

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**

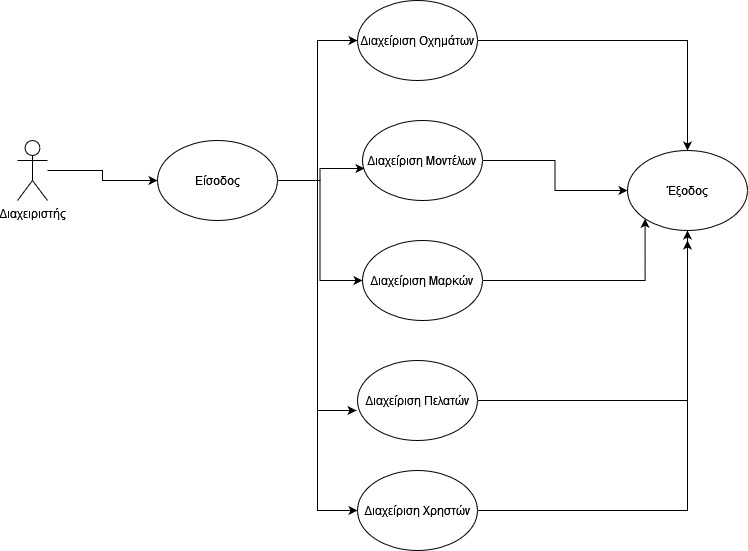
Κάθε χρήστης αφού συνδεθεί με το σύστημα, θα λάβει ένα διαπιστευτήριο (access token) το με το οποίο θα αποφασίζεται απο το σύστημα αν ο συγκεκριμένος χρήστης έχει πρόσβαση στις συγκεκριμένες μεθόδους του συστήματος.

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.

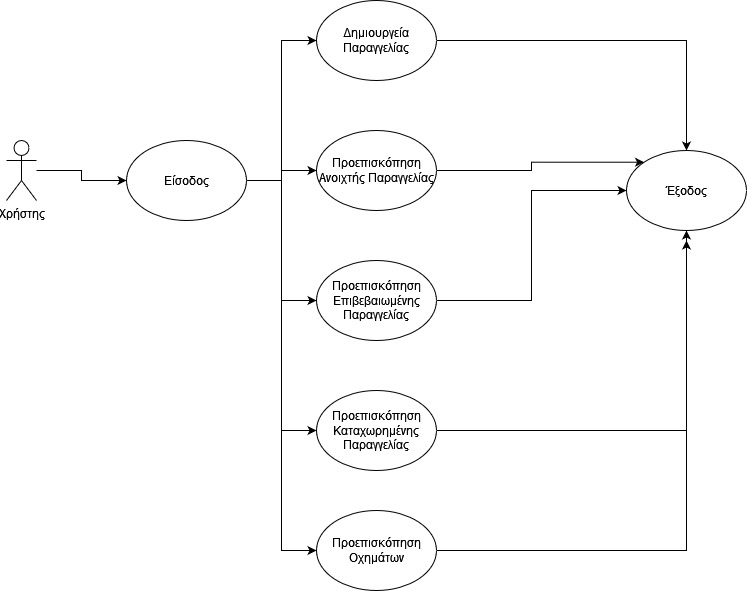


**Εικόνα 2 - Διάγραμμα Περιπτώσεων Χρήσης Διαχειριστή**

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.

Πως λειτουργεί το REST

Το REST (Reprensentational State Transfer) περιγράφει μια αρχιτεκτονική πελάτη-εξυπηρετήτη, κατα την οποία τα δεδομένα μεταφέρονται στην εφαρμογή πελάτη δομημένα μέσω μηνυμάτων JSON ή ακόμα και xml.

Ένα REST σύστημα, διαθέτει τις παρακάτω ιδιότητες:

* Προγραμματιστική διεπαφή (API) : Η διεπαφή αυτή διαθέτει ορισμένους πόρους (resources) προς του καταναλωτές (clients).
* Ανεξαρτησία πελάτη/εξυπηρετητή: Ο πελάτης και ο εξυπηρετητής δρούν αυτόνομα. Η εφαρμογή πελάτη γνωρίζει μόνο συγκεκριμένα αναγνωριστικά πόρων (URIs) τα οποία διατίθονται απο την εφαρμογή του εξυπηρετητή.
* Stateless: Η εφαρμογή του εξυπηρετητή δεν αποθηκευεί την κατάσταση ενός αιτήματος απο την εφαρμογή πελάτη. Κάθε άιτημα προς την εφαρμογή του εξυπηρετητή επιδρά σε μόνο ένα πόρο και κάθε επιμέρους αίτημα είναι αυτοτελές.
* Πολυεπίπεδο (Layered) : Η αρχιτεκτονική αυτή είναι πολυεπίπεδη, το οποίο της δίνει την δυνατότητα τα διαφορετικά μέρη της να υλοποιούνται απο διαφορετικές υλοποιήσεις εξυπηρετητών.
* Κώδικας κατά απαίτηση (Code on Demand) : Δίνει την δυνατότητα στην εφαρμογή του εξυπηρετητή να στείλει κομμάτια κώδικα στην εφαρμογή πελάτη, ώστε η πρώτη να καθορίσει τον τρόπο και την ροή εκτέλεσης.

Πως λειτουργεί το Grpc

Το gRPC χρησιμοποιεί τη μορφοποίηση μηνυμάτων Protobuf αντί των JSON που συνηθίζονται στα REST APIs και RPC APIs. Η χρήση του Protobuf βοηθά στην αντιμετώπιση θεμάτων ταχύτητας και βάρους, προσφέροντας μεγαλύτερη αποδοτικότητα στην αποστολή μηνυμάτων. Είναι ανεξάρτητο πλατφόρμας και γλώσσας όπως το JSON, αλλά επικοινωνεί μεταξύ τους μέσω δυαδικής σειριοποίησης.

Το gRPC χρησιμοποιεί το πρωτόκολλο HTTP/2 αντί του HTTP/1.1 που χρησιμοποιείται συνήθως στα συστήματα REST. Το HTTP/2 είναι πιο γρήγορο, αποδοτικό και μειώνει την καθυστέρηση του δικτύου μέσω του multiplexing. Το gRPC προσφέρει τρεις τύπους ροών: εξυπηρετητή, πελάτη και αμφίδρομες ροές μηνυμάτων.

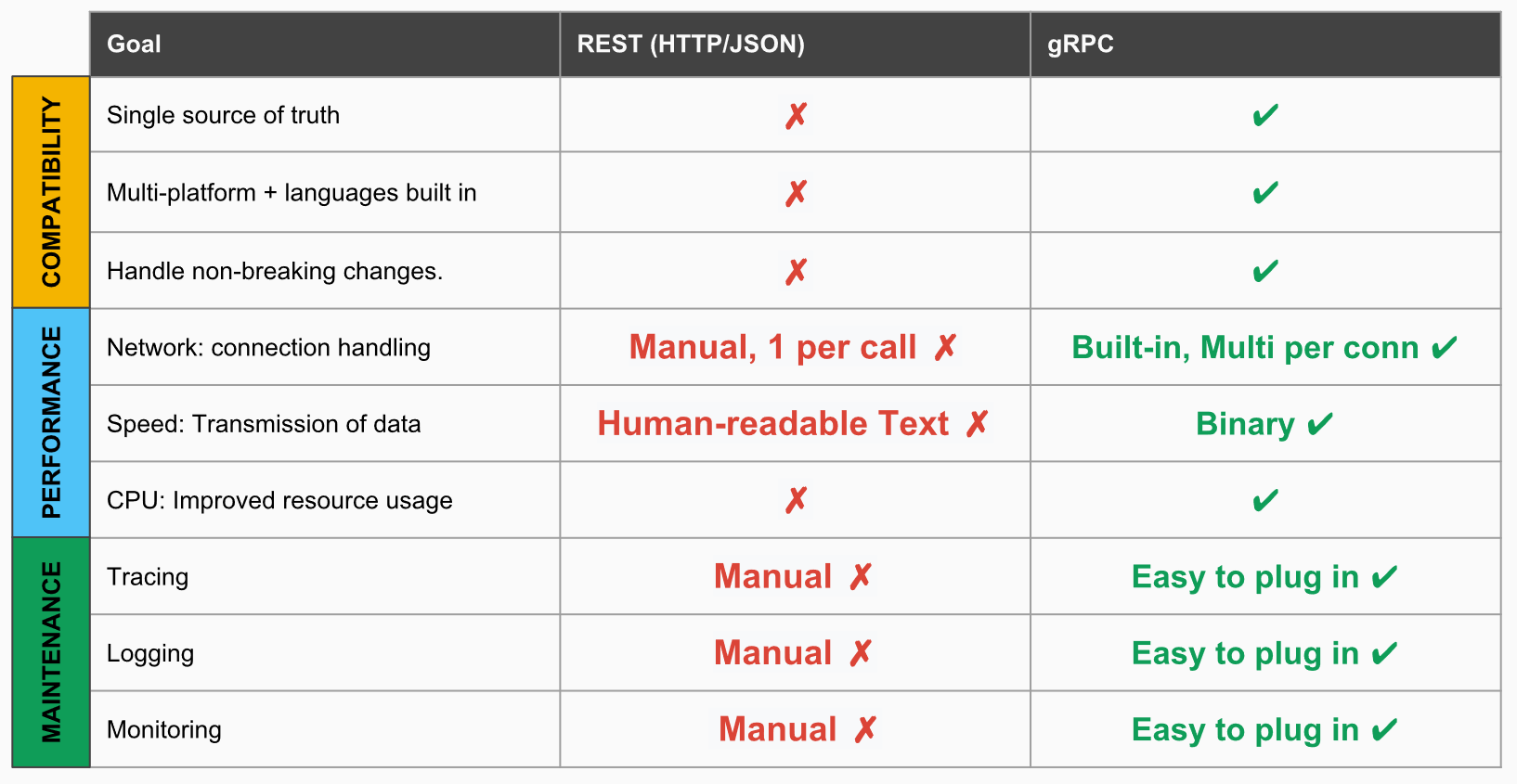
Σε αντίθεση με τα REST APIs που συνήθως χρησιμοποιούν εργαλεία τρίτων, το gRPC παρέχει ενσωματωμένη γεννήτρια κώδικα (Protoc compiler). Αυτό επιτρέπει τη δημιουργία κώδικα σε ποικίλες γλώσσες και χρησιμοποιείται σε πολύγλωσσα περιβάλλοντα, όπου οι υπηρεσίες τρέχουν σε ξεχωριστές πλατφόρμες και προγραμματίζονται σε διαφορετικές γλώσσες πρόγραμματισμού.

Τα κύρια χαρακτηριστικά του είναι τα εξής:

* Ανεξάρτητο από πλατφόρμες και γλώσσες προγραμματισμού.
* Χρησιμοποιεί δυαδική απεικόνιση για την αναπαράσταση των μηνυμάτων.
* Αυξάνει την ταχύτητα μετάδοσης των μηνυμάτων λόγο της δομής δομής των μηνυμάτων.
* Κάνει χρήση του HTTP/2 πρωτοκόλου.
* Παρέχει μηχανισμό για διαχείρηση των διαφορετικών εκδόσεων των μηνυμάτων.

Συμπεράσματα

Για εφαρμογές, μεγάλου σκέλους, οι οποίες καλούνται να εξυπηρετούν πολλά αιτήματα σε σύντομο χρονικό διάστημα και οι οποίες είναι δημιουργημένες, βασιζόμενες πάνω στην αρχιτεκτονική υπηρεσιοστρεφούς εφαρμογής η χρήση gRPC δείχνει να είναι μια καλύτερη λύση σε σχέση με την χρήση του REST πρωτοκόλου.



**Εικόνα 7 - Σύγκριση gRPC – REST**

Σε γενικές γραμμές οι κύριες διαφορές, οι οποίες καθιστούν το gRPC μια καλύτερη επιλογή, είναι ότι το gRPC παρέχει την δυνατότητα με την ίδια δομή μηνύματος να δημιουργηθούν σε διάφορες γλώσσες προγραμματισμού εφαρμογές, τόσο πελάτη όσο και εξυπηρετητή.

Παρέχει την δυνατότητα παράλληλης διαχείρησης των μηνυμάτων, λόγο της χρήσης του HTTP/2 καθώς και την ταχύτατη μεταφορά των μηνυμάτων λόγο της δυαδικής αναπαράστασης των μηνυμάτων. Σαν αντίκτυπο, η δυαδική αναπαράσταση, προσφέρει και την καλύτερη διαχείρηση των πόρων του συστήματος.

* 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 Εκδότη μηνυμάτων (Publisher) και τον Καταναλωτή αυτών (Consumer). Κάθε εκδότης δημιουργεί ένα μύνημα μέσα στην ουρά σε ένα συγκεκριμένο θέμα (Topic). Στην συνέχεια ένας οι περισσότεροι καταναλωτές 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 8 - 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 ενοποιημένο αναγνωριστικό πόρου του σημείου πρόσβασης 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.

Τα δεδομένα τα οποία ανταλλάσονται μεταξύ των συστημάτων, αποθηκεύονται μέσα σε μία βάση δεδομένων (postgres). Κάθε υπηρεσία διαθέτει την δική της βάση δεδομένων, έτσι ώστε να διασφαλίσουμε την ακεραιότητα τους σε περίπτωση που κάποια απο τις βάσεις καταστεί μη λειτουργική για κάποιο χρονικό διάστημα.

Για την καταγραφή των συμβάντων του συστήματος, έτσι ώστε να μπορούν να αποτυπωθούν ενδεχόμενα προβλήματα έχουμε χρησιμοποιήσει μια παραλλαγή του ELK stack (Elasticsearch, Logstash και Kibana).

Το Logstash χρησιμοποιείτε για την αποθήκευση και καταγραφή των συμβάντων των συστημάτων. Τα μηνύματα αυτά στην συνέχεια μεταφέρονται στο elasticsearch έτσι ώστε να δημιουργηθεί ένα ευρετήριο αναζήτησης, ώστε να είναι ταχύτερη η αναζήτηση των συμβάντων αυτών.

Στην συνέχεια χρησιμοποιώντας το Kibana μπορούμε να ανατρέξουμε στα καταγεγραμμένα συμβάντα και να έχουμε μια πιο εύχρηστη αποτύπωση αυτών. Με την χρήση φίλτρων μπορούμε να περιορίσουμε τα αποτελέσματα και σε περίπτωση προβλήματος να εντοπίσουμε την αιτία που το πυροδότησε.

A screenshot of a computer

Description automatically generated

Εικόνα 9 - Kibana

Χρήση Εφαρμογής

Εφαρμογή Διαχειριστή

Αρχικά με την εκκίνηση της εφαρμογής, ο μόνος χρήστης που δημιουργείται αυτόματα είναι ο χρήστης διαχειριστή. Αφού ο χρήστης εισέλθει στην εφαρμογή έχει την επιλογή να πραγματοποιήσει σύνδεση, έτσι ώστε να γίνει αυθεντικοποίηση με βάση τα στοιχεία τα όποια θα εισάγει.

A screenshot of a computer

Description automatically generated

**Εικόνα 10 - Αρχική Οθόνη**

Κατα την επιλογή για σύνδεση στο σύστημα, ο χρήστης ανακατευθείνεται στην οθόνη του Keycloak για την αυθεντικοποίηση του. Σε αυτό το σημείο ο χρήστης πρέπει να εισάγει τα στοιχεία του (όνομα χρήστη και κωδικό πρόσβασης). Σε περίπτωση λάθους θα εμφανιστεί το αντίστοιχο μήνυμα στην οθόνη του, έτσι ώστε ο χρήστης να εισάγει ξανά τα στοιχεία του.

A screenshot of a login form

Description automatically generated

Εικόνα 11 – Αποτυχία σύνδεσης

Αφού πραγματοποιήσει σύνδεση θα μεταφερθεί στην επόμενη σελίδα η οποία θα του προσφέρει ένα μενού για την διαχείρηση των δεδομένων του συστήματος.

A screenshot of a computer

Description automatically generated

Εικόνα 12 - Μενού διαχειριστή

Στο μενού αυτό, ο διαχειριστής της εφαρμογής έχει την δυνατότητα να δημιουργήσει τις εγγραφές που χρειάζεται το σύστημα, έτσι ώστε να είναι λειτουργικό. Οι βασικές εγγραφές είναι οι Χρήστες (Users), οι οποίοι είναι οι χρήστες του συστήματος, αυτοί δηλαδή που θα έχουν την δυνατότητα για την δημιουργεία κάποιας παραγγελίας μέσα στο σύστημα.

Για να δημιουργήσει ο διαχειριστής έναν νέο χρήστη, πρέπει να κατευθυνθεί στην επιλογή «Users» του μενού και στην συνέχεια να επιλέξει το κουμπί δημιουργείας νέου χρήστη.

A screenshot of a computer

Description automatically generated

Εικόνα 13 - Δημιουργεία νέου χρήστη

Στην συνέχεια θα πραγματοποιηθεί ανακατεύθυνση στην οποία θα πρέπει να εισάγει τα απαραίτητα στοιχεία για την δημιουργεία της εγγραφής στο σύστημα. Τα στοιχεία αυτά είναι το όνομα χρήστη καθώς και ο κωδικός πρόσβασης και κάποια βασικά στοιχεία επικοινωνίας.

A screenshot of a car flow

Description automatically generated

Εικόνα 14 - Στοιχεία σύνδεσης

A screenshot of a car flow

Description automatically generated

Εικόνα 15 - Στοιχεία επικοινωνίας

Εν συνεχεία, αφού ο διαχειριστής έχει συμπληρώσει όλα τα απαραίτητα πεδία, το κουμπί εγγραφής (Register) θα ενεργοποιηθεί αυτόματα, ώστε να μπορέσει να ολοκληρωθεί η διαδικασία της εγγραφής. Αφού ο διαχειριστής πατήσει στο κουμπί, η εγγραφή του χρήστη θα δημιουργηθεί μέσα στην βάση καθώς και μέσα στο keycloak, ώστε να μπορεί ο χρήστης να πραγματοποιήσει την σύνδεση στο σύστημα μέσω της αυθεντικοποίησης. Ο κωδικός πρόσβασης δεν αποθηκεύεται ώς μέρος της βάσης, γιατί αυτό θα ήταν κάτι το οποίο θα εγκυμονούσε κινδύνους, αλλά μεταφέρεται στο keycloak κατά την εγγραφή του χρήστη.

A screenshot of a computer

Description automatically generated

Εικόνα 16 - Ολοκλήρωσης Εγγραφή Χρήστη

A screenshot of a computer

Description automatically generated

Εικόνα 17 - Δημιουργεία χρήστη στο Keycloak

Στην συνέχεια, ο διαχειριστής μπορεί να δημιουργήσει έναν πελάτη, ο οποίος πρέπει να συνδεθεί με κάποιον χρήστη, ο οποίος θα έχει δημιουργηθεί με βάση τα προηγούμενα βήματα. Όπως και κατά την δημιουργεία της εγγραφής του χρήστη στο σύστημα, έτσι και εδώ ο διαχειριστής έχει την δυνατότητα να εισάγει τα δεδομένα του πελάτη. Μερικά απο αυτά τα δεδομένα είναι:

* Κωδικός πελάτη: Ένας μοναδικός κωδικός για την αναγνώριση του πελάτη μέσα στο σύστημα.
* Όνομα
* Τύπος πελάτη: Εταιρεία ή πελάτης λιανικής
* Χρήστης: Ο χρήστης ο οποίος θα συνδεθεί με τον πελάτη.

Κάποια περαίτερω στοιχεία που θα χρειαστεί να συμπληρώσει ο διαχειριστής είναι τα στοιχεία επικοινωνίας του πελάτη καθώς και η διεύθυνση κατοικίας του.

A screenshot of a computer

Description automatically generated

Εικόνα 18 - Βασικά στοιχεία πελάτη

A screenshot of a video game

Description automatically generated

Εικόνα 19 - Στοιχεία επικοινωνίας πελάτη

A screenshot of a car flow

Description automatically generated

Εικόνα 20 - Στοιχεία διεύθυνσης πελάτη

Σε τελευταίο στάδιο, για να ολοκληρωθεί η διαδικασία της εγκατάστασης των βασικών δομών του συστήματος, ο διαχειριστής καλείται αν δημιουργήσει τις εγγραφές των μοντέλων καθώς και μαρκών των αυτοκινήτων που θα υποστηρίζει το σύστημα.

Για την δημιουργεία των μαρκών, θα πρέπει να κατευθυνθεί στο μενού «Brands» και όπως και προηγουμένος να πατήσει στο κουμπί για την δημιουργεία νέας μάρκας.

A screenshot of a car

Description automatically generated

Εικόνα 21 - Δημιουργεία μάρκας

Στην συνέχεια, στην σελίδα στην οποία θα κατευθυνθεί, θα πρέπει να κάνει εισαγωγεί των στοιχεία για την μάρκα την οποία θέλει να δημιουργήσει. Τα στοιχεία αυτά είναι ο κωδικός/μοναδικό αναγνωριστικό καθώς και το όνομα προβολής της μάρκας στο σύστημα.

A screenshot of a car flow

Description automatically generated

Εικόνα 22 - Δημιουργεία μάρκας

Ώς τελευταίο βήμα για την ολοκλήρωση της εγκατάστασης του συστήματος, πρέπει να δημιουργηθούν και τα μοντέλα των οχημάτων. Για την δημιουργεία των μοντέλων, όπως και στα προηγούμενα βήματα, θα πρέπει ο χρήστης μέσα απο το μενού επιλογών να κατευθυνθεί στο μενού «Models» και να επιλέξει την δημιουργεία νέας εγγαρφής.

A screenshot of a car

Description automatically generated

Εικόνα 23 - Δημιουργεία μοντέλου

Στην συνέχεια, με την ίδια ακριβώς λογική που ακολουθήθηκε και στις προηγούμενες σελίδες, ο χρήστης καλείται να εισάγει κάποια βασικά στοιχεία και χαρακτηριστικά του μοντέλου όπως ο κωδικός του μεντέλου, ο οποίος θα χρησιμοποιηθεί σαν μοναδικό αναγνριστικό της εγγραφής στο σύστημα, η περιγραφή, το έτος κατασκευής του μοντέλου και η μάρκα του μοντέλου.

A black screen with a blue background

Description automatically generated

Εικόνα 24 - Δημιουργεία Μοντέλου

Για το μοντέλο, χρειαζόμαστε επίσης και τον εξοπλισμό τον οποίο χαρακτηρίζει το μοντέλο. Ο εξοπλισμός απαρτίζεται απο τον βασικό εξοπλισμό και τον προαιρετικό. Στο σύστημά μας, το είδος του εξοπλισμού (βασικός ή προαιρετικός), καθορίζεται απο τις τιμές που έχουν τα πεδία «Internal Code» και «Code». Στις περιπτώσεις που τα πεδία αυτά έχουν την ίδια τιμή στο σύστημα ο εξοπλισμός θεωρείται ώς βασικός ενώ σε κάθε άλλη περίπτωση θεωρείται ώς προαιρετικός.

A screenshot of a computer

Description automatically generated

Εικόνα 25 - Δημιουργεία εξοπλισμού μοντέλου

Αφού έχει ολοκληρωθεί όλη η διαδικασία με την δημιουργεία εγγραφών, ώστε να είναι λειτουργικό το σύστημα, θα πρέπει στην συνέχεια να γίνει παραμετροποίηση του ERP συστήματος με το οποίο θα συγχρονίζονται τα δεδομένα.

Αρχικά θα πρέπει ο διαχειριστής του συστήματος να μεταβεί στο μενού «Settings» και απο εκεί να επιλέξει το «Advanced Settings» και στην συνέχεια να μεταβεί στο μενού «Service Connections».

A screenshot of a computer

Description automatically generated

Εικόνα 26 - Service Connections

Από το μενού που θα εμφανιστεί θα πρέπει να επιλέξει το «Integration Setup» το οποίο θα εμφανίσει την σελίδα για την εγκατάσταση των παραμέτρων για την διασύνδεση με το σύστημα μας.

A screenshot of a computer

Description automatically generated

Εικόνα 27 - Integration Setup

Στην οθόνη που θα εμφανιστεί θα πρέπει να εισάγουμε κάποιες τιμές στα κελιά ώστε να ενεργοποιήσουμε την επικοινωνία μέσω του Kafka με το σύστημα μας. Τα πεδία αυτά με τις αντίστοιχες τιμές τους είναι

* General Tab
  + Base Uri: Το Uri του Kafka Rest Proxy – προεπιλεγμένη τιμή: <http://host.docker.internal:8082>
* Data Synchronization Tab
  + Consumer Topic: Το θέμα από το οποίο θα λαμβάνουμε μηνύματα σχετικά με τα δεδομένα απο το σύστημά μας σε περίπτωση δημιουργείας ή ανανέωσης αυτών – προεπιλεγμένη τιμή: dev.general.datasync.erp.json
  + Setup Topic Producer: Το θέμα στο οποίο θα καταγράφουμε τις αλλαγές του ERP ώστε να μεταφέρονται στο σύστημά μας – προεπιλεγμένη τιμή: dev.general.datasync.erp.json
* Ordering Tab
  + Consumer Topic: Το θέμα από το οποίο θα λαμβάνουμε μηνύματα σχετικά με νέες παραγγελίες – προεπιλεγμένη τιμή: dev.general.orders.erp.json
  + Producer Topic: Στο θέμα αυτό θα καταγράφονται οι ανανεώσεις των παραγγελιών ανάλογα με την αλλαγή της κατάστασής τους μέσα στο ERP – προεπιλεγμένη τιμή: dev.general.orders.json

Αφού έχουν συμπληρώθει οι προαναφερθείσες τιμές, θα πρέπει να ενεργοποιηθεί το Enabled ώστε να ενεργοποιήσουμε την διασύνδεση μεταξύ των συστημάτων. Σε περίπτωση λάθους με την διασύνδεση θα εμφανιστεί και το αντίστοιχο μήνυμα λάθους.

A screenshot of a computer screen

Description automatically generated

Εικόνα 28 - Μήνυμα Λάθους

Κατά την επιτυχή διασύνδεση των συστημάτων το κουμπί Enabled θα ενεργοποιηθεί και το ClusterId θα πάρει αυτόματα τιμή με αυτή απο το Kafka Cluster.

A screenshot of a computer

Description automatically generated

Εικόνα 29 - Επιτυχής Διασύνδεση

Τελευταίο στάδιο, είναι η ενεργοποίηση των εργασιών που τρέχουν στο παρασκήνιο, οι οποίες ανταλλάσουν δεδομένα με το σύστημά μας, ώστε και τα δύο συστήματα να είναι συγχρονισμένα. Αφού τις ενεργοποιήσουμε, η κάθε διεργασία θα τρέχει στο παρασκήνιο ανά 5 λέπτα και είτε θα μεταβιβάζει είτε θα παραλαμβάνει δεδομένα.

A screenshot of a computer

Description automatically generated

Εικόνα 30 - Λίστα Διεργασίων Παρασκηνίου

Εφαρμογή Χρήστη

Εισερχόμενος στην εφαρμογή ο χρήστης, μέσω αυθεντικοποίησης με το Keycloak, μπορεί να δει το μενού των παραγγελιών, ανάλογα με την κατάσταση στην οποία βρίσκεται κάθε παραγγελία καθώς και την λίστα με τα οχήματα που έχουν δημιουργηθεί μέσα απο τις διάφορες παραγγελίες.

A screen shot of a car

Description automatically generated

Εικόνα 31 - Μενού χρήστη

Πηγαίνοντας στο μενού των ανοιχτών παραγγελιών, μπορούμε να δημιουργήσουμε μια νέα προσφορά. Η προσφορά αυτή θα προωθηθεί στο σύστημα του ERP προς αξιολόγηση και απο αυτήν θα δημιουργηθεί μια παραγγελία.

A screenshot of a computer

Description automatically generated

Εικόνα 32 - Δημιουργεία Προσφοράς

Κατά την δημιουργεία της προσφοράς, θα πρέπει ο χρήστης να επιλέξει το μοντέλο για το οποίο ενδιαφέρεται. Πολλά απο τα πεδία της φόρμας, έχουν προσυπληρωθεί αυτόματα με βάση τα στοιχεία του χρήστη μέσα απο το σύστημα.

A screenshot of a computer

Description automatically generated

Εικόνα 33 - Δημιουργεία Προσφοράς (2)

Κατά την επιλογή του μοντέλου, μεταφέρθηκαν στην προσφορά αυτόματα τα εξαρτήματα του μοντέλου και ο χρήστης έχει την επιλογή να επιλέξει ή όχι κάποια απο αυτά. Στην συνέχεια, καταχωρόντας την προσφορά, αύτη πλέον δεν μπορεί να τροποποιηθεί και η κατάστασή της έχει αλλάξει σε «Επιβεβαίωση Παραγγελίας».

A screenshot of a computer

Description automatically generated

Εικόνα 34 - Επιβεβαίωση Παραγγελίας

Η προσφορά έχει πλέον μεταφερθεί στο ERP σύστημα και αυτόματα έχει σταλεί και ενημερωτικό e-mail στον πελάτη ώστε να γνωρίζει το κόστος και την κατάσταση της προσφοράς.

A screenshot of a computer

Description automatically generated

Εικόνα 35 - Προσφορά στο ERP

A screenshot of a email

Description automatically generated

Εικόνα 36 - Ενημερωτικό e-mail προσφοράς

Εν συνεχεία, ο χρήστης του ERP καλείται να μετατρέψει την προσφορά αυτή σε παραγγελία. Για να το επιτύχει, θα πρέπει να εκτελέσει το «Make Order» απο την γραμμή ενεργειών.

A screenshot of a computer

Description automatically generated

Εικόνα 37 - Make Order

Μετά απο την εκτέλεση της ενέργειας αυτής, ο χρήστης μπορεί να βρεί την παραγγελία του πλέον κάτω απο τις εγκριθείσες παραγγελίες και αντιστοίχος θα έχει λάβει ένα μήνυμα ηλεκτρονικού ταχυδρομείου με την παραγγελία αυτή την φορά.

A screenshot of a car

Description automatically generated

Εικόνα 38 - Εγκριθείσα Παραγγελία

A screenshot of a mail

Description automatically generated

Εικόνα 39 - Μήνυμα Ηλεκτρονικού Ταχυδεομείου Παραγγελίας

Στην συνέχεια, ο χρήστης του ERP καλείται, ώστε να προχωρήσει στην τιμολόγηση, να συμπληρώσει τα απαραίτητα πεδία Αριθμός Πλαισίου (VIN) και Άδεια Οχήματος (License Number).

A screenshot of a computer

Description automatically generated

Εικόνα 40 - Παραγγελία ERP

Έπειτα, απο την γραμμή εργαλείων θα πρέπει να προχωρήσει στην εκτέλεση της ενέργειας της τιμολόγησης. Κατά την τιμολόγηση ο χρήστης θα μπορεί πλέον να βρεί την παραγγελία του κάτω απο το μενού των τιμολογημένων παραγγελιών και θα έχει την δυνατότητα να δει το τελικό παραστατικό του τιμολογίου.

A screenshot of a computer

Description automatically generated

Εικόνα 41 - Τιμολογημένη Παραγγελία (ERP)

A screenshot of a computer

Description automatically generated

Εικόνα 42 - Παραστατικό τιμολογημένης παραγγελίας

Μετά το πέρας της τιμολόγησης δημιουργείται στο σύστημα και το αντίστοιχο όχημα, όπως αυτό προκύπτει απο την διεκπεραίωση της όλης ροής. Το όχημα αυτό, είναι εμφανές μέσα στην λίστα των οχημάτων του χρήστη.

A screenshot of a computer

Description automatically generated

Εικόνα 43 - Όχημα

Σύνοψη

Αυτή η διπλωματική εργασία διερευνά τη χρήση του gRPC (Remote Procedure Call) ως εναλλακτική για την JSON με σκοπό τον προσανατολισμό της επικοινωνίας των μικρουπηρεσιών εντός του πλαισίου του Business Central ERP. Η μελέτη αποσκοπεί στο να αξιολογήσει τα πλεονεκτήματα και τις προκλήσεις που συνδέονται με τη χρήση του gRPC σε σύγκριση με τα παραδοσιακά πρωτόκολλα επικοινωνίας βασισμένα σε JSON.

Η έρευνα ξεκινά με μια σφαιρική ανάλυση της αρχιτεκτονικής των μικρουπηρεσιών και της αυξανόμενης χρήσης τους στον τομέα του Enterprise Resource Planning (ERP), επικεντρώνοντας ειδικά στο Business Central ERP της Microsoft. Οι μικρουπηρεσίες παρέχουν μοντουλαριότητα και επεκτασιμότητα, αλλά εξαρτώνται σημαντικά από την αποτελεσματική επικοινωνία μεταξύ τους.

Ο κύριος εστίαση της διπλωματικής εργασίας είναι η υλοποίηση του gRPC ως πρωτοκόλλου επικοινωνίας μεταξύ των μικρουπηρεσιών. Το gRPC προσφέρει οφέλη στην απόδοση σε σχέση με το παραδοσιακό JSON χρησιμοποιώντας το Protocol Buffers, ένα δυαδικό φορμά σειριοποίησης, προκαλώντας μειωμένη καθυστέρηση και αυξημένη αποτελεσματικότητα στη μεταφορά δεδομένων.

Το εμπειρικό κομμάτι της μελέτης περιλαμβάνει την ανάπτυξη και την εφαρμογή ενός πρωτότυπου συστήματος που ενσωματώνει το gRPC στην αρχιτεκτονική των μικρουπηρεσιών του Business Central ERP. Η διπλωματική αξιολογεί βασικούς δείκτες απόδοσης, όπως η καθυστέρηση, η επιτάχυνση και η χρήση πόρων, προκειμένου να καταστείλει την επίδραση της υιοθέτησης του gRPC στη συνολική αποτελεσματικότητα του συστήματος.

Επιπλέον, η έρευνα εξετάζει τις προκλήσεις και τους σκοπούς που συνδέονται με την εφαρμογή του gRPC, συμπεριλαμβανομένης της συμβατότητάς του με τα υπάρχοντα συστήματα, τις ασφαλειακές επιπτώσεις και την ευκολία ενσωμάτωσής του στο Business Central ERP.

Συνοψίζοντας, η διπλωματική εργασία παρουσιάζει μια σφαιρική ανάλυση της ενσωμάτωσης του gRPC σε μια αρχιτεκτονική μικρουπηρεσιών, ειδικά σχεδιασμένη για το πλαίσιο του Business Central ERP. Τα ευρήματα συνεισφέρουν σημαντικές πληροφορίες για τα πλεονεκτήματα και τις προκλήσεις της υιοθέτησης του gRPC ως πρωτοκόλλου επικοινωνίας στον τομέα των συστημάτων ERP, παρέχοντας έτσι βάση για περαιτέρω έρευνα και πρακτικές εφαρμογές στον τομέα.

Βιβλιογραφεία

* “gRPC vs Rest” (<https://grpc.io/>)
* “gRPC vs. REST: Key Similarities and Differences” (<https://blog.dreamfactory.com/grpc-vs-rest-how-does-grpc-compare-with-traditional-rest-apis/>)
* “Microservice Architecture” (<https://microservices.io/>)
* “Microservices” (<https://en.wikipedia.org/wiki/Microservices>)
* “ELK” (<https://www.elastic.co/guide/index.html>)

* 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