Εργασία στο Μάθημα της Τεχνολογίας Λογισμικού

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Εθνικό Μετσόβιο Πολυτεχνείο

Χειμερινό εξάμηνο 2021-2022

Διδάσκων: Β. Βεσκούκης

Θεματικό πεδίο

Το θεματικό πεδίο της εργασίας είναι το πρόβλημα της διαλειτουργικότητας στα διόδια αυτοκινητοδρόμων με διαφορετικά συστήματα αυτόματης διέλευσης. Ηδη, από το Νοέμβριο του 2020, τα ηλεκτρονικά συστήματα διέλευσης διοδίων διαλειτουργούν, δηλαδή το καθένα επιτρέπει τη διέλευση με τον πομποδέκτη οποιουδήποτε άλλου. Τα συστήματα αυτά είναι των οδών: aodos.gr, gefyra.gr, egnatia.eu, kentrikiodos.gr, moreas.com.gr, neaodos.gr, olympiaodos.gr. Το γεγονός αυτό αποτέλεσε την αφορμή για το θέμα της εργασίας, η οποία με κανένα τρόπο δεν συσχετίζεται με το πραγματικό σύστημα.

Η διαλειτουργικότητα αυτή δημιουργεί οφειλές μεταξύ των εταιριών διαχείρισης, οι οποίες πρέπει να εκκαθαρίζονται. Μια οφειλή μιας εταιρίας διαχείρισης Α προς μία εταιρία Β, δημιουργείται όταν ένα όχημα-κάτοχος πομποδέκτη του αυτοκινητοδρόμου Α, περάσει από διόδια του αυτοκινητοδρόμου Β. Τότε η εταιρία διαχείρισης του Α οφείλει στην εταιρία διαχείρισης του Β το αντίτιμο της διέλευσης. Επειδή στη γενική περίπτωση συμβαίνει και το αντίστροφο, δηλ οχήματα με πομποδέκτη του αυτοκινητόδρομου Β διέρχονται από διόδια του Α, τότε δημιουργούνται και οφειλές του Β προς τον Α. Τελικά, οι οφειλές των Α και Β προς αλλήλους συμψηφίζονται και καταβάλλεται μόνο η διαφορά από την εταιρία διαχείρισης του αυτοκινητόδρομου οι συνδρομητές του οποίου έκαναν διελεύσεις μεγαλύτερης αξίας στον άλλο αυτοκινητόδρομο.

Το λογισμικό που καλείστε να κατασκευάσετε είναι αυτό που θα εκτελείται σε έναν φορέα ανεξάρτητο από τους λειτουργούς των αυτοκινητόδρομων που αναφέρθηκαν, προκειμένου να υλοποιήσει τη διαχείριση της διαλειτουργικότητας, όπως περιγράφηκε. Το λογισμικό αυτό θα λαμβάνει με κατάλληλο τρόπο τα δεδομένα από τους λειτουργούς και θα υπολογίζει, σε συμφωνημένες χρονικές στιγμές, τις μεταξύ τους οφειλές μετά τους συμψηφισμούς. Δεχόμαστε ότι ισχύουν τα εξής:

- Οι λειτουργοί των αυτοκινητόδρομων δεν χρησιμοποιούν το ίδιο σύστημα διαχείρισης διελεύσεων. Καθένα χρησιμοποιεί ένα δικό του σύστημα και θα πρέπει να πραγματοποιήσει το ίδιο όποιες τροποποιήσεις - επεκτάσεις απαιτηθούν. Το σύστημα διαχείρισης διελεύσεων του κάθε αυτοκινητοδρόμου είναι εκτός του αντικειμένου ευθύνης της εφαρμογής που αναπτύσσουμε.
- Σε όλα τα σημεία διελεύσεων είναι διαθέσιμες οι λειτουργίες "provider" (δήλωση σε ποιον πάροχο ανήκει ένας πομποδέκτης), "balance" (υπόλοιπο χρημάτων στον πομποδέκτη) και "pass" (διέλευση και καταγραφή της οφειλής).
- Η φόρτιση ενός πομποδέκτη με χρήματα ("load") είναι δυνατή μόνο στο δίκτυο στο οποίο αυτός ανήκει.

Διατύπωση ζητουμένου

Το αντικείμενο της εργασίας του μαθήματος είναι η ανάπτυξη ενός πληροφοριακού συστήματος για την υλοποίηση της διαλειτουργικότητας στα διόδια, όπως περιγράφηκε, στο οποίο υποθέτουμε ότι θα έχουν πρόσβαση οι διάφοροι εμπλεκόμενοι (stakeholders). Η εργασία θα περιλαμβάνει την αναγνώριση και προδιαγραφή των απαιτήσεων, την αρχιτεκτονική και λεπτομερή σχεδίαση, την υλοποίηση επιλεγμένων λειτουργιών, τον έλεγχο και, ασφαλώς, την τεκμηρίωση όλων αυτών. Η προδιαγραφή των απαιτήσεων και η σχεδίαση, η διαχείριση του κώδικα, το testing, καθώς και η διοίκηση του έργου, θα γίνουν με χρήση κατάλληλων εργαλείων.

Η αναγνώριση των εμπλεκομένων (stakeholders) και των λειτουργιών που θα περιλαμβάνει το σύστημα θα γίνει μέσω συζητήσεων στο μάθημα. Με την ολοκλήρωση αυτών, κατάλογος με τους εμπλεκόμενους και τις λειτουργίες θα είναι διαθέσιμος στο χώρο του μαθήματος στο helios.ntua.gr. Από τον κατάλογο αυτό κάποιες λειτουργίες θα υλοποιηθούν από όλες τις ομάδες, ενώ κάποιες άλλες θα επιλέγονται από κάθε ομάδα, σύμφωνα με τον αριθμό των μελών της και οδηγίες που θα ανακοινωθούν.

Τα τμήματα του πληροφοριακού συστήματος που θα κατασκευάσετε έχουν ως εξής:

1. Ένα υποσύστημα back-end, το οποίο θα υποστηρίζει λειτουργίες διαχείρισης δεδομένων μεταξύ των διαφορετικών λειτουργών των αυτοκινητόδρομων. Οι λειτουργίες αυτές θα διατίθενται μέσω ενός REST API, οι προδιαγραφές του οποίου θα σας δοθούν.
2. Μία εφαρμογή CLI (Command Line Interface) για τις λειτουργίες προσπέλασης δεδομένων. Η εφαρμογή θα λειτουργεί ως client του REST API που παρέχεται από το back-end υποσύστημα, προσφέροντας στο χρήστη της τη δυνατότητα να εκτελεί λειτουργίες στα δεδομένα.
3. Μία δικτυακή εφαρμογή (Web Application), η οποία θα προσφέρει στο χρήστη δυνατότητες παρουσίασης των δεδομένων, όπως ενδεικτικά: πίνακες, διαγράμματα, οπτικοποίηση σε χάρτη. Η εφαρμογή αυτή θα λειτουργεί σε περιβάλλον Web browser και θα αποτελεί το front-end του συστήματος, το οποίο επίσης θα είναι client του REST API.

Ομάδες εργασίας

Η εργασία θα γίνει σε **ομάδες των 5 ατόμων**, οι οποίες θα υλοποιήσουν τον πλήρη κύκλο ανάπτυξης του συστήματος (ανάλυση απαιτήσεων, σύνταξη προδιαγραφών, σχεδιασμός και αρχιτεκτονική, υλοποίηση και έλεγχοι αποδοχής, εγκατάσταση και λειτουργία). Επιτρέπονται ομάδες με λιγότερα από 5 άτομα και με έως 6 άτομα, με κατάλληλη προσαρμογή των παραδοτέων, όπως εξηγείται στη συνέχεια.

Για τη συνεργατική διαχείριση των εκδόσεων της τεκμηρίωσης και του πηγαίου κώδικα, είναι υποχρεωτική η χρήση του Github. Επίσης, είναι υποχρεωτική η χρήση και άλλων εργαλείων όπως περιγράφεται παρακάτω. Η συγκρότηση των ομάδων θα πρέπει να έχει ολοκληρωθεί το αργότερο μέχρι τις **27.10.2021**.

Ελάχιστες κοινές τεχνικές προδιαγραφές

Το σύστημα που θα αναπτύξετε θα πρέπει να υποστηρίζει τα εξής χαρακτηριστικά (ελάχιστες κοινές προδιαγραφές):

1. Οι λειτουργίες που θα υλοποιηθούν θα επιλεγούν από κατάλογο που θα ανακοινωθεί στο helios.ntua.gr, σύμφωνα με το πλήθος των μελών κάθε ομάδας και λοιπές οδηγίες που θα δοθούν.
2. Το υποσύστημα back-end θα παρέχει κατάλληλο REST API, το οποίο προαιρετικά μπορεί να είναι συμβατό με το πρότυπο OpenAPI 3.0. Για τα endpoints που θα αναπτύξετε θα ανακοινωθούν προδιαγραφές κατά τη διάρκεια υλοποίησης της εργασίας και θα είναι κοινά για όλες τις ομάδες.
3. Η γλώσσα των διεπαφών χρήστη στην εφαρμογή CLI θα είναι η αγγλική. Η γλώσσα των διεπαφών χρήστη για τις άλλες εφαρμογές, όπου απαιτείται, θα είναι η ελληνική ή η αγγλική.
4. Κάθε ομάδα θα πρέπει να κάνει χρήση ενός εργαλείου αυτοματισμού του «χτισίματος» του λογισμικού (build automation).
5. Κάθε ομάδα θα συντάξει σενάρια ελέγχου και θα ενσωματώσει την εκτέλεση των αντίστοιχων δοκιμών με αυτόματο τρόπο για τις λειτουργίες του back-end υποσυστήματος.
6. Θα πρέπει να υποστηρίζεται το πρωτόκολλο HTTPS για όλες τις διεπαφές, μέσω self-signed certificate.

Οι απαιτήσεις αυτές εξειδικεύονται ανάλογα με το πλήθος των μελών της ομάδας, όπως φαίνεται στον πίνακα παραδοτέων στην επόμενη ενότητα.

Τεχνικές απαιτήσεις - εργαλεία

Θα χρησιμοποιηθούν τα ακόλουθα εργαλεία:

- **Github** ως τεριβάλλον διαχείρισης κώδικα. Παρακαλείστε να μην δημιουργήσετε κάποιο repository πριν ανακοινωθούν οι σχετικές οδηγίες.
- **Visual Paradigm 16.3 CE** ως εργαλείο παραγωγής διαγραμμάτων UML. Μπορείτε επίσης να χρησιμοποιήσετε την trial έκδοση Visual Paradigm enterprise (δωρεάν για ένα μήνα).
- **YouTrack** για τη διαχείριση έργων. Το YouTrack προσφέρεται ως δικτυακή εφαρμογή από το εργαστήριο Λογισμικού. Σχετικά θα υπάρξουν αναλυτικές οδηγίες.
- **UpSource** για reviews πηγαίου κώδικα. Το UpSource προσφέρεται ως δικτυακή εφαρμογή από το εργαστήριο Λογισμικού. Σχετικά θα υπάρξουν αναλυτικές οδηγίες.

Development stack που μπορούν να χρησιμοποιηθούν

- **Υλοποίηση κώδικα**: Python, javascript με nodejs/express. Επιτρέπονται και άλλες επιλογές (java, .net) με δήλωσή σας.
- **Διαχείριση δεδομένων**: ένα εκ των MySQL, MariaDB, PostgreSQL, Mongo, Elastic Search
- **Frontend**: η επιλογή frameworks και τεχνολογιών για frontend (π.χ. βιβλιοθήκες JS για διαγράμματα / γραφήματα στο web), είναι ελεύθερη.

Παραδοτέα

Λίστα παραδοτέων

Η λίστα των παραδοτέων του μαθήματος φαίνεται στον παρακάτω πίνακα.

ΠΑΡΑΔΟΤΕΟ	Ομάδες 1-2 ατόμων	Ομάδες 3-4 ατόμων	Ομάδες 5-6 ατόμων
Τεκμηρίωση			
Εγγραφο StRS - Stakeholder Requirements Specification	για 1 stakeholder	για 2 stakeholders	για 3 stakeholders
Εγγραφο SRS - Software Requirements Specification	1 use case	2 use cases	3 use cases
Διαγράμματα ER ή NoSQL JSON schemas	ΝΑΙ		
Διαγράμματα UML Activity / State	αντίστοιχα με τα use cases		
Διαγράμματα UML Sequence	αντίστοιχα με τα use cases		
Διαγράμματα UML Deployment	ΝΑΙ		
Διαγράμματα UML Component	ΝΑΙ		
Υλοποίηση			
Εισαγωγή, διαχείριση και πρόσβαση σε δεδομένα (backend)	ΝΑΙ		
Database dump (sql ή json)	ΝΑΙ		
RESTful API	ΝΑΙ		
API documentation - demo	ΝΑΙ		
Command line interface (CLI)		ΝΑΙ	ΝΑΙ
Επικοινωνία με το χρήστη (frontend)	για 1 use case	για 2 use cases	για 2 use cases
Εκτελέσιμη μορφή (build & deploy your code from source)	ΝΑΙ		
Testing			
Back-end functional tests	ΝΑΙ	ΝΑΙ	ΝΑΙ
CLI unit tests		ΝΑΙ	ΝΑΙ
CLI functional tests			ΝΑΙ
API functional tests	ΝΑΙ	ΝΑΙ	ΝΑΙ

Μορφότυποι Παραδοτέων

Η απαιτούμενη μορφή των παραδοτέων φαίνεται στον παρακάτω πίνακα.

ΠΑΡΑΔΟΤΕΟ	Μορφότυπος	Ονομα αρχείου παραδοτέου	Σχόλιο
Τεκμηρίωση			
Εγγραφο StRS - Stakeholder Requirements Specification	docx ή odt, σύμφωνα με template που διατίθεται	strs-softeng-XX.zip	Χρησιμοποιήστε τα πρότυπα έγγραφα 2021. Δεν γίνονται δεκτά pdf, παρά μόνο αν συνοδεύονται από αρχεία σε επεξεργάσιμη μορφή.
Εγγραφο SRS - Software Requirements Specification		srs-softeng-XX.zip	
Διαγράμματα ER ή NoSQL JSON schemas	Ενιαίο αρχείο Visual Paradigm (.vpp)	softeng21-XX.vpp	Προσοχή!: ένα αρχείο vpp για όλα τα διαγράμματα. Δεν γίνονται δεκτά διαγράμματα από σχεδιαστικά λογισμικά.
Διαγράμματα UML Activity / State			
Διαγράμματα UML Sequence			
Διαγράμματα UML Deployment			
Διαγράμματα UML Component			
Υλοποίηση			
Εισαγωγή, διαχείριση και πρόσβαση σε δεδομένα (backend)	Πηγαίος κώδικας στη γλώσσα υλοποίησης	(σύμφωνα με το περιβάλλον ανάπτυξης)	Θα πρέπει να περιλάβετε ένα αρχείο README, παρέχοντας μια συνοπτική περιγραφή των συστατικών της εφαρμογής και των βημάτων που απαιτούνται για το στήσιμο του περιβάλλοντος ανάπτυξης (IDE, βιβλιοθήκες, κλπ).
Database dump (sql ή json)	SQL ή json		
RESTful API	Πηγαίος κώδικας στη γλώσσα υλοποίησης		
API documentation - demo	Postman scripts		
Command line interface (CLI)	Πηγαίος κώδικας στη γλώσσα υλοποίησης		
Επικοινωνία με το χρήστη (frontend)			
Εκτελέσιμη μορφή (build & deploy your code from source)			
Testing			
Back-end functional tests	test scripts	(σύμφωνα με το περιβάλλον ανάπτυξης)	
CLI unit tests			
CLI functional tests			
API functional tests			

Προθεσμίες

Παράδοση 1: Περιλαμβάνει τα έγγραφα τεκμηρίωσης και τα διαγράμματα UML που τα συνοδεύουν, καθώς και τη σχεδίαση της ΒΔ. Η υποβολή θα πρέπει να έχει γίνει στο σύστημα helios το αργότερο **έως τα μεσάνυκτα της 15ης Νοεμβρίου 2021**.

Παράδοση 2: Περιλαμβάνει το σύνολο της τεκμηρίωσης (με ενημερωμένα τα της πρώτης παράδοσης, αν απαιτείται), καθώς και την υλοποίηση του backend και του REST API. Η υποβολή των εγγράφων θα γίνει στο σύστημα helios, ενώ ο πηγαίος κώδικας των backend και REST API θα καταχωρηθεί στο Github. Η προθεσμία είναι **έως τα μεσάνυκτα της 21ης Δεκεμβρίου 2021**.

Τελική παράδοση: Η τελική έκδοση του συνόλου της εργασίας περιλαμβάνει το σύνολο της τεκμηρίωσης (με όποιες ενημερώσεις απαιτούνται) και το σύνολο του πηγαίου κώδικα. Η τεκμηρίωση θα υποβληθεί στο helios και επίσης θα περιέχεται στον αντίστοιχο φάκελο στο github. Η παράδοση θα πρέπει να έχει γίνει έως **τα μεσάνυκτα της ημέρας πριν από την εξέταση του μαθήματος**.

Αναλυτικά οι παραδόσεις φαίνονται στον ακόλουθο πίνακα.

ΠΑΡΑΔΟΤΕΟ	**Παράδοση 1: 15.11.2021**	**Παράδοση 2: 21.12.2021**	**Τελική παράδοση: Ημ/νία εξέτασης**
Τεκμηρίωση			
Εγγραφο StRS - Stakeholder Requirements Specification	helios	helios	helios & [your github]/doc
Εγγραφο SRS - Software Requirements Specification	helios	helios	helios & [your github]/doc
Διαγράμματα ER ή NoSQL JSON schemas	helios	helios	helios & [your github]/doc
Διαγράμματα UML Activity / State	helios	helios	helios & [your github]/doc
Διαγράμματα UML Sequence		helios	helios & [your github]/doc
Διαγράμματα UML Deployment		helios	helios & [your github]/doc
Διαγράμματα UML Component		helios	helios & [your github]/doc
Υλοποίηση			
Εισαγωγή, διαχείριση και πρόσβαση σε δεδομένα (backend)		[your github repo]/backend	[your github]/backend
Database dump (sql ή json)		[your github repo]/database	[your github]/database
RESTful API		[your github repo]/api	[your github]/api
API documentation - demo		[your github repo]/api	[your github]/api
Command line interface (CLI)			[your github]/cli
Επικοινωνία με το χρήστη (frontend)			[your github]/frontend
Εκτελέσιμη μορφή (build & deploy your code from source)			[your github]/{backend, api, cli}

ΠΑΡΑΔΟΤΕΟ	Παράδοση 1: 15.11.2021	Παράδοση 2: 21.12.2021	Τελική παράδοση: Ημ/νία εξέτασης
Testing			
Back-end functional tests			[your github]/test-backend
CLI unit tests			[your github]/test-cli
CLI functional tests			[your github]/test-cli
API functional tests			[your github]/test-api

Βαθμολογία - βαρύτητες

Η βαθμολογία του μαθήματος προκύπτει κατά 50% από την εργασία και κατά 50% από την τελική εξέταση. Εκτός της υποβολής, η εργασία παρουσιάζεται και προφορικά από τα μέλη της ομάδας, σε ημερομηνίες μετά τη λήξη της εξεταστικής, οι οποίες θα ανακοινωθούν. Για τη λήψη προβιβάσιμου βαθμού στο μάθημα απαιτείται προβιβάσιμος βαθμός σε αμφότερες την εργασία και την τελική εξέταση.

Η βαθμολογία της εργασίας προκύπτει με τις ακόλουθες βαρύτητες:

- Τεκμηρίωση 35%
- Υλοποίηση 40%
- Testing 15%
- Χρήση εργαλείων 10%
- Γενική εικόνα 10%

Σημείωση 1: Κανονικοποίηση, εφόσον απαιτείται, γίνεται σε επίπεδο θέματος και όχι για το σύνολο της εργασίας ή της εξέτασης. Η βαθμολογία της εργασίας των μελών μιας ομάδας ενδέχεται να διαφοροποιηθεί ανάλογα με τα δεδομένα που θα καταγραφούν στο helios, στο Github και στο YouTrack/UpSource.

Σημείωση 2: Η τελική φόρμουλα βαθμολόγησης, όπως έχει ανακοινωθεί στο μάθημα, είναι η ακόλουθη:

```
FinalGrade = ceiling(sumproduct(PartialGrades;weights);1)
```

όπου ceiling(7,1;1) = ceiling(7,9;1) = 8

Συμπληρώσεις κλπ τροποποιήσεις της εκφώνησης θα ανακοινώνονται στο helios.

Εργασία στο Μάθημα της Τεχνολογίας Λογισμικού Προδιαγραφές του REST web API και του Command-Line Interface - CLI [v2]

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Εθνικό Μετσόβιο Πολυτεχνείο

Χειμερινό εξάμηνο 2021-2022

Διδάσκων: Β. Βεσκούκης

REST web API: Γενικές αρχές

Το σύστημα λογισμικού διαχείρισης διαλειτουργικότητας διοδίων στους αυτοκινητόδρομους που θα αναπτύξετε θα πρέπει να υποστηρίζει ένα RESTful Application Programming Interface (REST API) για την αποθήκευση και ανάκτηση των δεδομένων που αποθηκεύονται στη Βάση που θα υλοποιήσετε. Οι κλήσεις που καλείστε να υλοποιήσετε πρέπει να υλοποιηθούν όλες, ανεξάρτητα από το αν θα χρησιμοποιηθούν από τις περιπτώσεις χρήσης που θα επιλέξετε.

API documentation

Το REST API που θα υλοποιήσετε θα πρέπει να περιλαμβάνει documentation με τη μορφή OpenAPI 3.0 (προτιμότερο) ή σε μορφή json αρχείου που παράγεται από το λογισμικό Postman.

Base URL

Το REST API θα είναι διαθέσιμο στο ακόλουθο base URL για όλες τις εργασίες:

https://{{host}}:9103/interoperability/api

όπου {{host}} είναι ο localhost ή όπου γίνει το deployment.

Τα επιμέρους Resources (REST endpoints) που θα διατίθενται μέσω του API θα είναι προσβάσιμα μέσω του παραπάνω base URL, ως εξής:

{baseURL}/{service}/{path-to-resource}

Όπου {service} μία από τις υπηρεσίες που θα διατίθεται, όπως αναφέρονται παρακάτω. Για παράδειγμα, το endpoint για την ανάκτηση του αριθμού διελεύσεων του οχήματος με ID "RV87TIY76692" για το μήνα Νοέμβριο 2021 είναι το

```
https://localhost:9103/interoperability/api/events/RV87TIY76692/20211101/20211130
```

Όλα τα αποτελέσματα που επιστρέφει το API θα είναι ταξινομημένα ως προς το πεδίο χρόνου που περιέχουν με αύξουσα τάξη.

Μορφότυποι δεδομένων

Το REST API θα υποστηρίζει τον μορφότυπο JSON (content-type: application/json) και τον μορφότυπο CSV (content-type: text/csv). Η επιλογή του μορφότυπου θα καθορίζεται στην αίτηση ως εξής (query parameter):

```
{baseURL}/{service}/{path-to-resource}?format={json|csv}
```

Αν η παράμετρος format δεν παρέχεται σε κάποια αίτηση, να θεωρήσετε ότι το json θα είναι η default τιμή. Η κωδικοποίηση χαρακτήρων (character encoding) θα πρέπει να είναι UTF8. Για παράδειγμα, η προηγούμενη κλήση με αίτημα μορφότυπου δεδομένων "csv", έχει ως εξής:

```
https://localhost:9103/interoperability/api/events/RV87TIY76692/20211101/20211130&format=csv
```

Διαπίστευση χρηστών (προαιρετική)

Για λόγους ελέγχου πρόσβασης των διαφορετικών stakeholders στο σύστημα, για τη χρήση του API μπορείτε προαιρετικά να υλοποιήσετε μηχανισμό διαπίστευσης των χρηστών της επιλογής σας. Στην περίπτωση αυτή, λογαριασμοί χρηστών θα δημιουργούνται από το διαχειριστή του συστήματος μέσω του Command Line Interface (CLI), όπως θα περιγραφεί στη συνέχεια.

Κατά την κλήση του API, τα διαπιστευτήρια του χρήστη (πχ user access token), κωδικοποιημένα με τον τρόπο που εσείς κρίνετε πιο συμβατό με τη σχετική βέλτιστη πρακτική, θα πρέπει να παρέχονται σε ειδικό για το σκοπό αυτό custom HTTP header X-OBSERVATORY-AUTH.

Διαχείριση σφαλμάτων

Κάθε κλήση στο API θα πρέπει να επιστρέφει τα κατάλληλα HTTP status codes σε περίπτωση σφάλματος. Ειδικότερα, θα επιστρέφονται οι ακόλουθοι κωδικοί:

200	Success	Σε περίπτωση επιτυχούς κλήσης
400	Bad request	Σε περίπτωση που οι παράμετροι που δίνονται σε μία κλήση δεν είναι έγκυρες (π.χ. κενό υποχρεωτικό πεδίο)
401	Not authorized	Σε περίπτωση που η αίτηση γίνεται από μη διαπιστευμένο χρήστη
402	No data	Σε περίπτωση που η απάντηση στην κλήση είναι κενή
500	Internal server error	Σε περίπτωση οποιουδήποτε άλλου σφάλματος

Πρόσβαση και διαχείριση

Login & Logout (προαιρετικά)

Το back-end σας θα υποστηρίζει δύο endpoints για το Login και το Logout των χρηστών. Ειδικότερα:

1. **{baseURL}/login**: Υποστηρίζει την μέθοδο POST και λαμβάνει τις παραμέτρους username, password του χρήστη κωδικοποιημένους ως "application/x-www-form-urlencoded". Σε περίπτωση επιτυχούς διαπίστευσης του χρήστη, επιστρέφει ένα json object με το token αυτού: {πχ "token":"FOO"}.
2. **{baseURL}/logout**: Υποστηρίζει τη μέθοδο POST και δε λαμβάνει παραμέτρους (ΠΡΟΣΟΧΗ: το token του χρήστη που πρέπει να «αποσυνδεθεί» περιέχεται στον ειδικό γι' αυτό το σκοπό custom HTTP header, όπως αναφέρθηκε παραπάνω). Σε περίπτωση επιτυχίας, επιστρέφει μόνο το status code 200 (empty response body).

Διαχειριστικά Endpoints

Το back-end σας θα υποστηρίζει τα παρακάτω endpoints, τα οποία θα λειτουργήσουν επικουρικά για τον πλήρως αυτοματοποιημένο έλεγχο που θα γίνει κατά την εξέταση της εργασίας:

1. **{baseURL}/admin/healthcheck**: Υποστηρίζει τη μέθοδο GET και επιβεβαιώνει τη συνδεσιμότητα (end-to-end connectivity) μεταξύ του χρήστη και της βάσης δεδομένων. Το back-end, δηλαδή, θα πρέπει να ελέγξει τη συνδεσιμότητα με τη ΒΔ για να απαντήσει στο αίτημα. Σε περίπτωση επιτυχούς σύνδεσης επιστρέφεται το json object: {"status":"OK", "dbconnection":[connection string]}, διαφορετικά επιστρέφεται {"status":"failed", "dbconnection":[connection string]}. Στο connection string περιέχεται ό,τι απαιτείται για τη ΒΔ της επιλογής σας.
2. **{baseURL}/admin/resetpasses:** Υποστηρίζει τη μέθοδο POST και προβαίνει σε αρχικοποίηση του πίνακα γεγονότων διέλευσης (διαγραφή όλων των γεγονότων). Επίσης, αρχικοποίηση του default διαχειριστικού λογαριασμού (username: admin, password: freepasses4all), εφόσον υλοποιήσετε λογαριασμούς χρηστών. Σε περίπτωση επιτυχίας, επιστρέφεται το json object: {"status":"OK"}, διαφορετικά επιστρέφεται {"status":"failed"} με (προαιρετική) πληροφορία σχετικά με την αποτυχία.
3. **{baseURL}/admin/resetstations:** Υποστηρίζει τη μέθοδο POST και προβαίνει σε αρχικοποίηση του πίνακα σταθμών διοδίων με τις τιμές που σας δόθηκαν ως ενδεικτικά δεδομένα. Προαιρετικά, με δικές σας τιμές που περιέχουν τα ονόματα των σταθμών και τα λοιπά δεδομένα που θα επιλέξετε να συμπεριλάβετε στη ΒΔ, όπως πχ η θέση, κ.ά.. Σε περίπτωση επιτυχίας, επιστρέφεται το json object: {"status":"OK"}, διαφορετικά επιστρέφεται {"status":"failed"} με (προαιρετική) πληροφορία σχετικά με την αποτυχία.
4. **{baseURL}/admin/resetvehicles:** Υποστηρίζει τη μέθοδο POST και προβαίνει σε αρχικοποίηση του πίνακα οχημάτων με τις τιμές που σας δόθηκαν ως ενδεικτικά δεδομένα. Σε περίπτωση επιτυχίας, επιστρέφεται το json object: {"status":"OK"}, διαφορετικά επιστρέφεται {"status":"failed"} με (προαιρετική) πληροφορία σχετικά με την αποτυχία.

Λειτουργία συστήματος

Ακολούθως δίνονται τέσσερα χαρακτηριστικά endpoints που μπορούν να χρησιμοποιηθούν για την αλληλεπίδραση διαφόρων stakeholders προς το σύστημα και την επιτέλεση των βασικών λειτουργιών του. Είστε ελεύθεροι να υλοποιήσετε επιπλέον endpoints, μόνο εφόσον αυτό απαιτείται για τις περιπτώσεις χρήσης που θα επιλέξετε. Οι ημερομηνίες που δίνονται ως REST παράμετροι πρέπει να έχουν τη μορφή YYYYMMDD.

a. {baseURL}/PassesPerStation/:stationID/:date_from/:date_to

Επιστρέφεται λίστα με την ανάλυση των διελεύσεων (Passes) για τον σταθμό διοδίων και για την περίοδο που δίνονται στο URL. Η παράσταση ημερομηνιών που επιστρέφονται πρέπει να είναι της μορφής "YYYY-MM-DD HH:MM:SS"

Πεδίο	**Τύπος**	**Περιγραφή**
Station	String	Το μοναδικό ID του σημείου διέλευσης
StationOperator	String	Ο διαχειριστής των διοδίων του σημείου διέλευσης
RequestTimestamp	String	Η ημερομηνία/ώρα της κλήσης του endpoint

Πεδίο	Τύπος	Περιγραφή
PeriodFrom	String	Η αιτούμενη περίοδος (από)
PeriodTo	String	Η αιτούμενη περίοδος (έως)
NumberOfPasses	Integer	Ο αριθμός γεγονότων διέλευσης στην περίοδο
PassesList:	List	(Περιλαμβάνει [NumberOfPasses] στοιχεία)
PassIndex	Integer	Α/Α διέλευσης στην περίοδο (1, 2, 3, ...)
PassID	String	Το ID του γεγονότος διέλευσης
PassTimeStamp	String	Η χρονική στιγμή της διέλευσης
VevicleID	String	Η ταυτότητα του οχήματος
TagProvider	String	Ο πάροχος του tag διελεύσεων για το όχημα
PassType	String	"home" ή "visitor"
PassCharge	Float	Το κόστος διέλευσης

b. {baseURL}/PassesAnalysis/:op1_ID/:op2_ID/:date_from/:date_to

Επιστρέφεται λίστα με την ανάλυση ανά σημείο των γεγονότων διέλευσης που πραγματοποιήθηκαν με tag του **op2_ID** σε σταθμούς του **op1_ID**. Ολα τα γεγονότα που επιστρέφονται είναι τύπου "visitor" στην κλήση PassesPerStation. Η παράσταση ημερομηνιών που επιστρέφονται πρέπει να είναι της μορφής "YYYY-MM-DD HH:MM:SS"

Πεδίο	Τύπος	Περιγραφή
op1_ID	String	Το ID του operator1
op2_ID	String	Το ID του operator2
RequestTimestamp	String	Η ημερομηνία/ώρα της κλήσης του endpoint
PeriodFrom	String	Η αιτούμενη περίοδος (από)
PeriodTo	String	Η αιτούμενη περίοδος (έως)
NumberOfPasses	Integer	Ο αριθμός γεγονότων διέλευσης στην περίοδο
PassesList:	List	(Περιλαμβάνει [NumberOfPasses] στοιχεία)
PassIndex	Integer	Α/Α διέλευσης στην περίοδο (1, 2, 3, ...)
PassID	String	Το ID του γεγονότος διέλευσης
StationID	String	Το ID του σταθμού διέλευσης
TimeStamp	String	Η χρονική στιγμή της διέλευσης
VehicleID	String	Η ταυτότητα του οχήματος
Charge	Float	Το κόστος διέλευσης

c. {baseURL}/PassesCost/:op1_ID/:op2_ID/:date_from/:date_to

Επιστρέφεται ο αριθμός των γεγονότων διέλευσης που πραγματοποιήθηκαν με tag του **op2_ID** σε σταθμούς του **op1_ID**, καθώς και το κόστος τους, δηλαδή το ποσό που ο op2_ID οφείλει στον op1_ID, για τη δοσμένη περίοδο.

Πεδίο	Τύπος	Περιγραφή
op1_ID	String	Το ID του operator1
op2_ID	String	Το ID του operator2
RequestTimestamp	String	Η ημερομηνία/ώρα της κλήσης του endpoint
PeriodFrom	String	Η αιτούμενη περίοδος (από)
PeriodTo	String	Η αιτούμενη περίοδος (έως)
NumberOfPasses	Integer	Ο αριθμός γεγονότων διέλευσης στην περίοδο
PassesCost	Float	Το συνολικό κόστος των διελεύσεων

2d. {baseURL}/ChargesBy/:op_ID/:date_from/:date_to

Επιστρέφεται ο αριθμός των γεγονότων διέλευσης που πραγματοποιήθηκαν σε σταθμούς του **op_ID**, από οχήματα όλων των άλλων operators, καθώς και το κόστος τους, δηλαδή το ποσό που καθένας από τους λοιπούς operators οφείλει στον Operator **op_ID**, για τη δοσμένη περίοδο.

Πεδίο	Τύπος	Περιγραφή
op_ID	String	Το ID του operator
RequestTimestamp	String	Η ημερομηνία/ώρα της κλήσης του endpoint
PeriodFrom	String	Η αιτούμενη περίοδος (από)
PeriodTo	String	Η αιτούμενη περίοδος (έως)
PPOList:	List	Περιλαμβάνει τόσα στοιχεία όσοι και οι (distict) operators, των οποίων οχήματα πέρασαν από διόδια του op_ID τη δοσμένη περίοδο
VisitingOperator	String	Το ID του visiting operator
NumberOfPasses	Integer	Ο αριθμός γεγονότων διέλευσης στην περίοδο
PassesCost	Float	Το συνολικό κόστος των διελεύσεων το οποίο οφείλεται από τον VisitingOperator στον op_ID

Command Line Interface - CLI

Γενικές αρχές

Οι προδιαγραφές του CLI περιλαμβάνουν κλήσεις οι οποίες είναι ισοδύναμες με εκείνες του REST API. Το CLI πρέπει να είναι διαθέσιμο μόνο από την κονσόλα (command line, ssh) του συστήματος που φιλοξενεί την εφαρμογή που θα κατασκευάσετε. Λογαριασμοί χρηστών, εφόσον υλοποιηθούν, δημιουργούνται μόνο μέσω του CLI.

Ανάκτηση δεδομένων

Το CLI θα υποστηρίζει τον μορφότυπο JSON και τον μορφότυπο CSV, όπως ακριβώς και στο REST API. Η επιλογή του μορφότυπου θα καθορίζεται σε αντίστοιχη υποχρεωτική παράμετρο όπως αναφέρεται πιο κάτω. Όλα τα αποτελέσματα που επιστρέφει το CLI θα είναι ταξινομημένα ως προς το χρόνο στον οποίο αναφέρονται με αύξουσα τάξη. Το CLI θα καλείται από τη γραμμή εντολών με κλήσεις της μορφής:

```
$ se21XX scope --param1 value1 [--param2 value2 ...] --format fff
```

Όπου XX το αναγνωριστικό της ομάδας σας και SCOPE όπως αναφέρεται στον πίνακα που ακολουθεί. Σε περίπτωση που δεν δίνονται παράμετροι, θα πρέπει να εμφανίζονται οι υποστηριζόμενες από το CLI παράμετροι για όλα τα scopes. Για παράδειγμα, η κλήση για την ανάκτηση των γεγονότων διέλευσης από τον σταθμό AO01 για το μήνα Νοέμβριο είναι η εξής:

```
$ se21XX passesperstation --station AO01 --from 20211101 --to 20211130 --format json
```

Θα πρέπει να υποστηρίζονται οι ακόλουθες παράμετροι ανά scope.

Scope	Επίπεδο χρήστη	Υποχρεωτικές λοιπές παράμετροι	Αντίστοιχη κλήση REST API
healthcheck	Κανένα	Καμία	/admin/healthcheck
resetpasses	Κανένα	Καμία	/admin/resetpasses
resetstations	Κανένα	Καμία	/admin/resetstations
resetvehicles	Κανένα	Καμία	/admin/resetvehicles
login (*)	Κανένα	--username --passw	/login
logout (*)	Διαπιστευμένος Χρήστης (*)	Καμία	/logout
passesperstation	Διαπιστευμένος Χρήστης (*)	--station --datefrom --dateto	/PassesPerStation
passesanalysis	Διαπιστευμένος Χρήστης (*)	--op1 --op2 --datefrom --dateto	/PassesAnalysis
passescost	Διαπιστευμένος Χρήστης (*)	--op1 --op2 --datefrom --dateto	/PassesCost
chargesby	Διαπιστευμένος Χρήστης (*)	--op1 --datefrom --dateto	/ChargesBy
admin	Διαχειριστής	Βλ. παρακάτω διαχείριση συστήματος	

Η παράμετρος --format είναι υποχρεωτική για όλες τις κλήσεις και λαμβάνει τις τιμές json ή csv.

Διαχείριση συστήματος

Για τις διαχειριστικές λειτουργίες (scope: admin) θα πρέπει να υποστηρίζονται οι ακόλουθες παράμετροι.

Παράμετρος	**Επιτρεπτές τιμές**	**Υποχρεωτικές λοιπές παράμετροι**	**Λειτουργία**	**Επιστρέφει**
--usermod (*)		--username --passw	Δημιουργία νέου χρήστη ή αλλαγή password	
--username (*)	αλφαριθμητικό με λατινικούς χαρακτήρες	--passw		
--passw (*)	password (οτιδήποτε εκτός κενών)			
--users (*)	username (αλφαριθμητικό με λατινικούς χαρακτήρες)		Εμφάνιση κατάστασης χρήστη	Αποτέλεσμα εκτέλεσης: username
--passesupd		--source	Πρόσθεση νέων Passes από αρχείο CSV	Αποτέλεσμα εκτέλεσης όπως στην τεκμηρίωση του API
--source	Filename			

(*) μόνο εφόσον υλοποιήσετε σύστημα διαχείρισης λογαριασμών χρηστών

Παραδείγματα περιπτώσεων κλήσεων

ΔΙΑΧΕΙΡΙΣΗ	**API**	**CLI**

Αρχικοποίηση και έλεγχος συστήματος (Εκτέλεση μόνο από συγκεκριμένο IP ή με διαπίστευση χρήστη και από την κονσόλα για το CLI)	`/admin/resetpasses`
	`/admin/resetvehicles`
	`/admin/resetstations`
	`/admin/healthcheck`
		`se21xx resetpasses`
		`se21xx resetvehicles`
		`se21xx resetstations`
		`se21xx healthcheck`

Είσοδος - έξοδος από το σύστημα (προαιρετικά)	`/login` `/logout` σε συνδυασμό με κατάλληλα header	`se21xx login --username [user] --passw [password]` `se21xx logout`

Διαχείριση χρηστών (προαιρετικά)	-	`se21xx admin --usermod --username [user]`
		`--passw [newpassw]`
		`se21xx admin --users`

Εισαγωγή διελεύσεων από αρχείο	-	`se21 admin --passesupd --source` `"/data/newpassesXXXX.csv"`
ΛΕΙΤΟΥΡΓΙΑ	**API**	**CLI**

Διελεύσεις ανά σταθμό	`/PassesPerStation/:stationID/:datefrom/:dateto`	`se21xx passesperstation --station [stationID]`
		`--datefrom [datefrom] --dateto [dateto]`
		`--format csv`

Ανάλυση διελεύσεων μεταξύ 2 λειτουργών	`/PassesAnalysis/:op1/:op2/:datefrom/:dateto`	`se21xx passesanalysis --op1 [op1] --op2 [op2]`
		`--datefrom [datefrom] --dateto [dateto]`
		`--format csv`

Κόστος διελεύσεων μεταξύ 2 λειτουργών	`/PassesCost/:op1/:op2/:datefrom/:dateto`	`se21xx passescost --op1 [op1] --op2 [op2] --datefrom`
		`[datefrom] --dateto [dateto]`
		`--format json`

Διελεύσεις από οχήματα άλλων λειτουργών	`/ChargesBy/:op/:datefrom/:dateto`	`se21xx chargesby --op1 [op] --datefrom [datefrom]` `--dateto [dateto] --format json`