Έγγραφο απαιτήσεων λογισμικού (SRS)

ΠΡΟΣΑΡΜΟΓΗ ΤΟΥ ΑΝΤΙΣΤΟΙΧΟΥ ΕΓΓΡΑΦΟΥ ΤΟΥ ΠΡΟΤΥΠΟΥ ISO/IEC/IEEE 29148:2011

intelliQ

1. Εισαγωγή

1.1 Εισαγωγή: σκοπός του λογισμικού

Συγγραφείς	Δημήτριος Γεωργούσης, Γεώργιος-Αλέξιος Καπετανάκης
Περιγραφή	Το λογισμικό αυτό αποτελεί περιβάλλον αλληλεπίδρασης με «έξυπνα ερωτηματολόγια» για την πραγματοποίηση διαδικτυακών ερευνών κάθε είδους. Ως «έξυπνο» χαρακτηρίζεται ένα ερωτηματολόγιο όταν η πορεία των απαντήσεων καθορίζεται δυναμικά από αυτές που έχουν προηγηθεί, δηλαδή, παρέχεται μεγαλύτερη εξατομίκευση της εμπειρίας χρήστη. Απευθύνεται σε **πρόσωπα που πραγματοποιούν έρευνες** και το **κοινό αυτών**. Για τους πρώτους παρέχεται εφαρμογή που ελέγχει την εγκυρότητα τέτοιων ερωτηματολογίων, τα αποθηκεύει σε κατάλληλη μορφή και ανταποκρίνεται σε αιτήματα ερωταπαντήσεων του κοινού. Παρέχονται, λοιπόν, **διαχειριστικές** λειτουργίες όχι μόνο για τα ίδια τα ερωτηματολόγια αλλά και για τη συλλογή απαντήσεων σε αυτά. Οι **ερωτηθέντες** αλληλοεπιδρούν με το λογισμικό συνδεόμενοι διαδικτυακά στον σύνδεσμο της εφαρμογής – frontend. Αυτή παρέχει απλό και εύχρηστο περιβάλλον για την εγκατάσταση συνεδρίας για την υποβολή απαντήσεων και, στο τέλος, επιστρέφει στον χρήστη κατάλογο με τις επιλογές του κατά τη διάρκειά της.

1.2 Διεπαφές (interfaces)

1.2.1 Διεπαφές με εξωτερικά συστήματα

Web Browser

Ο Web Browser με τον οποίο συνδέεται ο χρήστης στην εφαρμογή μας.

Εσωτερικές Συνιστώσες - Components

SessionStorage

Αποθηκεύει τοπικά τα δεδομένα του χρήστη κατά τη διάρκεια χρήσης του frontend. Τα δεδομένα διατηρούνται σε refresh αλλά όχι σε κλείσιμο και ξανά-άνοιγμα του browser.

Περιεχόμενα Στοιχεία - Resident Elements

SessionStorage

Αποθηκεύει τοπικά τα δεδομένα του χρήστη κατά τη διάρκεια χρήσης του frontend. Τα δεδομένα διατηρούνται σε refresh αλλά όχι σε κλείσιμο και ξανά-άνοιγμα του browser.

Σχέσεις

Σχέση	Από	Σε
unnamed	Web Browser	Σχόλιο για τον Web browser
unnamed	Web Browser	HTTPS / TCP Port: 443

Svelte - Vite / front-end

Οι χρήστες συνδέονται στο link της εφαρμογής και αλληλεπιδρούν με αυτή.

Εσωτερικές Συνιστώσες - Components

Communicate with Backend

Επικοινωνία με το backend για αποστολή και λήψη δεδομένων.

Present And Accept User Data

Αποτελεί το μέρος της εφαρμογής με το οποίο αλληλεπιδρά άμεσα ο browser

Store User Information

Επικοινωνία με την τοπική αποθήκη δεδομένων συνόδου του Browser του χρήστη, ώστε να λειτουργεί κατάλληλα η εφαρμογή.

Περιεχόμενα Στοιχεία - Resident Elements

Communicate with Backend

Επικοινωνία με το backend για αποστολή και λήψη δεδομένων.

Present And Accept User Data

Αποτελεί το μέρος της εφαρμογής με το οποίο αλληλεπιδρά άμεσα ο browser

Store User Information

Επικοινωνία με την τοπική αποθήκη δεδομένων συνόδου του Browser του χρήστη, ώστε να λειτουργεί κατάλληλα η εφαρμογή.

Σχέσεις

Σχέση	Από	Σε
unnamed	Svelte - Vite / front-end	Σχόλιο frontend εφαρμογής
unnamed	Svelte - Vite / front-end	REST API / HTTP / TCP Port: 9103
unnamed	HTTPS / TCP Port: 443	Svelte - Vite / front-end

Node.js / back-end

Η εφαρμογή που θα περιέχει τα endpoints και τη λογική της ιστοσελίδας.

Εσωτερικές Συνιστώσες - Components

Admin Controller

Διαχειρίζεται τις διαδρομές του Admin.

app

Κεντρικός διαχειριστής όλων των διαδρομών. Χρησιμοποιεί τους Routers για κατανομή κατάλληλης εργασίας στον καθένα

Datatabase Communication

Middleware που στέλνει και λαμβάνει δεδομένα από τη βάση δεδομένων.

Format

Middleware που μορφοποιεί τα δεδομένα ανάλογα με τις επιθυμίες του χρήστη.

Functionality Controller

Validator

Επικύρωση λαμβανόμενων αιτημάτων/δεδομένων.

Περιεχόμενα Στοιχεία - Resident Elements

Admin Controller

Διαχειρίζεται τις διαδρομές του Admin.

app

Κεντρικός διαχειριστής όλων των διαδρομών. Χρησιμοποιεί τους Routers για κατανομή κατάλληλης εργασίας στον καθένα

Datatabase Communication

Middleware που στέλνει και λαμβάνει δεδομένα από τη βάση δεδομένων.

Format

Middleware που μορφοποιεί τα δεδομένα ανάλογα με τις επιθυμίες του χρήστη.

Functionality Controller

Validator

Επικύρωση λαμβανόμενων αιτημάτων/δεδομένων.

Σχέσεις

Σχέση	Από	Σε
unnamed	Node.js / back-end	MongoDB Wire Protocol / TCP Port: 27017
unnamed	REST API / HTTP / TCP Port: 9103	Node.js / back-end
unnamed	Σχόλιο backend component	Node.js / back-end

MongoDB Database

Η βάση που θα αποθηκεύει τα ερωτηματολόγια και τις απαντήσεις σε αυτά.

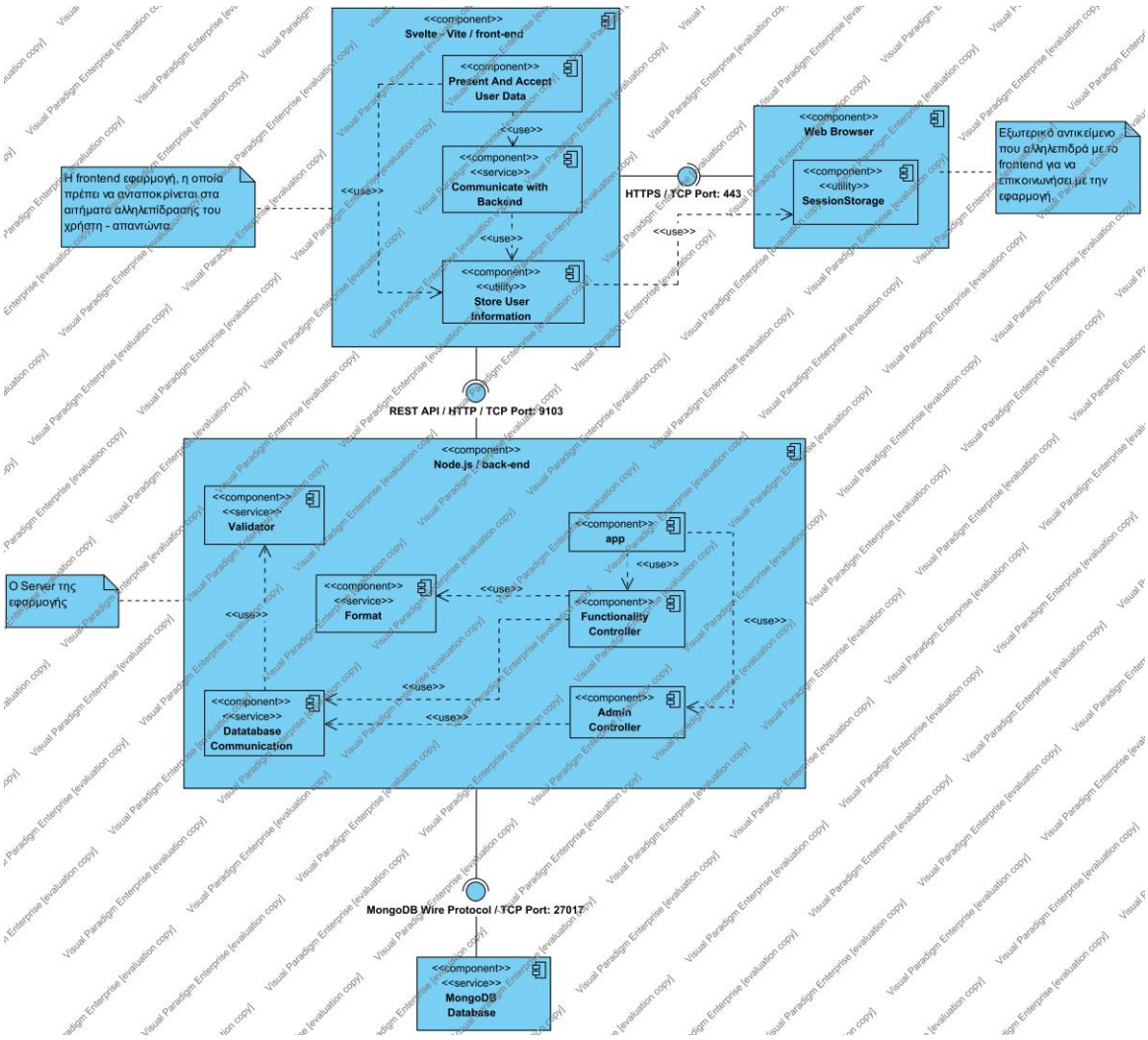
Stereotypes

<<service>>

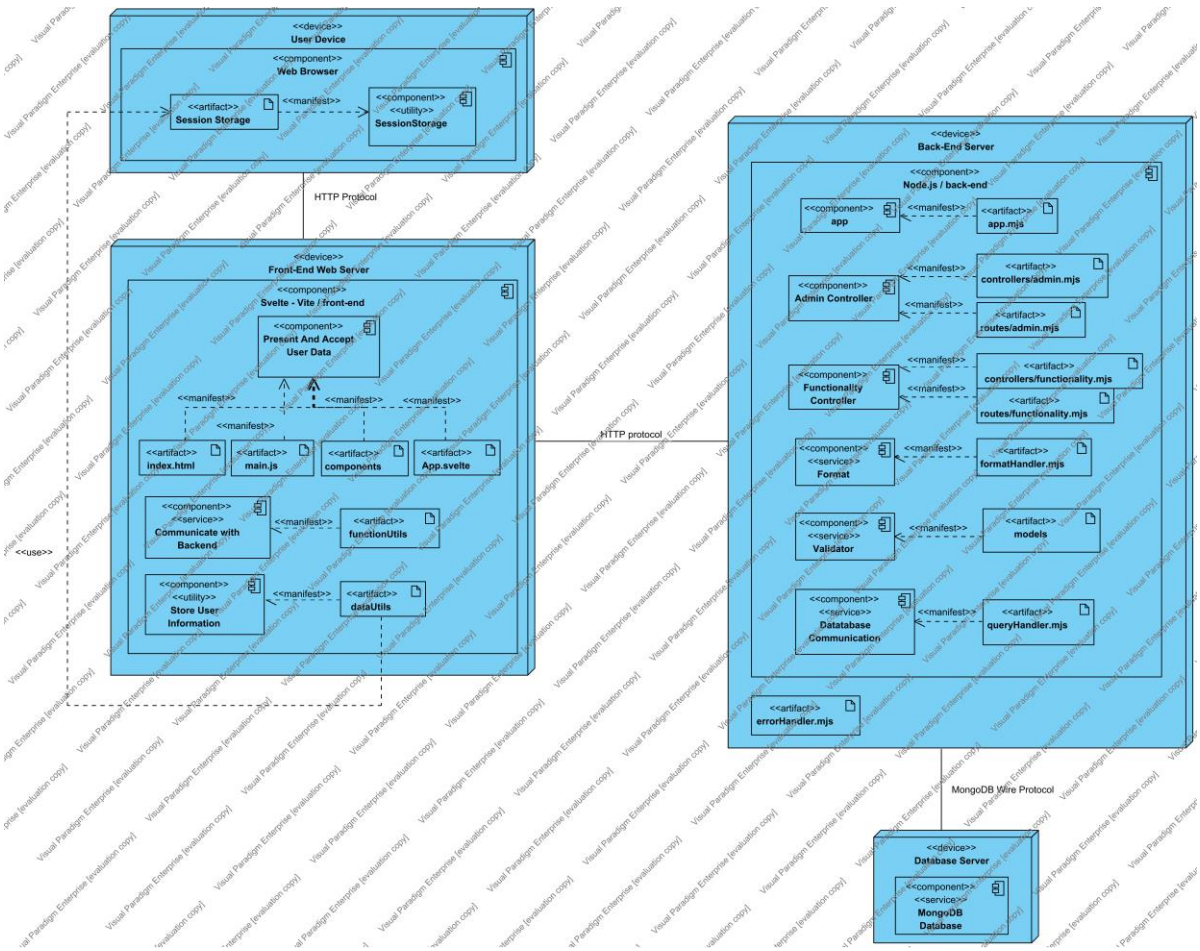
Σχέσεις

Σχέση	Από	Σε
unnamed	MongoDB Wire Protocol / TCP Port: 27017	MongoDB Database

IntelliQ Components



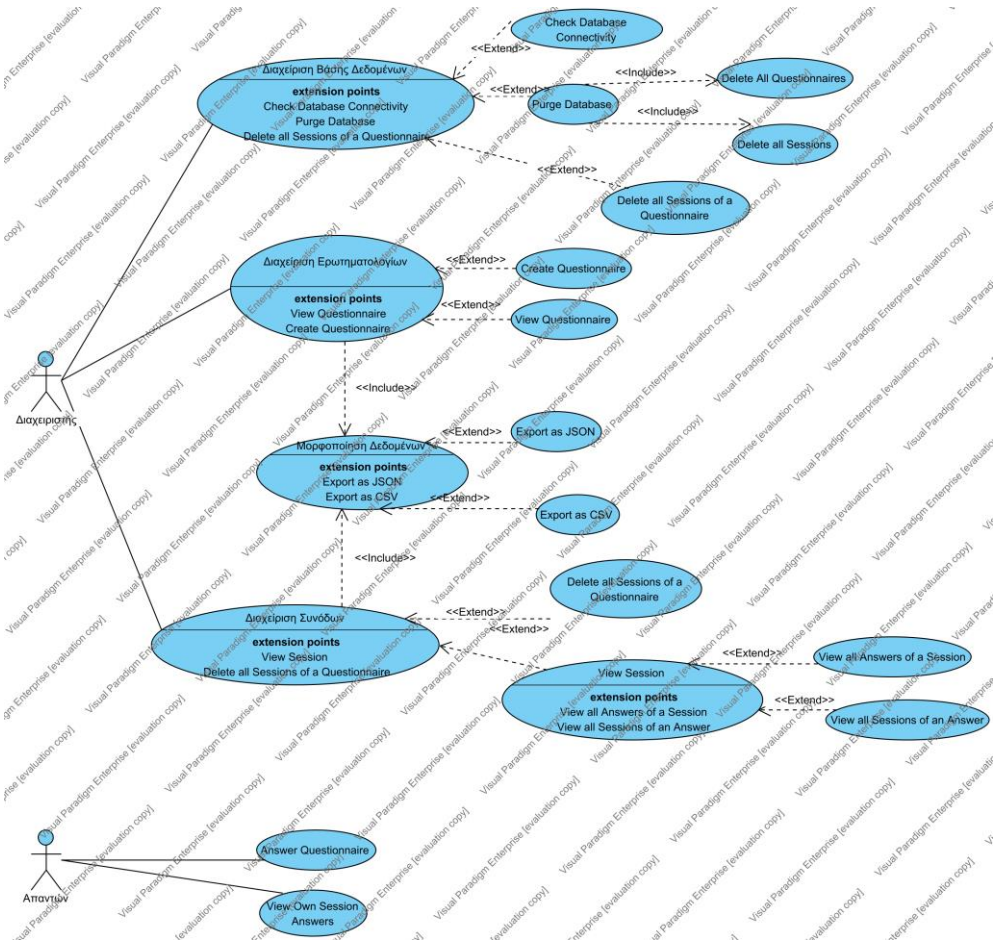
IntelliQ Deployment



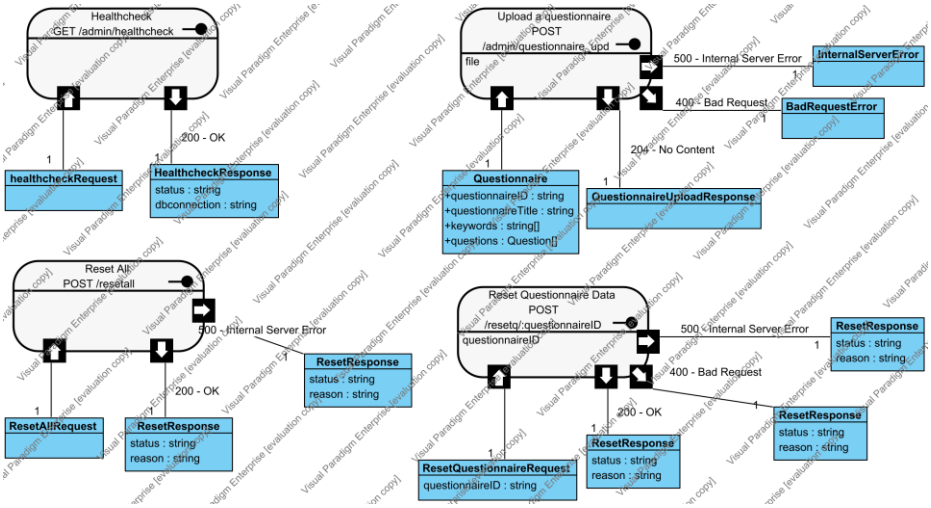
1.2.2 Διεπαφές με το χρήστη

Το λογισμικό έχει πλούσιες δυνατότητες επικοινωνίας με τον χρήστη, τις οποίες δείχνουμε με το Use Case Διάγραμμα και αναλυτικότερα με τα UML Class/API Διαγράμματα (διαχειριστή και υπόλοιπων χρηστών), αφού το μεγαλύτερο μέρος της επικοινωνίας χρήστη – λογισμικού στο σύστημά μας γίνεται μέσω αιτημάτων στα endpoints.

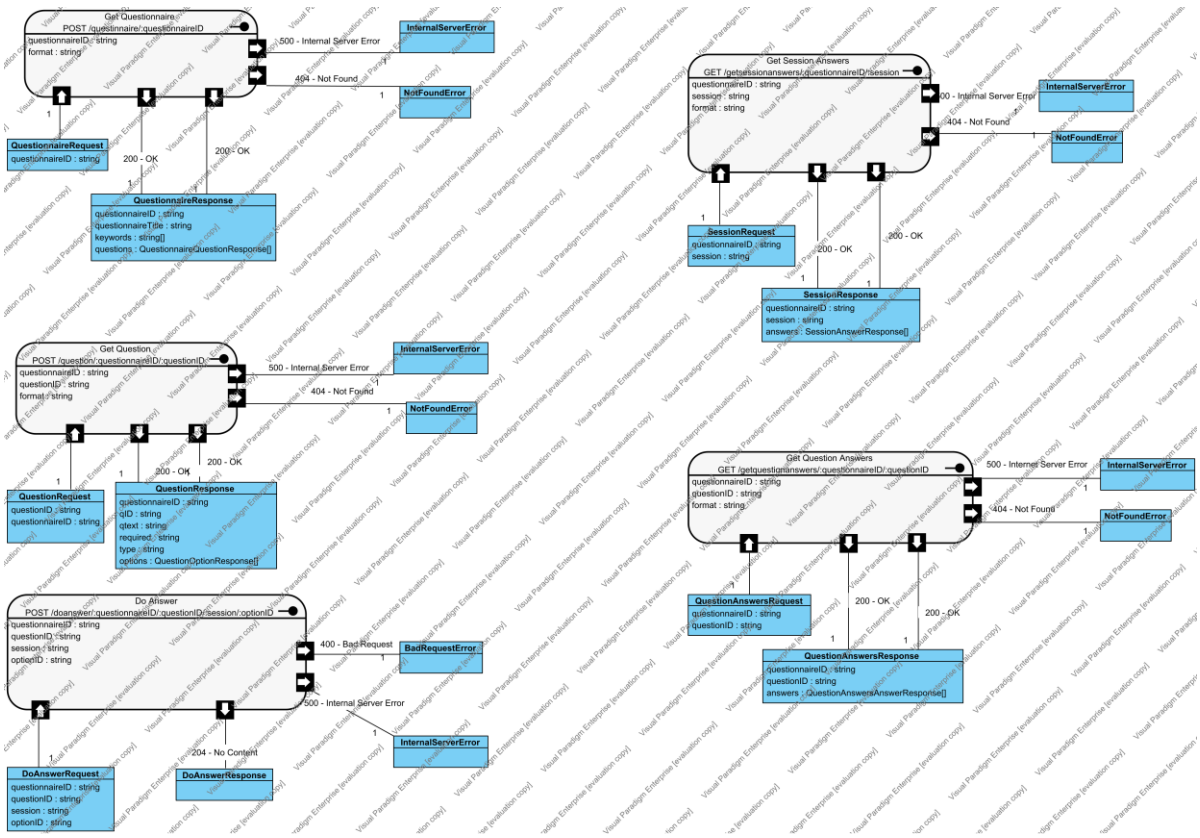
All Use Cases



Admin Endpoints



Functionality Endpoints



Ωστόσο, η διεπαφή επικοινωνίας με τους χρήστες δεν καλύπτει όλες αυτές τις δυνατότητες μιας και είμαστε διμελής ομάδα. Οι περιπτώσεις χρήσεις εξυπηρετούνται μέσω άμεσων HTTP μηνυμάτων από τον χρήστη με εξαίρεση τα Endpoints που εμπεριέχονται στο Use Case που υλοποιήσαμε και θα δείξουμε παρακάτω.

2. Αναφορές - πηγές πληροφοριών

Το backend είναι βασισμένο στην αποθήκευση δεδομένων σε mongoDB βάση, οπότε η κατανόησή του προϋποθέτει εξοικείωση με αυτό το «οικοσύστημα» καθώς και γνώσεις συνηθισμένων εργαλείων σχετικά με την λειτουργία backend εφαρμογών όπως διαχείριση JSON δεδομένων και κάποιας κατάλληλης γλώσσας προγραμματισμού (προτείνεται JavaScript που είναι και η δική μας επιλογή). Σχετικά θέματα που έχουν ενδιαφέρον είναι το middleware σε εφαρμογές, η επικύρωση εισερχόμενων δεδομένων (validation) και η έννοια του REST API.

Η κατανόηση της επικοινωνίας με την χρήστη μέσω frontend χρειάζεται βασικές γνώσεις χρήσης κάποιου κατάλληλου framework (εμείς διαλέξαμε το svelte, που είναι αρκετά εύχρηστο και «ελαφρύ»), HTML και CSS.

Παραθέτουμε μερικές πηγές για τα παραπάνω θέματα:

- MongoDB Documentation
- Working with JSON - Learn web development | MDN (mozilla.org)
- Using Express middleware (expressjs.com)

- What is REST - REST API Tutorial (restfulapi.net)
- Docs • Svelte
- HTML: HyperText Markup Language | MDN (mozilla.org)

3. Προδιαγραφές απαιτήσεων λογισμικού

3.1 Περιπτώσεις χρήσης

Υλοποιούμε μία περίπτωση χρήσης – μέσω της υλοποίησης της frontend εφαρμογής μας.

3.1.1 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 1: Answer Questionnaire

3.1.1.1 Χρήστες (ρόλοι) που εμπλέκονται

♀Απαντών

Το πρόσωπο που χρησιμοποιεί την εφαρμογή για να απαντήσει ερωτηματολόγια.

3.1.1.2 Προϋποθέσεις εκτέλεσης

Η σύνδεση του χρήστη σε Web Browser και η πρόσβασή του στο URL της frontend εφαρμογής μας ή, αν ένας χρήστης επιθυμεί να απαντά άμεσα στις ερωτήσεις ενός ερωτηματολογίου, αρκεί να στέλνει μηνύματα HTTP στα αντίστοιχα End Points.

3.1.1.3 Περιβάλλον εκτέλεσης

Η συγκεκριμένη περίπτωση χρήσης απαιτεί την διαθεσιμότητα όλων των κόμβων του συστήματός μας και για αυτό την διαλέξαμε. Αναφέρουμε πληροφορίες σχετικά με τους κόμβους που δείξαμε στο Deployment Διάγραμμα ώστε να καταστούν πιο σαφείς οι συνθήκες επιτυχούς εκτέλεσης του Use Case – Απάντηση σε Ερωτηματολόγιο.

User Device

Η συσκευή του χρήστη της εφαρμογής.

Front-End Web Server

Ο εξυπηρετητής που "τρέχει" την εφαρμογή.

Back-End Server

Ο εξυπηρετητής που τρέχει την backend εφαρμογή.

Database Server

Ο εξυπηρετητής που έχει τη βάση δεδομένων.

3.1.1.4 Δεδομένα εισόδου - εξόδου

3.1.1.4.1 Δεδομένα εισόδου

Ως δεδομένα εισόδου θεωρούνται οτιδήποτε υποβάλει ο χρήστης προς το σύστημα στη προσπάθειά του να απαντήσει ένα ερωτηματολόγιο.

- *DoAnswerRequest*

Το αντικείμενο υποβολής απάντησης σε μία ερώτηση.

- *QuestionnaireRequest*

Το αντικείμενο αίτησης για απόκτηση ενός ερωτηματολογίου.

- *QuestionRequest*

Το αντικείμενο αίτησης για ανάκτηση μιας ερώτησης ενός ερωτηματολογίου.

Επειδή η είσοδος γίνεται μέσω της εφαρμογής η μόνη εσφαλμένη επιλογή που μπορεί να δώσει ο χρήστης είναι λάθος τιμή αναγνωριστικού ερωτηματολογίου και αναγνωρίζεται από το query στη βάση δεδομένων το οποίο θα επιστρέψει μήνυμα λάθους.

3.1.1.4.2 Δεδομένα εξόδου

Ως δεδομένα εξόδου θεωρούμε οτιδήποτε μπορεί να μας επιστρέψει η εφαρμογή στα παραπάνω αιτήματα εισόδου.

- *DoAnswerResponse*

Η απάντηση στον χρήστη στη περίπτωση αυτού του Use Case είναι κενή.

- *QuestionnaireResponse*

Το αντικείμενο αυτό αφαίρεση του Questionnaire που περιγράφουμε στη βάση μας. Βλέπε σχετικό σύνδεσμο.

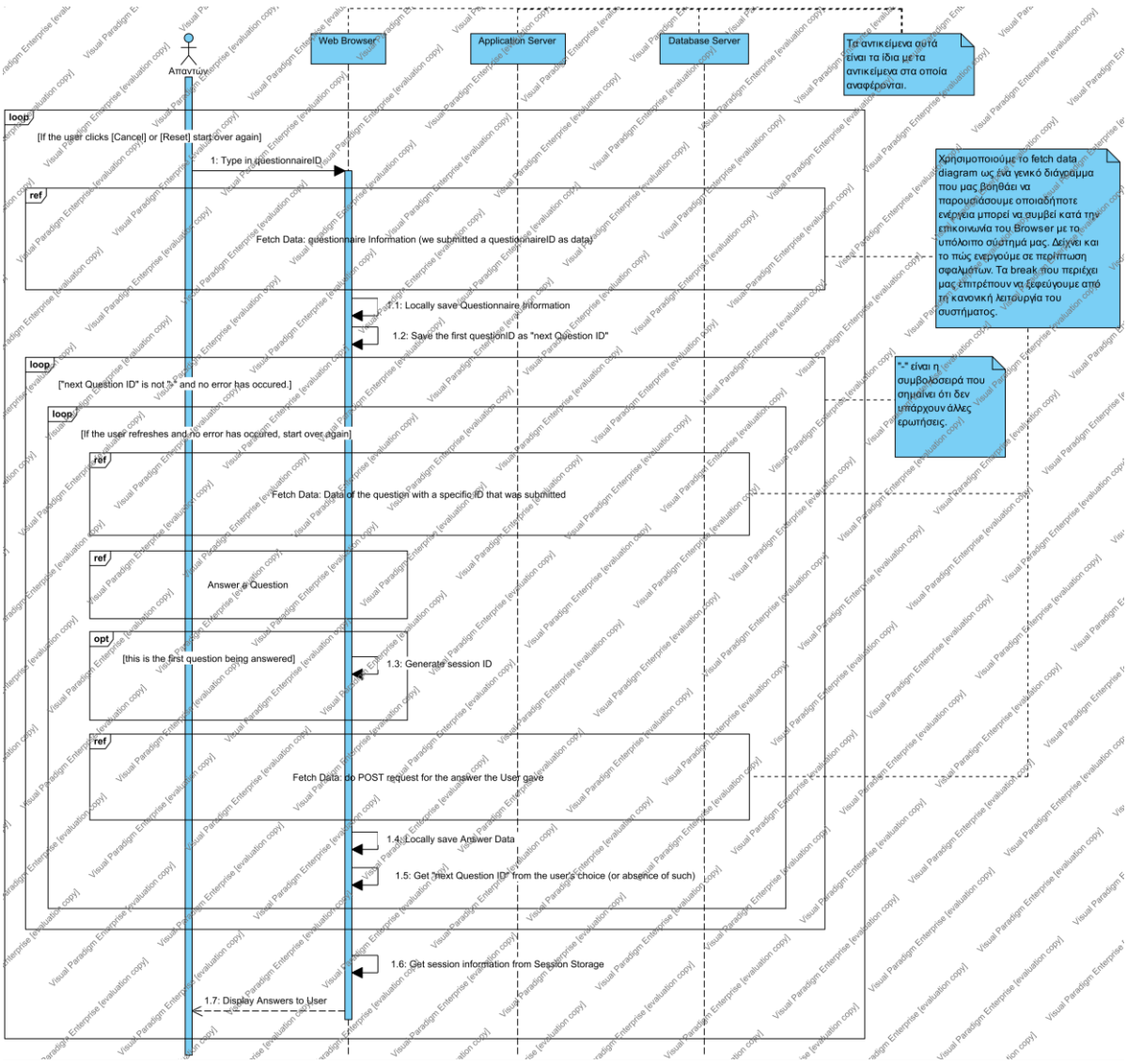
- *QuestionResponse*

Αντικείμενο με την ίδια πληροφορία με αυτό που έχουμε στη βάση μας έχοντας ως επιπλέον πεδίο το questionnairID του αντίστοιχου ερωτηματολογίου.

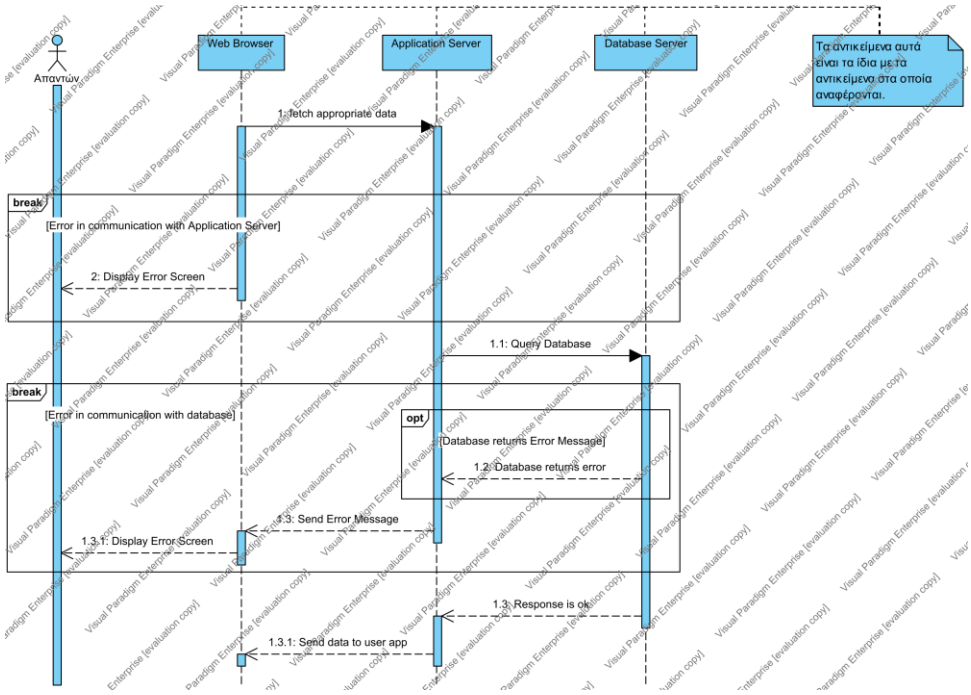
3.1.1.5 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

Δείχνουμε αρχικά, τα Sequence Διαγράμματα που εμπεριέχουν όλες τις δυνατές ενέργειες.

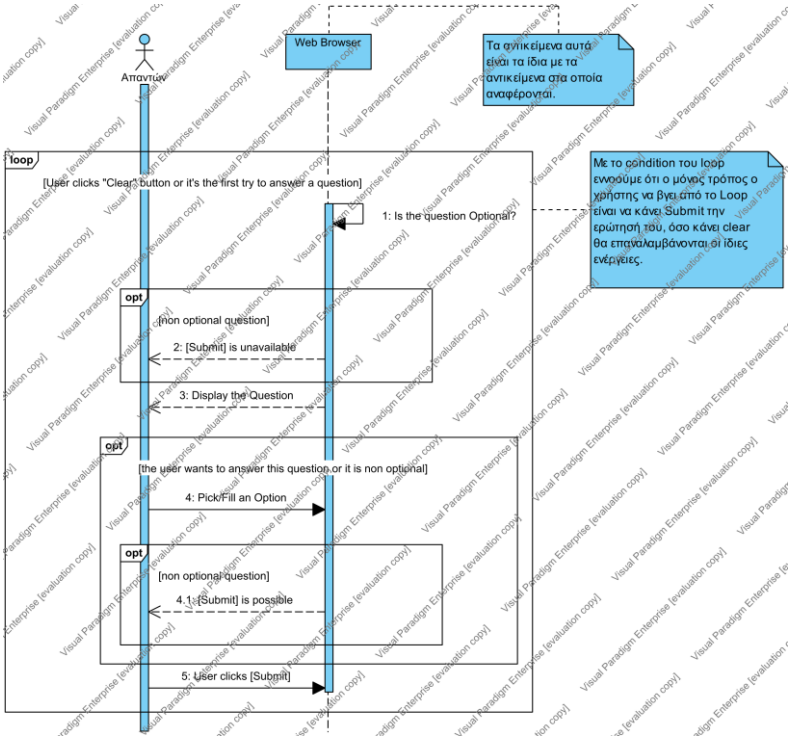
Answer Questionnaire Sequence



Fetch Data



Answer a Question



Και περιγράφουμε τα μηνύματα που ανταλλάσσονται μεταξύ των οντοτήτων στα διαγράμματα αυτά.

Answer Questionnaire Sequence

Μηνύματα

From	No.	Name	Type	Action Type	To	Async.
Απαντών	1	Type in questionnaireID	→	Unspecified	Web Browser	
Web Browser	1.1	Locally save Questionnaire Information		Unspecified	Web Browser	
Web Browser	1.2	Save the first questionID as "next Question ID"		Unspecified	Web Browser	
Web Browser	1.3	Generate session ID		Unspecified	Web Browser	
Web Browser	1.4	Locally save Answer Data		Unspecified	Web Browser	
Web Browser	1.5	Get "next Question ID" from the user's choice (or absence of such)		Unspecified	Web Browser	
Web Browser	1.6	Get session information from Session Storage		Unspecified	Web Browser	
Web Browser	1.7	Display Answers to User	→	Reply	Απαντών	

Fetch Data

Μηνύματα

From	No.	Name	Type	Action Type	To	Async.
Web Browser	1	fetch appropriate data	→	Unspecified	Application Server	
Web Browser	2	Display Error Screen	→	Reply	Απαντών	
Application Server	1.1	Query Database	→	Unspecified	Database Server	

From	No.	Name	Type	Action Type	To	Async.
Database Server	1.2	Database returns error	→	Reply	Application Server	
Application Server	1.3	Send Error Message	→	Reply	Web Browser	
Web Browser	1.3.1	Display Error Screen	→	Reply	Απαντών	
Database Server	1.3	Response is ok	→	Reply	Application Server	
Application Server	1.3.1	Send data to user app	→	Reply	Web Browser	

Answer a Question

Μηνύματα

From	No.	Name	Type	Action Type	To	Async.
Web Browser	1	Is the question Optional?	↩	Unspecified	Web Browser	
Web Browser	2	[Submit] is unavailable	→	Reply	Απαντών	
Web Browser	3	Display the Question	→	Reply	Απαντών	
Απαντών	4	Pick/Fill an Option	→	Unspecified	Web Browser	
Web Browser	4.1	[Submit] is possible	→	Reply	Απαντών	
Απαντών	5	User clicks [Submit]	→	Unspecified	Web Browser	

Η βασική λειτουργία μπορεί να περιγραφεί ως εξής:

1. Ο χρήστης δίνει ένα questionnaireID και αναμένει να λάβει την πρώτη ερώτηση αυτού του ερωτηματολογίου.
2. Η ερώτηση προβάλλεται στον χρήστη με κατάλληλη μορφή αναλόγως αν είναι open string ερώτηση ή/και πρέπει να απαντηθεί αναγκαστικά.

3. Ο χρήστης επιλέγει μια απάντηση ή τη συμπληρώνει μόνος του.
4. Υποβάλει την απάντησή του πατώντας το Submit κουμπί
    a. Προαιρετικό βήμα: Αν είναι η πρώτη απάντηση που υποβάλλεται, δημιούργησε ένα αναγνωριστικό συνόδου (session ID) για αυτό το γεγονός απάντησης, το οποίο θα χρησιμοποιήσεις και στις απαντήσεις σου στις υπόλοιπες ερωτήσεις του ερωτηματολογίου.
5. Αναμένει την προβολή της επόμενης ερώτησης.
6. Συνεχίζει από το 2. Μέχρι να μην υπάρχουν άλλες ερωτήσεις.
7. Τέλος, του προβάλλεται μια περίληψη του γεγονότος απαντήσεων που μόλις ολοκλήρωσε.

Ο χρήστης έχει πάντοτε διαθέσιμο τα Clear και Cancel κουμπιά. Τα οποία καθαρίζουν την επιλογή του στη τωρινή ερώτηση και ακυρώνουν όλη τη σύνοδο πηγαίνοντας στο βήμα 1 αντίστοιχα.

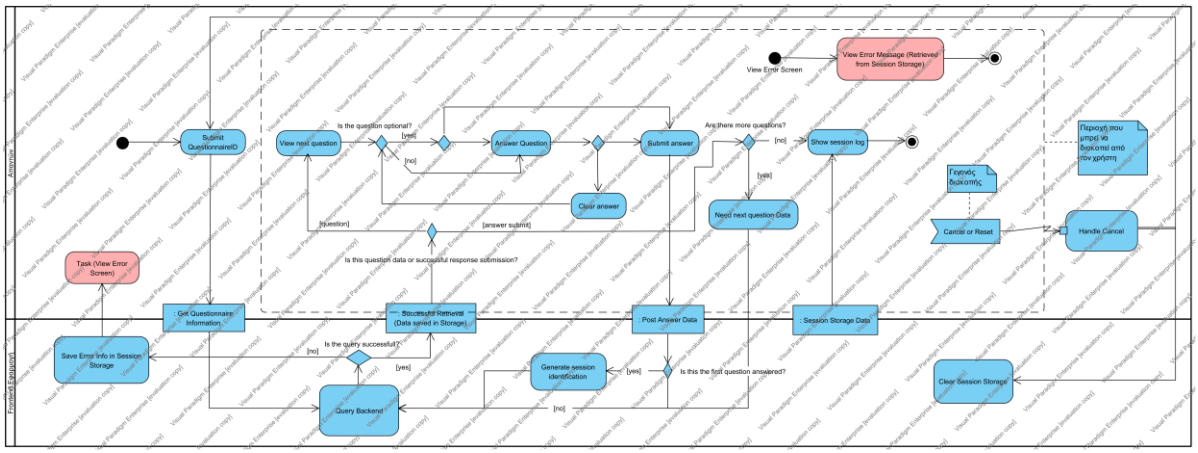
Στην οθόνη του βήματος 7 έχει διαθέσιμο το κουμπί Reset που τον επαναφέρει στο βήμα 1.

Σε περίπτωση σφάλματος (λάθος αίτημα ερωτηματολογίου ή πρόβλημα στη λειτουργία της εφαρμογής) ο χρήστης ειδοποιείται μέσω ειδικής οθόνης η οποία έχει Reset επιλογή που τον επαναφέρει στο βήμα 1.

Σχετικά με το βήμα 4a: η frontend εφαρμογή μας δοκιμάζει να δημιουργήσει αναγνωριστικό συνόδου 3 φορές σε περίπτωση που το μήνυμα που λάβει από το backend δίνει την εντύπωση ότι το αναγνωριστικό που δημιούργησε αντιστοιχεί σε ήδη αποθηκευμένη σύνοδο (που έχει γίνει στο παρελθόν από τον ίδιο ή άλλο χρήστη). Προφανώς, όλες οι απαντήσεις που αναφέρονται σε ένα ερωτηματολόγιο και κατά τη διάρκεια ενός γεγονότος απάντησης αναφέρουν το ίδιο αναγνωριστικό συνόδου.

Σχηματικά, βλέπουμε τα εξής:

Answer Questionnaire



3.1.1.7 Δεδομένα εξόδου

Τα δεδομένα εξόδου εκτός από τα προηγούμενα αντικείμενα που αφορούσαν την πλευρά του χρήστη περιλαμβάνουν και τα αντικείμενα απάντησης που αποθηκεύονται από κάθε

υποβολή απάντησης. Το πώς/πότε παράγονται φαίνεται στα διαγράμματα (Fetch Data διάγραμμα το οποίο έχεις ως δεδομένα την απάντηση του χρήστη και είναι εμφανές Answer Questionnaire διάγραμμα). Περιγράφουμε και αυτό το αντικείμενο:

Answer

Ένα έγγραφο (document) της βάσης δεδομένων. Η εγγραφή αυτή χρησιμοποιείται για να περιγράψει τις απαντήσεις των χρηστών στις ερωτήσεις μας.

Περίληψη Πεδίων

Όνομα	Περιγραφή
_uniqueID	Το αναγνωριστικό της απάντησής μας.
ans	Η απάντηση που υποβλήθηκε στην ερώτηση.
qID	Η ερώτηση στην οποία αναφέρεται αυτή η απάντηση.
questionnaireID	Το ερωτηματολόγιο στο οποίο αναφέρεται αυτή η απάντηση.
session	Το γεγονός συνόδου στο οποίο αναφερόμαστε.

3.1.1.8 Παρατηρήσεις

Η frontend εφαρμογή παρουσιάζει στο τέλος και περίληψη των απαντήσεων του χρήστη στο ερωτηματολόγιο που προηγήθηκε, οπότε υλοποιεί σε μερικό βαθμό και το " View Own Session Answers" Use Case.

3.2 Απαιτήσεις επιδόσεων

Το λογισμικό πρέπει να ανταποκρίνεται στους χρήστες και να δίνει κατάλληλα μηνύματα σε κάθε περίπτωση λάθους εντός εύλογου χρονικού διαστήματος. Η διάταξή του πρέπει να είναι εύληπτη και εύχρηστη. Το backend έχει σχεδιαστεί ώστε να επιστρέφει μηνύματα λαθών που αφορούν τη βάση και το frontend εμφανίζει τα μηνύματα αυτά αλλά και μηνύματα γενικής αποτυχίας, όπως μη διαθεσιμότητα του backend γενικά.

Έχουν χρησιμοποιηθεί αρκετά «ελαφριά» εργαλεία για την δημιουργία του και είναι πρόγραμμα που δεν απαιτεί ιδιαίτερους υπολογιστικούς πόρους, οπότε υποστηρίζονται αρκετοί πολλαπλοί χρήστες – αναλόγως τις δυνατότητες του hosting server να τους εξυπηρετήσει. Προς το σκοπό αυτό έχουμε προσπαθήσει και στο backend και στο frontend να αξιοποιήσουμε όσο το δυνατό περισσότερο τις δυνατότητες ασύγχρονης λειτουργίας της JavaScript.

Το λογισμικό είναι κλιμακώσιμο καθώς το API του έχει σχεδιαστεί ώστε να υπακούει στις αρχές του RESTful API και μπορεί να επεκταθεί με middleware. Επίσης, αν δεν επιθυμούμε παρέμβαση στον πηγαίο κώδικα, πιθανό bottleneck ίσως αποτελεί η βάση δεδομένων, ωστόσο, και αυτή μπορεί να αντικατασταθεί με ό,τι κρίνει ο χρήστης κατάλληλο. Αρκεί να υποστηρίζει interface παρόμοιο με της MongoDB.

Το λογισμικό παρουσιάζει συμβατότητα με πληθώρα συσκευών, αφού μοναδική του απαίτηση είναι η πρόσβαση στον διαδικτυακό σύνδεσμο του frontend. Αρκεί ο χρήστης να έχει πρόσβαση σε έναν Web Browser.

Το backend έχει τη δυνατότητα να επιστρέφει πληθώρα μεταδεδομένων για τις πληροφορίες που επεξεργάζεται, οπότε βοηθάει τον χρήστη στο να διαπιστώσει πιθανά bottlenecks και γενικά να κατανοήσει τη πραγματικού – χρόνου συμπεριφορά του.

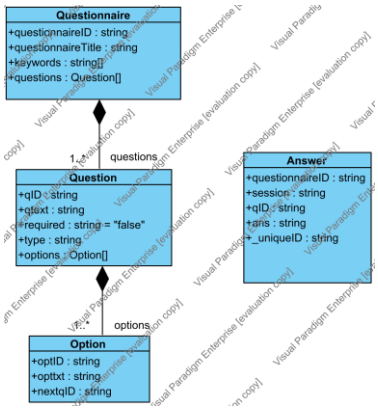
Από testing που πραγματοποιήσαμε όσον αφορά τη χρήση του συνολικού συστήματος (μέσω του frontend ενώ συνδέεται με το υπόλοιπο λογισμικό όπως περιγράφει το Deployment Diagram) μάλλον η περίπτωση που παρουσιάζει μεγαλύτερη της επιθυμητής καθυστέρηση είναι η αποστολή αιτήματος από το frontend ενώ το backend είναι εκτός λειτουργίας. Βέβαια, ο λόγος είναι ότι το frontend περιμένει ένα εύλογο χρονικό διάστημα μέχρι να διαπιστώσει ότι το backend δεν ανταποκρίνεται.

3.3 Απαιτήσεις οργάνωσης δεδομένων

3.3.1 Απαιτήσεις και περιορισμοί πρόσβασης σε δεδομένα

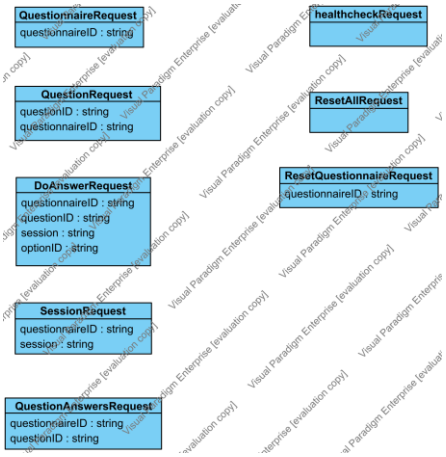
Δείχνουμε τις δομές δεδομένων που χρησιμοποιεί το λογισμικό μας.

Database Schema

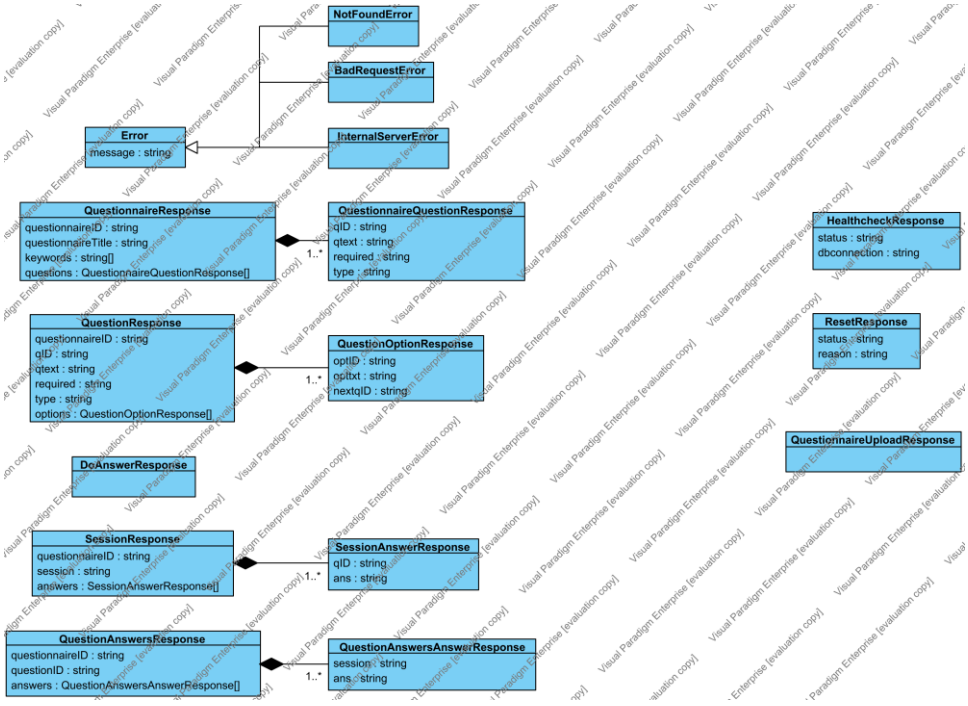


Και οι δομές των αιτημάτων (αριστερά βρίσκονται αυτές που αναφέρονται σε λειτουργικά endpoints και δεξιά αυτές που αναφέρονται σε διαχειριστικά endpoints).

Request Bodies



Response Bodies



Δεν έχουν υλοποιηθεί δυνατότητες ασφάλειας/διαχωρισμού διαχειριστών από τους υπόλοιπους χρήστες, οπότε οποιοσδήποτε μπορεί να λειτουργήσει ως κάποιος από τους δύο ρόλους υποβάλλοντας HTTP requests στο API μας. Το frontend απευθύνεται μόνο σε χρήστες – απαντώντες στα ερωτηματολόγια.

Οι παραπάνω δομές πρέπει να είναι μοντελοποιημένες ως JSON (Content-Type: application/json) με εξαίρεση τα Response Bodies στη περίπτωση επιτυχίας λειτουργικών endpoints που μπορεί να είναι csv αν ο χρήστης συμπεριλάβει την προαιρετική query παράμετρο “format” με τιμή csv (Content-Type: text/csv).

3.5 Λοιπές απαιτήσεις

3.5.1 Απαιτήσεις διαθεσιμότητας λογισμικού

Η εφαρμογή είναι επιθυμητό να είναι διαθέσιμη καθημερινά για όλους τους χρήστες μας. Συνεπώς, οι εξυπηρετητές που τρέχουν το Frontend Application, το Backend και τη βάση πρέπει να υποστηρίζουν αυτή τη λειτουργία.

Επίσης, η εφαρμογή πρέπει να τηρεί συνήθη standard ανταλλαγής HTTP(S) μηνυμάτων μέσω διαδικτύου για να συμφωνεί με οποιονδήποτε ISP μπορεί να έχει κάποιος χρήστης μας.

3.5.2 Απαιτήσεις ασφάλειας

Δεν έχει υλοποιηθεί login ή/και προστασία των δεδομένων. Η ασφάλεια στηρίζεται στο ήθος των χρηστών μας.

Το frontend επικοινωνεί με τους User Browsers μέσω https ως μια μικρή απόπειρα παροχής υπηρεσιών ασφαλείας.