποίησής τους.

Το πρωτόκολο αυτό, δημιουργήθηκε απο την Google το 2015 και περιλαμβάνει πολλές βελτιστοποιήσεις προς τον τρόπο με τον οποίο γίνονται οι απομακρισμένες κλήσεις. Για παράδειγμα, κάνει χρήση των Protocol Buffers (Protobuf) το οποίο παρέχει ισχυρή αποτύπωση στους τύπους δεδομένων, σειριοποίηση δεδομένων και παραγωγή κώδικα σε ποικίλες γλώσσες προγραμματισμού.

* 1. Αρχιτεκτονική

Το σύστημα έχει ως στόχο την αποτύπωση της διασύνδεσης των επιμέρους υπηρεσιών με διάφορους τρόπους ανάλογα με την αναγκαιότητα για σύγχρονη ή ασύγχρονη επικοινωνία. Οι διάφορες υπηρεσίες επικοινωνούν μεταξύ τους, ασύγχρονα, με την χρήση κάποια ουράς μυνημάτων (Kafka) ενώ για την διεπαφή του χρήστη (frontend) χρησιμοποιύμε σύγχρονα μυνήματα επικοινωνίας με το προτόκολλο gRPC.

Για την επικοινωνία με την ουρά μυνημάτων (Kafka) έχουμε δύο διαφορετικά μέρη που επικοινωνούν μεταξύ τους, τον Publisher και τον Consumer. Κάθε Publisher δημιουργεί ένα μύνημα μέσα στην ουρά σε ένα συγκεκριμένο θέμα (Topic). Στην συνέχεια ένας οι περισσότεροι Consumers εγγράφονται στο συγκεκριμένο θέμα και κάθε φορά που υπάρχει ένα νέο μύνημα στην ουρά το λαμβάνουν και το επεργάζονται.

A screenshot of a computer

Description automatically generated

**Εικόνα 7 - Θέματα Ουράς Μηνυμάτων**

Για την σύγχρονη επικοινωνία έχουμε χρησιμοποιήσει ένα reverse proxy (kong) ο οποίος αναλαμβάνει να κάνει ανακατεύθυνση των μυνημάτων προς την αντίστοιχη υπηρεσία (service).

Η ανακατεύθυνση αυτή γίνεται με βάση το url του endpoint του οποίου καλείται. Κάθε service κατα την εκκίνησή του δημιουργεί μια εγγραφή, στον reverse proxy χρησιμοποιώντας μια Διεπαφή Προγραμματισμού Εφαρμογών (API), για κάθε controller που έχουμε δημιουργήσει.

Στην συνέχεια, όταν κάποιο αίτημα (request) σταλθεί στον reverse proxy αυτός είναι υπεύθυνος ώστε να αποστείλει το μήνυμα (dispatch) στην αντίστοιχη υπηρεσία που το εξυπηρετεί.

* Το κείμενο να είναι γραμμένο σε font size 10pt με μονό διάστιχο (single spacing) και απόσταση παραγράφων 3pt (after).
* Η πρώτη παράγραφος κάθε ενότητας να μην έχει εσοχή πρώτης γραμμής ενώ οι επόμενες να έχουν εσοχή πρώτης γραμμής.
* Τα περιθώρια σελίδας να είναι 3cm και στις τέσσερεις πλευρές (πάνω, κάτω, αριστερά, δεξιά).
* Τα Headings να είναι όλα με font Arial Black και όχι Bold. Το Heading 1 να είναι 12pt, το Heading 2 να είναι 11pt, το Heading 3 να είναι 10pt. Να μην χρησιμοποιείτε Heading 4 και πέρα.
* Να μην αφήνετε κενές γραμμές πριν ή μετά από τα headings και κάθε επίπεδο heading να απέχει 18pt before και 6pt after.
* Οι λεζάντες (captions) στα σχήματα και τους πίνακες να είναι αριστερά στοιχισμένες και να είναι **Arial bold 9pt**.
* Σε κάθε σελίδα να υπάρχει footer (Arial 8pt) με τον τίτλο της διατριβής στα αριστερά. Στο footer επίσης να υπάρχει αρίθμηση σελίδας στα δεξιά και πάλι με font Arial 8pt.
* Σε κάθε σελίδα να υπάρχει header (Arial 8pt) με το όνομα του φοιτητή στα δεξιά και το λεκτικό «Μεταπτυχιακή Διατριβή» στα αριστερά.
* Τα header και footer να απέχουν από τα άκρα του χαρτιού 2.5cm (στο Page Setup).
* Η διατριβή να περιέχει απαραίτητα:
  + Περίληψη (Abstract) σε χωριστή σελίδα (μισή σελίδα Ελληνικά και μισή στα Αγγλικά).
  + Εισαγωγή – Σύντομη Περιγραφή Προβλήματος/Αντικειμένου (μέχρι 3 σελίδες).
  + ....
  + Συμπεράσματα – Περίληψη
  + Βιβλιογραφία