ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΤΜΗΜΑΤΟΣ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΚΑΤΑΣΚΕΥΗ ΕΡΓΑΛΕΙΟΥ ΜΕΤΡΗΣΗΣ ΠΟΙΟΤΗΤΑΣ ΛΟΓΙΣΜΙΚΟΥ

Διπλωματική Εργασία

του/της

Αναστάσιος Τιλσίζογλου

Θεσσαλονίκη, Ιούνιος 2023

ΚΑΤΑΣΚΕΥΗ ΕΡΓΑΛΕΙΟΥ ΜΕΤΡΗΣΗΣ ΠΟΙΟΤΗΤΑΣ ΛΟΓΙΣΜΙΚΟΥ

Αναστάσιος Τιλσίζογλου

Πτυχίο Μηχανικών Πληροφορικής, ΑΤΕΙ Κεντρικής Μακεδονίας, 2018

Διπλωματική Εργασία

υποβαλλόμενη για τη μερική εκπλήρωση των απαιτήσεων του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΤΙΤΛΟΥ ΣΠΟΥΔΩΝ ΣΤΗΝ ΕΦΑΡΜΟΣΜΕΝΗ ΠΛΗΡΟΦΟΡΙΚΗ

Επιβλέπων Καθηγητής
Απόστολος Αμπατζόγλου

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την ηη/μμ/εεεε

Αμπατζόγλου Απόστολος	Χατζηγεωργίου Αλέξανδρος	Ξυνόγαλος Στυλιανός
.....	.....	.....

Τιλσίζογλου Αναστάσιος (mai22071)

.....

Περίληψη

Η παρούσα διπλωματική εργασία επικεντρώνεται στην κατασκευή ενός εργαλείου μέτρησης ποιότητας λογισμικού. Ο σκοπός της έρευνας είναι η μετατροπή μιας υπάρχουσας εφαρμογής μέτρησης ποιότητας λογισμικού από πηγαίο κώδικα σε Web Service. Αυτό το Web Service θα καλείται από την υπάρχουσα πλατφόρμα SDK4ED.

Η πλατφόρμα SDK4ED έχει ως στόχο να ελαχιστοποιήσει το κόστος, τον χρόνο ανάπτυξης και την πολυπλοκότητα των διαδικασιών ανάπτυξης λογισμικού χαμηλής ενέργειας. Παρέχει εργαλεία για αυτόματη βελτιστοποίηση πολλαπλών απαιτήσεων ποιότητας, όπως τεχνικό χρέος, ενεργειακή απόδοση, αξιοπιστία και απόδοση.

Πέρα από τη μετατροπή της εφαρμογής σε Web Service, η έρευνα περιλαμβάνει την υλοποίηση της λειτουργίας του υπολογισμού μετρικών μόνο για τις αλλαγμένες κλάσεις για κάθε commit. Αυτό σημαίνει ότι οι μετρικές υπολογίζονται μόνο για τον κώδικα που έχει αλλάξει ανάμεσα στις διαδοχικές εκδόσεις της εφαρμογής. Επιπλέον, θα γίνει προσπάθεια αλλαγής σε 2 τμήματα της μεθοδολογίας. Το πρώτο αφορά τη σταθερά K και θα είναι σταθμισμένη ανάλογα με τον μέσο όρο αλλαγμένων γραμμών κώδικα των πιο πρόσφατων commits σε σχέση με τα παλαιότερα, αντί για τον μέσο όρο όλων των commits.

Λέξεις Κλειδιά: εργαλείο μέτρησης ποιότητα, λογισμικό, τεχνικό χρέος, web service, μετρικές, SDK4ED, κόστος, χρόνος ανάπτυξης, πολυπλοκότητα, ανάπτυξη λογισμικού

Abstract

The present thesis focuses on the construction of a software quality measurement tool. The aim of the research is to convert an existing software quality measurement application from source code to a Web Service. This Web Service will be called by the existing SDK4ED platform.

The SDK4ED platform aims to minimize the cost, development time, and complexity of low-energy software development processes. It provides tools for automated optimization of multiple quality requirements, such as technical debt, energy efficiency, reliability, and performance.

In addition to converting the application into a Web Service, the research includes implementing the functionality of computing metrics only for the modified classes for each commit. This means that metrics are calculated only for the code that has changed between successive versions of the application. Furthermore, efforts will be made to change two parts of the methodology. The first concerns the constant and will be weighted based on the average number of changed lines of code in the most recent commits compared to the older ones, instead of the overall average of all commits.

Keywords: software, quality, measurement, tool, technical debt, web service, metrics, SDK4ED, cost, development time, complexity, software development.

Πρόλογος – Ευχαριστίες

Θα ήθελα να αφιερώσω αυτήν την εργασία σε όλους τους σημαντικούς ανθρώπους στη ζωή μου, και ιδιαίτερα στην οικογένειά μου, που με έχουν στηρίξει καθ' όλη τη διάρκεια της ζωής μου. Από τη γέννησή μου μέχρι σήμερα, στη σχολική και ακαδημαϊκή μου πορεία, στις επιτυχίες, αλλά και στις αποτυχίες, στις χαρές και τις λύπες, αυτά τα πρόσωπα βρίσκονται πάντα δίπλα μου για να μου δώσουν ώθηση και στήριξη. Είναι αυτοί που με βοηθούν να εξελίσσομαι ως άνθρωπος, να αναπτύσσομαι ως συνειδητοποιημένο μέλος της κοινωνίας μας, αλλά και να επιτυγχάνω ως επαγγελματίας στον τομέα που επέλεξα να σπουδάσω και να ακολουθήσω.

Ευχαριστώ θερμά όλους τους καθηγητές του τμήματος μου για την αμέριστη βοήθεια και υποστήριξή τους κατά τη διάρκεια των σπουδών μου. Με τις αξιόλογες γνώσεις που μου παρείχαν και το κατάλληλο κίνητρο που μου μετέφεραν, έχω αποκτήσει την αυτοπεποίθηση και την απαιτούμενη αυτονομία για να αντιμετωπίζω με επιτυχία τις προκλήσεις που προκύπτουν στον επαγγελματικό μου δρόμο. Οφείλω σε αυτούς την ανάπτυξη των δεξιοτήτων μου και την απόκτηση των γνώσεων που μου επιτρέπουν να είμαι ανταγωνιστικός στο χώρο της εργασίας μου.

Τέλος, θα ήθελα να ευχαριστήσω ιδιαίτερα τον επιβλέποντα καθηγητή μου, κ. Απόστολο Αμπατζόγλου, καθώς και τον υποψήφιο διδάκτορα Νικόλαο Νικολαΐδη, για την αμέριστη και ουσιαστική καθοδήγηση που μου παρείχαν κατά τη διάρκεια της έρευνας μου. Η ενθάρρυνση και οι συμβουλές τους μου επέτρεψαν να εστιάσω στα σημαντικά στοιχεία της έρευνάς μου και να αναπτύξω τις ικανότητές μου περαιτέρω. Τους ευχαριστώ θερμά για τις εξαιρετικά ωφέλιμες κριτικές παρατηρήσεις τους και τη στήριξή τους στο έργο μου.

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

1 Εισαγωγή

1.1 Πρόβλημα – Σημαντικότητα του θέματος

Η παρούσα διπλωματική εργασία ασχολείται με την κατασκευή ενός εργαλείου μέτρησης ποιότητας λογισμικού. Το κύριο χαρακτηριστικό αυτού του εργαλείου είναι η ικανότητα να μετρά και αξιολογεί την ποιότητα του λογισμικού με βάση συγκεκριμένες μετρικές. Η μέτρηση της ποιότητας του λογισμικού είναι ένα σημαντικό θέμα στον τομέα της ανάπτυξης λογισμικού, καθώς βοηθά στην αξιολόγηση της απόδοσης, της αξιοπιστίας και της επίδοσης του λογισμικού.

Ένα ενδιαφέρον στοιχείο αυτής της εργασίας είναι η μετατροπή μιας υπάρχουσας εφαρμογής μέτρησης ποιότητας λογισμικού από πηγαίο κώδικα σε ένα Web Service. Αυτό επιτρέπει την πρόσβαση και τη χρήση του εργαλείου μέσω του δικτύου, κάνοντάς το πιο ευέλικτο και ευκολόχρηστο για τους χρήστες. Επιπλέον, η διεπαφή μεταξύ αυτού του εργαλείου μέτρησης ποιότητας και της υπάρχουσας πλατφόρμας SDK4ED προσφέρει ακόμα μεγαλύτερες δυνατότητες για την ανάλυση και παρακολούθηση της ποιότητας του λογισμικού κατά τη διαδικασία ανάπτυξης. Η πλατφόρμα SDK4ED, στην οποία το Web Service απευθύνεται, έχει σχεδιαστεί με σκοπό να ελαχιστοποιήσει το κόστος, τον χρόνο ανάπτυξης και την πολυπλοκότητα των διαδικασιών ανάπτυξης λογισμικού χαμηλής ενέργειας.

Η διπλωματική εργασία επικεντρώνεται επίσης στην υλοποίηση της λειτουργίας υπολογισμού μετρικών μόνο για τις αλλαγμένες κλάσεις για κάθε commit. Αυτό σημαίνει ότι οι μετρικές υπολογίζονται μόνο για τον κώδικα που έχει αλλάξει ανάμεσα στις διαδοχικές εκδόσεις της εφαρμογής. Αυτή η προσέγγιση επιτρέπει την εστίαση στις συγκεκριμένες αλλαγές που έχουν γίνει στον κώδικα και μπορεί να βοηθήσει στην ανίχνευση προβλημάτων και τη βελτίωση της ποιότητας του λογισμικού.

Γενικά, η διπλωματική εργασία προσπαθεί να συνδυάσει τη μετρική ποιότητας λογισμικού με την ανάπτυξη εργαλείων και πλατφορμών που επιτρέπουν την ανάλυση και τη βελτίωση της ποιότητας του λογισμικού. Ο συνδυασμός του εργαλείου μέτρησης ποιότητας λογισμικού και της πλατφόρμας SDK4ED παρέχει ένα ολοκληρωμένο περιβάλλον για την αξιολόγηση και τη βελτίωση της ποιότητας του λογισμικού κατά τη διαδικασία ανάπτυξης.

Συνοψίζοντας, η διπλωματική εργασία εστιάζει στην κατασκευή ενός εργαλείου μέτρησης ποιότητας λογισμικού, με έμφαση στη μετατροπή του σε ένα Web Service και την ενσωμάτωσή του στην πλατφόρμα SDK4ED. Με αυτό το εργαλείο, οι μηχανικοί λογισμικού

μπορούν να μετρούν και να αξιολογούν την ποιότητα του λογισμικού τους βασιζόμενοι σε συγκεκριμένες μετρικές, ενώ η ενσωμάτωσή του στην πλατφόρμα SDK4ED επιτρέπει την εντατική παρακολούθηση της ποιότητας κατά τη διαδικασία ανάπτυξης.

1.2 Σκοπός – Στόχοι

Σκοπός της παρούσας διπλωματικής είναι η κατασκευή ενός εργαλείου μέτρησης ποιότητας λογισμικού με τη χρήση προηγμένων τεχνικών και μεθόδων. Οι σκοποί και οι παρατηρήσεις της έρευνας εστιάζουν στην ανάπτυξη ενός αξιόπιστου και αποτελεσματικού εργαλείου που θα επιτρέπει την αξιολόγηση και τη βελτίωση της ποιότητας του λογισμικού.

Το υπόβαθρο της έρευνας περιλαμβάνει την αναγνώριση της ανάγκης για αξιόπιστες και αποτελεσματικές μεθόδους μέτρησης ποιότητας λογισμικού. Η ποιότητα του λογισμικού έχει κρίσιμη σημασία για την επίτευξη των στόχων αξιοπιστίας, απόδοσης και ικανοποίησης των χρηστών. Ωστόσο, η αξιολόγηση της ποιότητας είναι πολύπλοκη και απαιτεί εξειδικευμένες μεθόδους και εργαλεία.

Το πλαίσιο της έρευνας καθορίζεται από τις σύγχρονες τάσεις στον τομέα της ποιότητας του λογισμικού και της μέτρησής της. Συνοπτικά, αυτό περιλαμβάνει την ανάλυση των υφιστάμενων μεθόδων μέτρησης ποιότητας λογισμικού, των προτύπων ποιότητας και των μετρήσιμων κριτηρίων που χρησιμοποιούνται στη βιομηχανία. Επιπλέον, το πλαίσιο της έρευνας περιλαμβάνει την εξέταση των προκλήσεων και των προβλημάτων που σχετίζονται με τη μέτρηση της ποιότητας του λογισμικού, καθώς και τις ευκαιρίες για βελτίωση και καινοτομία στον τομέα αυτό.

Μέσω αυτής της έρευνας, θα δημιουργηθεί ένα πλήρες και αξιόπιστο εργαλείο μέτρησης ποιότητας λογισμικού, το οποίο θα παρέχει αντικειμενικές μετρήσεις και αξιολογήσεις για τις διάφορες πτυχές της ποιότητας. Το εργαλείο αυτό αναμένεται να συμβάλει στη βελτίωση της ποιότητας του λογισμικού και να παρέχει στους ανάδοχους και ανάπτυξης λογισμικού ένα αξιόπιστο μέσο για την αξιολόγηση των προϊόντων τους.

1.3 Συνεισφορά

Η συνεισφορά της συγκεκριμένης μελέτης έχει διάφορα στοιχεία:

1. Κατασκευή ενός εργαλείου μέτρησης ποιότητας λογισμικού: Η μελέτη αναφέρεται στην ανάπτυξη ενός εργαλείου που μπορεί να μετρήσει και αξιολογήσει την ποιότητα του λογισμικού βάσει συγκεκριμένων μετρικών. Αυτό το εργαλείο είναι κρίσιμο για τη μελέτη και αποτελεί τον πυρήνα της συνεισφοράς.

2. Μετατροπή σε ένα Web Service: Η μετατροπή του εργαλείου μέτρησης ποιότητας από τον πηγαίο κώδικα σε ένα Web Service αποτελεί ένα σημαντικό μέρος της συνεισφοράς. Αυτή η μετατροπή επιτρέπει την πρόσβαση και τη χρήση του εργαλείου μέσω του δικτύου, καθιστώντας το πιο ευέλικτο και ευκολόχρηστο για τους χρήστες.
3. Διεπαφή με την πλατφόρμα SDK4ED: Η διεπαφή μεταξύ του εργαλείου μέτρησης ποιότητας και της υπάρχουσας πλατφόρμας SDK4ED αποτελεί ένα ακόμα σημαντικό στοιχείο της συνεισφοράς. Αυτή η διεπαφή επιτρέπει την ολοκλήρωση του εργαλείου μέτρησης στο πλαίσιο της πλατφόρμας SDK4ED και επιτρέπει στους χρήστες να εκτελούν αξιολογήσεις ποιότητας κατά τη διάρκεια της ανάπτυξης λογισμικού.
4. Αξιολόγηση με χρήση πραγματικών περιπτώσεων: Η μελέτη αξιολογεί την απόδοση και την ακρίβεια του εργαλείου μέτρησης ποιότητας με χρήση πραγματικών περιπτώσεων. Αυτό είναι σημαντικό για να διασφαλιστεί ότι το εργαλείο λειτουργεί σωστά και παρέχει χρήσιμα αποτελέσματα.

1.4 Βασική Ορολογία

Βελτιώνοντας την ποιότητα του λογισμικού, θα πρέπει να κατανοήσουμε και να χρησιμοποιήσουμε τη σχετική ορολογία. Παρακάτω θα βρείτε μερικούς βασικούς όρους που συνδέονται με την εργασία "Κατασκευή Εργαλείου Μέτρησης Ποιότητας Λογισμικού":

- Ποιότητα λογισμικού: Αναφέρεται στην ικανότητα ενός λογισμικού να ικανοποιεί τις απαιτήσεις, να εκτελείται σωστά και να παρέχει τις αναμενόμενες λειτουργίες και επίδοση.
- Μέτρηση ποιότητας: Αναφέρεται στην αξιολόγηση και αναγνώριση καταλληλότητας, αξιοπιστίας και απόδοσης του λογισμικού με τη χρήση μετρικών και δεικτών.
- Μέθοδοι μέτρησης ποιότητας: Αναφέρονται στις διαδικασίες και τεχνικές που χρησιμοποιούνται για να αξιολογηθεί η ποιότητα ενός λογισμικού. Αυτές οι μέθοδοι συνήθως περιλαμβάνουν τη συλλογή και ανάλυση μετρήσιμων δεδομένων, όπως η απόδοση, η αξιοπιστία και η ασφάλεια του λογισμικού, με σκοπό να παρέχουν αντικειμενικά κριτήρια για την ποιότητα του.
- Τεχνικό χρέος λογισμικού: Αναφέρεται στην αύξηση του κόστους ή των προβλημάτων που προκύπτουν στο μέλλον λόγω της επιλογής να γίνουν γρήγορες και προσωρινές λύσεις στον κώδικα λογισμικού κατά τη διάρκεια της ανάπτυξης. Αυτό μπορεί να οφείλεται σε περιορισμένο χρόνο ή πόρους, αλλά μπορεί να οδηγήσει σε προβλήματα

όπως ανεπαρκή λειτουργικότητα, δυσκολίες συντήρησης και αυξημένο κόστος αναπτύξεων στο μέλλον.

- Μετρικές τεχνικού χρέους: Μετρήσεις που χρησιμοποιούνται για να αξιολογήσουν την ποιότητα του κώδικα και την ύπαρξη τεχνικού χρέους, όπως η πολυπλοκότητα του κώδικα, οι ανεπάρκειες στη σχεδίαση ή η ύπαρξη μη ορθογραφημένου κώδικα. Αυτές οι μετρικές παρέχουν πληροφορίες για το επίπεδο τεχνικού χρέους και μπορούν να βοηθήσουν στη λήψη αποφάσεων για την αναδιοργάνωση, τη βελτίωση και τη συντήρηση του λογισμικού.

1.5 Διάρθρωση της μελέτης

Στην εισαγωγή της διπλωματικής εργασίας, παρουσιάζεται το πρόβλημα που αντιμετωπίζουμε και εξηγείται η σημασία του θέματος. Καθορίζονται επίσης οι στόχοι της έρευνας και η συνεισφορά της στον επιστημονικό τομέα. Επιπλέον, παρέχεται μια εισαγωγή στην ορολογία που χρησιμοποιείται και περιγράφεται η δομή της μελέτης.

Η βιβλιογραφική επισκόπηση και το θεωρητικό υπόβαθρο παρουσιάζουν μια εκτενή ανασκόπηση της βιβλιογραφίας που σχετίζεται με το θέμα. Εξηγείται η έννοια της τεχνολογίας λογισμικού, της ποιότητας του λογισμικού και του τεχνικού χρέους. Επιπλέον, αναλύονται μετρικές τεχνολογίας λογισμικού και περιγράφονται οι μετρήσεις που γίνονται για διάφορες πτυχές του λογισμικού. Τέλος, εξηγείται η σημασία των υπηρεσιών του web.

Η μεθοδολογία περιγράφει την προσέγγιση που ακολουθήθηκε για την υλοποίηση της διπλωματικής εργασίας. Αναλύονται οι τεχνολογίες και τα εργαλεία που χρησιμοποιήθηκαν και παρουσιάζεται μια ανάλυση του backend και του frontend της εφαρμογής.

Τέλος, γίνεται μια σύνοψη της διπλωματικής εργασίας και παρουσιάζονται τα συμπεράσματα από τα αποτελέσματα της έρευνας. Συζητούνται επίσης τα όρια και οι περιορισμοί της έρευνας και προτείνονται πιθανές μελλοντικές επεκτάσεις για περαιτέρω έρευνα.

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

2.1 Τι είναι τεχνολογία λογισμικού;

Η τεχνολογία λογισμικού αναφέρεται στο σύνολο των μεθόδων, εργαλείων, τεχνικών και διαδικασιών που χρησιμοποιούνται για την ανάπτυξη, τη διαχείριση και τη συντήρηση λογισμικού. Αποτελεί ένα σημαντικό πεδίο στην επιστήμη του υπολογιστών και έχει επιρροές από πολλούς τομείς, όπως η μηχανική, η μαθηματική ανάλυση και η διοίκηση έργων.

Η τεχνολογία λογισμικού περιλαμβάνει διάφορες πτυχές της διαδικασίας ανάπτυξης λογισμικού, όπως η ανάλυση απαιτήσεων, ο σχεδιασμός αρχιτεκτονικής, ο προγραμματισμός, η δοκιμή και η συντήρηση. Επίσης, περιλαμβάνει τεχνικές για τη διαχείριση του λογισμικού, την ανάπτυξη ομάδων εργασίας και την παρακολούθηση της ποιότητας.

Οι επιστημονικές έρευνες, τα εγχειρίδια και τα βιβλία που αναφέρονται στην τεχνολογία λογισμικού παρέχουν βασικές γνώσεις και πληροφορίες για τις βέλτιστες πρακτικές και τις προηγμένες μεθόδους που χρησιμοποιούνται στην ανάπτυξη και διαχείριση λογισμικού. Ορισμένες αξιόλογες πηγές που μπορείτε να ανατρέξετε για περαιτέρω μελέτη είναι οι εξής:

- "Software Engineering: A Practitioner's Approach" από Roger S. Pressman.
- "Clean Code: A Handbook of Agile Software Craftsmanship" από Robert C. Martin.
- "The Mythical Man-Month: Essays on Software Engineering" από Frederick P. Brooks Jr.
- "Introduction to the Theory of Computation" από Michael Sipser.
- "Software Engineering: Principles and Practice" από Hans van Vliet.
- "Agile Software Development, Principles, Patterns, and Practices" από Robert C. Martin.

Αυτές οι πηγές προσφέρουν συστηματική κάλυψη της τεχνολογίας λογισμικού, από τις βασικές αρχές μέχρι τις προηγμένες τεχνικές και μεθόδους. Μπορείτε να τις αξιοποιήσετε για να εμβαθύνετε στην κατανόηση της τεχνολογίας λογισμικού και να επεκτείνετε τις γνώσεις σας στον τομέα αυτόν.

2.2 Τι είναι ποιότητα λογισμικού;

Η ποιότητα του λογισμικού αποτελεί ένα κρίσιμο και σημαντικό θέμα στον τομέα της πληροφορικής και της ανάπτυξης λογισμικού. Η έννοια της ποιότητας του λογισμικού

αποτελεί ένα πολυδιάστατο και πολυπλοκότατο ζήτημα που περιλαμβάνει πολλούς παράγοντες και πτυχές. Θα μπορούσε να οριστεί ως η ικανότητα ενός λογισμικού να ικανοποιεί τις ανάγκες, τις προσδοκίες και τις απαιτήσεις των χρηστών του.

Η ποιότητα του λογισμικού αξιολογείται μέσω διάφορων παραμέτρων και ποιοτικών χαρακτηριστικών. Σύμφωνα με τον καθηγητή Stephen R. Schach, η ποιότητα του λογισμικού μπορεί να μετρηθεί με βάση διάφορα χαρακτηριστικά, όπως η λειτουργικότητα, η απόδοση, η αξιοπιστία, η ασφάλεια, η ευχρηστία και τη συντηρησιμότητα. Η επίτευξη υψηλής ποιότητας συνήθως απαιτεί την εφαρμογή καλών πρακτικών στον σχεδιασμό, την ανάπτυξη και τη δοκιμή του λογισμικού. Επίσης, πρέπει να είναι ασφαλές, επαρκώς επεκτάσιμο και να παρέχει καλή χρηστικότητα.

Η ποιότητα του λογισμικού επηρεάζει την ικανοποίηση των χρηστών, την αξιοπιστία του προϊόντος και την επιτυχία του στην αγορά. Επίσης, η διαρκής βελτίωση της ποιότητας απαιτεί την παρακολούθηση και την αντιμετώπιση τυχόν προβλημάτων και ανεπάρκειών που ανακύπτουν κατά τη διάρκεια του κύκλου ζωής του λογισμικού.

Για περαιτέρω έρευνα σχετικά με την ποιότητα του λογισμικού, προτείνεται η αναφορά στα έργα των καθηγητών Roger S. Pressman και Sommerville Ian, οι οποίοι έχουν αναπτύξει θεωρητικά πλαίσια και μεθοδολογίες για την αξιολόγηση και την εξασφάλιση της ποιότητας του λογισμικού.

Πηγές:

- Schach, S. R. (2010). Object-Oriented and Classical Software Engineering (8th ed.).
- Pressman, R. S. (2014). Software Engineering: A Practitioner's Approach (8th ed.).
- Sommerville, I. (2016). Software Engineering (10th ed.).
- IEEE Standard 730: Software Quality Assurance Plans.
- Witten, I. H., & Bainbridge, D. (2004). Software Engineering: Principles and Practice (2nd ed.).

Αυτές οι πηγές παρέχουν μια ολοκληρωμένη εικόνα για την έννοια της ποιότητας του λογισμικού και μπορούν να αποτελέσουν ένα καλό σημείο εκκίνησης για περαιτέρω μελέτη και έρευνα στον τομέα αυτό.

2.3 Τι είναι τεχνικό χρέος;

Το τεχνικό χρέος (technical debt) αναφέρεται στο κόστος που προκύπτει από τις συνειδητές ή μη συνειδητές συμβιβασμένες αποφάσεις κατά την ανάπτυξη λογισμικού.

Πρόκειται για την απόκλιση από τις βέλτιστες πρακτικές, που μπορεί να οφείλεται σε πιέσεις χρόνου, απουσία γνώσης, περιορισμένου προϋπολογισμού κ.λπ.

Το τεχνικό χρέος μπορεί να εκδηλωθεί με πολλούς τρόπους, όπως ανεπαρκής τεκμηρίωση, ανεπίλυτα προβλήματα ασφάλειας, περιορισμένη επεκτασιμότητα και ανεπιθύμητες εξαρτήσεις μεταξύ των συστατικών του λογισμικού. Όσο το τεχνικό χρέος αυξάνει, τόσο πιο δύσκολο γίνεται το συντήρηση και η εξέλιξη του λογισμικού.

Για να αντιμετωπίσουμε το τεχνικό χρέος, είναι σημαντικό να επενδύσουμε στην τεχνική βελτίωση, να αναθεωρήσουμε τον κώδικα, να επενδύσουμε στην αυτοματοποίηση διαδικασιών και να εφαρμόσουμε καλές πρακτικές ανάπτυξης λογισμικού. Με αυτόν τον τρόπο, μπορούμε να περιορίσουμε την ανοδική πορεία του τεχνικού χρέους και να διατηρήσουμε ένα υγιές και ευέλικτο σύστημα λογισμικού. Ορισμένες πηγές που μπορείτε να ανατρέξετε για περαιτέρω μελέτη του θέματος είναι:

- Martin Fowler, "Technical Debt Quadrant: Balancing Short-term Gain and Long-term Pain" - Αυτό το άρθρο παρουσιάζει την έννοια του τεχνικού χρέους και παρέχει μια ταξινόμηση των διαφόρων τύπων τεχνικού χρέους.
- Steve McConnell, "Technical Debt: 7 Ways to Manage It" - Αυτό το άρθρο παρέχει πρακτικές συμβουλές για τη διαχείριση και μείωση του τεχνικού χρέους στο λογισμικό.
- Ward Cunningham, "The WyCash Portfolio Management System" - Σε αυτήν την πρωτότυπη ανάρτηση στο WikiWikiWeb, ο Ward Cunningham πρωτοστάτησε στην εισαγωγή της έννοιας του τεχνικού χρέους.

Αυτές οι πηγές μπορούν να σας προσφέρουν περισσότερες λεπτομέρειες και ειδικές πρακτικές για τη διαχείριση του τεχνικού χρέους στο λογισμικό.

2.4 Μετρικές τεχνολογίας λογισμικού

Οι μετρικές τεχνολογίας λογισμικού αναφέρονται σε μεθόδους μέτρησης και αξιολόγησης του λογισμικού κατά τη διάρκεια της ανάπτυξης, της συντήρησης και της αξιολόγησής του. Οι μετρικές αυτές προσφέρουν ποσοτικές πληροφορίες για διάφορες πτυχές του λογισμικού και μπορούν να χρησιμοποιηθούν για την αξιολόγηση της ποιότητας, την πρόβλεψη των αποδόσεων, την εκτίμηση της πολυπλοκότητας και τη βελτιστοποίηση των διαδικασιών ανάπτυξης.Με αυτές τις μετρικές, οι προγραμματιστές και οι μηχανικοί λογισμικού μπορούν να αναγνωρίσουν πιθανά προβλήματα και να λάβουν μέτρα για τη

βελτίωση του κώδικα, τη μείωση των σφαλμάτων και την αύξηση της αποδοτικότητας και της αξιοπιστίας του συστήματος.

Αν και αποτελούν ένα χρήσιμο εργαλείο, πρέπει να συνδυάζονται με άλλες προσεγγίσεις και τεχνικές αξιολόγησης, όπως οι πρακτικές ανάπτυξης λογισμικού, οι αυτόματοι έλεγχοι, οι δοκιμές και η συλλογή ανατροφοδοσίας από τους χρήστες.Ο συνδυασμός διαφόρων μετρικών και αξιολογητικών κριτηρίων μπορεί να παρέχει μια πιο ολοκληρωμένη εικόνα της ποιότητας του λογισμικού και να βοηθήσει στην εξαγωγή αξιόπιστων συμπερασμάτων. Επιπλέον, οι μετρικές τεχνολογίας λογισμικού πρέπει να προσαρμόζονται στις ανάγκες και τα χαρακτηριστικά του συγκεκριμένου έργου λογισμικού, καθώς διάφορα έργα μπορεί να απαιτούν διαφορετικές μετρικές για την αξιολόγηση της επίδοσης και της ποιότητας.

Συνοψίζοντας, οι μετρικές τεχνολογίας λογισμικού αποτελούν ένα αξιόλογο εργαλείο για την αξιολόγηση, την παρακολούθηση και τη βελτίωση της ποιότητας, της απόδοσης και της αξιοπιστίας του λογισμικού. Προσφέρουν μετρήσεις και αναλύσεις που μπορούν να βοηθήσουν τους προγραμματιστές και τους οργανισμούς να πάρουν αποφάσεις και να λάβουν μέτρα για τη βελτίωση του λογισμικού. Ωστόσο, πρέπει να χρησιμοποιούνται με προσοχή και να λαμβάνονται υπόψη οι περιορισμοί, τα πλεονεκτήματα και τα περιβάλλοντα παράγοντες που επηρεάζουν το λογισμικό. Επιπλέον, η αξιολόγηση των μετρικών πρέπει να συμπληρώνεται με ανθρώπινη κρίση και εμπειρία για να επιτευχθεί ολοκληρωμένη και ακριβής ανάλυση του λογισμικού.

Υπάρχουν αρκετές μετρικές στην τεχνολογία λογισμικού και είναι στην ευχέρεια των μηχανικών να επιλέξουν συνετά να εξετάσουν κάποιες από αυτές για τα έργα στα οποία εργάζονται αλλά μερικές σύνηθες παρουσιάζονται παρακάτω. Αυτές χρησιμοποιήθηκαν και σε αυτήν την εργασία ώστε για κάθε ανάληση κάποιου έργου με το εργαλείο το οποίο αναπτύχθηκε να υπολογίζονται.

Όνομα Μετρικής	Λίγα λόγια
❖ CBO (Coupling Between Objects)	Ο τρόπος με τον οποίο δύο ή περισσότερα αντικείμενα συσχετίζονται μεταξύ τους σε ένα σύστημα λογισμικού. Η CBO μετράει τον βαθμό σύνδεσης μεταξύ των αντικειμένων, δηλαδή πόσο συχνά αλληλεπιδρούν, αναφέρονται ή εξαρτώνται ο ένας από τον άλλο. Αυξημένος συνδετικός βαθμός μπορεί να υποδηλώνει μεγαλύτερη εξάρτηση μεταξύ των αντικειμένων, μειώνοντας έτσι την ευελιξία, την ανεξαρτησία και την επαναχρησιμοποίηση του κώδικα. Από την άλλη πλευρά, μια χαμηλή τιμή CBO υποδεικνύει χαμηλή συνδετικότητα μεταξύ των αντικειμένων, προωθώντας την ανεξαρτησία και την αποσύνδεση των μερών του συστήματος. Η μετρική CBO βοηθά στην αξιολόγηση της ποιότητας του κώδικα και της αρχιτεκτονικής του συστήματος, και μπορεί να χρησιμοποιηθεί για την πρόβλεψη της συντηρησιμότητας και της ευκολίας ανάπτυξης του λογισμικού.
❖ DAC (Data Access Complexity)	Αναφέρεται στον υπολογισμό του ποσοστού πρόσβασης σε δεδομένα από μια κλάση ή ένα αντικείμενο. Η μετρική αυτή μας δίνει μια εικόνα για το πόσο συχνά ή πολύπλοκα αντικείμενα προσπελαύνουν τα δεδομένα σε μια κλάση. Με άλλα λόγια, μας βοηθά να κατανοήσουμε πόσο εξαρτάται μια κλάση από την πρόσβαση σε δεδομένα και πόσο περίπλοκη είναι η διαδικασία πρόσβασης αυτή. Έχοντας αυτήν την πληροφορία, μπορούμε να αναγνωρίσουμε τυχόν προβλήματα αποδοτικότητας, υψηλής πολυπλοκότητας ή εξάρτησης από συγκεκριμένα δεδομένα και να λάβουμε μέτρα για τη βελτίωση του σχεδιασμού και της απόδοσης του λογισμικού.

- DIT (Depth of Inheritance Tree)

Ο υπολογισμός του αριθμού των επιπέδων κληρονομίας που υπάρχουν μεταξύ μιας κλάσης και της ρίζας του δέντρου κληρονομίας. Δείχνει πόσο βαθιά βρίσκεται μια κλάση στην ιεραρχία κληρονομίας. Η μετρική αυτή μας παρέχει μια εικόνα του επιπέδου περίπλοκης κληρονομικής ιεραρχίας στο λογισμικό μας. Όσο μεγαλύτερο είναι το DIT, τόσο πιο πολύπλοκη είναι η κληρονομική δομή και το πιθανότερο είναι να αυξηθεί η δυσκολία στην κατανόηση και συντήρηση του λογισμικού. Η αναγνώριση υψηλών τιμών DIT μας βοηθά να λάβουμε μέτρα για την απλοποίηση της δομής κληρονομίας και την ελαχιστοποίηση των πιθανών προβλημάτων συντήρησης.

- LCOM (Lack of Cohesion in Methods)

Ο υπολογισμός του βαθμού έλλειψης συνοχής μεταξύ των μεθόδων μιας κλάσης. Δείχνει πόσο σχετικές είναι οι μέθοδοι μιας κλάσης μεταξύ τους. Μια υψηλή τιμή LCOM υποδηλώνει ότι οι μέθοδοι της κλάσης είναι ανεξάρτητες μεταξύ τους και υπάρχει έλλειψη συνοχής. Αυτό μπορεί να οδηγήσει σε δυσκολία στην κατανόηση, συντήρηση και επαναχρησιμοποίηση της κλάσης. Η αναγνώριση υψηλών τιμών LCOM μας βοηθά να εντοπίσουμε πιθανά προβλήματα σχεδίασης και να προβούμε σε αναδιαμόρφωση της κλάσης για να επιτύχουμε καλύτερη συνοχή και ευκολία χρήσης.

- MPC (Message Passing Coupling)

Είναι η αξιολόγηση του βαθμού σύζευξης μεταξύ αντικειμένων που επικοινωνούν μεταξύ τους μέσω μηνυμάτων. Σε απλά λόγια, μας δείχνει πόσο συχνά αντικείμενα ανταλλάσσουν μηνύματα μεταξύ τους. Μια υψηλή τιμή MPC υποδηλώνει ότι υπάρχει έντονη επικοινωνία μεταξύ των αντικειμένων μέσω μηνυμάτων, και αυτό μπορεί να οδηγήσει σε αυξημένη συζήτηση, συντήρηση και εξάρτηση μεταξύ των αντικειμένων. Η αναγνώριση υψηλών τιμών MPC μας βοηθά να αντιληφθούμε πιθανές πολυπλοκότητες στην επικοινωνία των αντικειμένων και να λάβουμε μέτρα για να βελτιστοποιήσουμε την επικοινωνία και τη συνολική δομή του συστήματος.

- NOCC (Number of Children Classes)

Αναφέρεται στον αριθμό των κλάσεων που κληρονομούν από μια γονική κλάση. Απλά, μας δείχνει πόσες κλάσεις είναι απευθείας παιδιά μιας συγκεκριμένης κλάσης. Αν έχουμε μια υψηλή τιμή NOCC, αυτό υποδηλώνει ότι η γονική κλάση έχει πολλά παιδιά, δηλαδή πολλές κλάσεις την κληρονομούν. Αυτό μπορεί να έχει επιπτώσεις στην πολυπλοκότητα του κώδικα και τη συντήρηση, καθώς οι αλλαγές στη γονική κλάση μπορεί να επηρεάσουν πολλές κλάσεις παιδιά. Η ανίχνευση υψηλής τιμής NOCC μας βοηθά να αντιληφθούμε τις σχέσεις κληρονομικότητας στο πρόγραμμα και να λάβουμε αποφάσεις για την αναδιαμόρφωση και τον σχεδιασμό του κώδικα μας.

- NOM (Number of Methods)

Μας δίνει τον αριθμό των λειτουργιών ή των ενεργειών που μπορούν να εκτελεστούν από μια κλάση. Όσο μεγαλύτερη είναι η τιμή της μετρικής NOM, τόσο περισσότερες μέθοδοι έχει η κλάση. Αυτό μπορεί να επηρεάσει την πολυπλοκότητα του κώδικα και την κατανόηση της λειτουργικότητας της κλάσης. Μια υψηλή τιμή NOM μπορεί να υποδείξει ότι η κλάση είναι υπερβολικά μεγάλη και θα μπορούσε να υποστεί διάσπαση σε πιο μικρές κλάσεις για καλύτερη οργάνωση και συντήρηση του κώδικα. Η μετρική NOM μας βοηθά να αξιολογήσουμε την πολυπλοκότητα των κλάσεων και να λάβουμε αποφάσεις για τον σχεδιασμό και τη βελτίωση του λογισμικού μας.

- RFC (Response for a Class)

Η μετρική RFC αναφέρεται στον αριθμό των μεθόδων που μπορούν να κληθούν από μια κλάση, συμπεριλαμβανομένων των μεθόδων που κληρονομούνται από γονικές κλάσεις. Ουσιαστικά, μας δείχνει πόσες διαφορετικές μεθόδους μπορούν να εκτελεστούν από μια κλάση. Όσο μεγαλύτερη είναι η τιμή της μετρικής RFC, τόσο πιο πολλές μεθόδους μπορεί να εκτελέσει η κλάση. Αυτό μπορεί να οδηγήσει σε μεγαλύτερη πολυπλοκότητα και ανεπιθύμητη εξάρτηση μεταξύ των κλάσεων. Η μετρική RFC μας βοηθά να αξιολογήσουμε το μέγεθος και την πολυπλοκότητα των κλάσεων, προσφέροντας μια ιδέα για τον αριθμό των λειτουργιών που μια κλάση υποστηρίζει και πόσο επικοινωνεί με άλλες κλάσεις.

- WMC (Weighted Methods per Class)

Η μετρική WMC (Βαρυστημένες Μέθοδοι ανά Κλάση) στην τεχνολογία λογισμικού αναφέρεται στον αριθμό των μεθόδων που υπάρχουν σε μια κλάση, λαμβάνοντας υπόψη τη σημασία ή το βάρος κάθε μέθοδου. Ουσιαστικά, η μετρική WMC μετράει πόσες μεθόδους υπάρχουν σε μια κλάση και εκτιμά τη συνολική πολυπλοκότητα της κλάσης με βάση αυτόν τον αριθμό. Όσο μεγαλύτερη είναι η τιμή της μετρικής WMC, τόσο πιο πολλές μέθοδοι υπάρχουν και τόσο πιο σύνθετη μπορεί να είναι η κλάση. Αυτό μπορεί να οδηγήσει σε δυσκολίες στη συντήρηση, επαναχρησιμοποίηση και δοκιμή του κώδικα. Η μετρική WMC μας βοηθά να αναγνωρίσουμε κλάσεις με υψηλή πολυπλοκότητα και να λάβουμε αποφάσεις για τυχόν βελτιώσεις στην αρχιτεκτονική και τη σχεδίαση του λογισμικού.

- Size1 or LOC (Lines of Code)

Η μετρική αυτή χρησιμοποιείται για να μετρήσει το μέγεθος ενός προγράμματος και να εκτιμήσει την πολυπλοκότητα του. Όσο μεγαλύτερος είναι ο αριθμός των γραμμών κώδικα, τόσο μεγαλύτερη είναι η πολυπλοκότητα του προγράμματος. Η μετρική LOC μας βοηθά να κατανοήσουμε το μέγεθος του προγράμματος και να εκτιμήσουμε την ποσότητα εργασίας που απαιτείται για τη συντήρηση, την επέκταση ή την ανάπτυξή του. Ωστόσο, η μετρική LOC δεν λαμβάνει υπόψη την ποιότητα του κώδικα ή την αποτελεσματικότητα του προγράμματος, οπότε πρέπει να χρησιμοποιείται με προσοχή και συνδυασμένη με άλλες μετρικές για μια πιο ολοκληρωμένη εκτίμηση.

- Size2 or Number of Properties: (Number of attributes & methods in class)

Αυτή η μετρική μας δίνει μια εικόνα του μεγέθους της κλάσης και του αριθμού των λειτουργιών που εκτελεί. Όσο πιο μεγάλος είναι ο αριθμός των γνωρισμάτων και μεθόδων, τόσο πιο σύνθετη είναι η κλάση και τόσο περισσότερες λειτουργίες εκτελεί. Αυτή η μετρική μπορεί να μας βοηθήσει να αναγνωρίσουμε πολύπλοκες κλάσεις που ενδέχεται να χρειάζονται περαιτέρω ανάλυση ή διασπορά των καθηκόντων τους. Ωστόσο, πρέπει να λαμβάνουμε υπόψη ότι η μετρική Size2 μπορεί να επηρεαστεί από τον τύπο και τη φύση του λογισμικού που αναπτύσσουμε, και πρέπει να συνδυάζεται με άλλες μετρικές για μια πιο ολοκληρωμένη αξιολόγηση.

- Interest in euros

Αποτελεί μια εκτίμηση του ποσού που απαιτείται για τη συντήρηση, την αλλαγή ή τη βελτίωση μιας κλάσης ή ενός συνόλου κλάσεων σε ένα έργο. Μπορεί επίσης να αναφέρεται στο συνολικό οικονομικό κόστος που συνδέεται με την ανάπτυξη, τη συντήρηση και τη διαχείριση του λογισμικού σε ολόκληρο το έργο. Η μετρική "Interest in euros" μας βοηθά να εκτιμήσουμε την οικονομική επίπτωση των αποφάσεων που λαμβάνουμε κατά τη διάρκεια της ανάπτυξης και της συντήρησης του λογισμικού, και μας επιτρέπει να προγραμματίσουμε και να διαχειριστούμε τους πόρους μας με πιο αποτελεσματικό τρόπο.

- Interest in hours

Ο χρόνος ή η προσπάθεια που απαιτείται για τη συντήρηση ή την τροποποίηση ενός συγκεκριμένου λογισμικού στοιχείου ή ενός έργου. Αποτελεί μια μέτρηση του επιπέδου συμμετοχής ή της ανάθεσης πόρων σε ώρες που απαιτούνται για την αντιμετώπιση εργασιών συντήρησης, επιδιορθώσεων σφαλμάτων, βελτιώσεων ή άλλων αλλαγών. Αυτή η μετρική μπορεί να χρησιμοποιηθεί για να εκτιμηθεί ο φόρτος εργασίας και οι ανάγκες πόρων για τις δραστηριότητες συντήρησης λογισμικού. Βοηθά τις οργανώσεις να κατανοήσουν την απαιτούμενη χρονική δέσμευση για τη συντήρηση ή την ενημέρωση των λογισμικών στοιχείων και συμβάλλει στον προγραμματισμό των έργων και την ανάθεση πόρων.

- Complexity

Η μετρική της πολυπλοκότητας στην τεχνολογία λογισμικού αναφέρεται στην αξιολόγηση του επιπέδου δυσκολίας και περιπλοκότητας ενός λογισμικού. Αποτελεί μια μέτρηση του αριθμού των στοιχείων, των συνδέσεων και των εξαρτήσεων που υπάρχουν σε ένα σύστημα λογισμικού. Περιλαμβάνει την ανάλυση των πολλαπλών μονάδων και της σχέσης μεταξύ τους, προκειμένου να κατανοήσουμε τον βαθμό περιπλοκότητας που υπάρχει στο λογισμικό. Η μετρική της πολυπλοκότητας μας βοηθά να αξιολογήσουμε τον βαθμό που ένα σύστημα είναι δυσνόητο ή δύσκολο στην κατανόηση, τη συντήρηση και την ανάπτυξη. Κατανοώντας την πολυπλοκότητα, μπορούμε να λάβουμε αποφάσεις για τον σχεδιασμό και τη βελτιστοποίηση του λογισμικού, προκειμένου να επιτύχουμε καλύτερη απόδοση και αξιοπιστία.

❖ Cumulative Interest	Η μετρική του συσσωρευτικού ενδιαφέροντος στην τεχνολογία λογισμικού αναφέρεται στον υπολογισμό του συνολικού ενδιαφέροντος που έχει επικεντρωθεί σε ένα συγκεκριμένο στοιχείο ή λειτουργία του λογισμικού. Αυτό το ενδιαφέρον μπορεί να προκύψει από τους χρήστες, τους πελάτες ή την ομάδα ανάπτυξης. Η μετρική αυτή μπορεί να χρησιμοποιηθεί για να κατανοήσουμε τη σημασία και την επίδραση ενός συγκεκριμένου στοιχείου στο σύνολο του λογισμικού. Με τον υπολογισμό του συσσωρευτικού ενδιαφέροντος, μπορούμε να εστιάσουμε τις προσπάθειές μας σε αυτά τα κρίσιμα σημεία και να λάβουμε αποφάσεις βασιζόμενες στην προτεραιότητα και την αξία που αποδίδουν οι ενδιαφερόμενοι φορείς σε αυτά.

Πίνακας 1: Επεξήγηση Μετρικών όπου υπολογίζονται στην εφαρμογή

2.5 Web Services

Τα web services αναφέρονται σε μια τεχνολογία που επιτρέπει την ανταλλαγή δεδομένων και την επικοινωνία μεταξύ διαφορετικών εφαρμογών ή συστημάτων μέσω του Διαδικτύου. Τα web services είναι σχεδιασμένα για να επιτρέπουν την αλληλεπίδραση και την ανταλλαγή δεδομένων μεταξύ διαφορετικών πλατφορμών και προγραμματιστικών γλωσσών.

Τα web services χρησιμοποιούν συνήθως το πρωτόκολλο HTTP για την ανταλλαγή δεδομένων, και συνήθως βασίζονται σε ανοιχτές προδιαγραφές και πρωτόκολλα όπως το XML (Extensible Markup Language) ή το JSON (JavaScript Object Notation) για τη μορφοποίηση και τον προσδιορισμό των δεδομένων που ανταλλάσσονται.[1]

Ένα από τα κύρια πλεονεκτήματα των web services είναι η δυνατότητα επικοινωνίας μεταξύ διαφορετικών συστημάτων και πλατφορμών, ανεξαρτήτως του τρόπου ανάπτυξής τους. Αυτό επιτρέπει στις εφαρμογές και τα συστήματα να ανταλλάσσουν δεδομένα και να συνεργάζονται μεταξύ τους, προσφέροντας ευελιξία και επεκτασιμότητα.

Εικόνα 1: Web Service

Τα web services μπορούν να είναι χωρισμένα σε δύο κύριες κατηγορίες: τα SOAP (Simple Object Access Protocol) και τα REST (Representational State Transfer). Τα SOAP web services χρησιμοποιούν το πρωτόκολλο SOAP για την ανταλλαγή δεδομένων, ενώ τα RESTful web services βασίζονται στις αρχές του REST και χρησιμοποιούν το πρωτόκολλο HTTP για την ανταλλαγή δεδομένων.

Τα web services έχουν ευρεία εφαρμογή σε διάφορους τομείς, όπως η διαμοιρασμένη αρχιτεκτονική, ο ενσωματωμένος ιστός (mashups), οι εφαρμογές κινητών συσκευών και οι σύγχρονες εφαρμογές στον νέφος (cloud applications). Οι web services παρέχουν ένα αποδοτικό και ευέλικτο μηχανισμό για την ανταλλαγή δεδομένων και την επικοινωνία μεταξύ

[1] Πηγή εικόνας

εφαρμογών και συστημάτων, συμβάλλοντας στην αύξηση της αποδοτικότητας και της επαναχρησιμοποίησης του λογισμικού.

2.5.1 APIs

Ένα API (Application Programming Interface) είναι ένα σύνολο κανόνων και διεπαφών που επιτρέπει σε δύο ή περισσότερα λογισμικά να επικοινωνούν μεταξύ τους. Απλούστερα, μπορεί να θεωρηθεί ως ένας διανομέας πληροφοριών που επιτρέπει στις εφαρμογές να ανταλλάσσουν δεδομένα μεταξύ τους.

Τα API ορίζουν τις μεθόδους, τα δεδομένα και τις διαδικασίες που μπορούν να χρησιμοποιηθούν για την αλληλεπίδραση με ένα συγκεκριμένο λογισμικό ή υπηρεσία. Οι εφαρμογές μπορούν να καλέσουν αυτές τις μεθόδους και να ανταλλάξουν δεδομένα χρησιμοποιώντας το API, παρέχοντας ή λαμβάνοντας τις αναγκαίες πληροφορίες.

Τα API μπορούν να χρησιμοποιηθούν για πολλούς σκοπούς, όπως η ανάπτυξη εφαρμογών, η ενσωμάτωση λειτουργικοτήτων από διάφορες πηγές και υπηρεσίες, η δημιουργία συστημάτων διαλειτουργικότητας και πολλά άλλα. Οι δημοφιλείς τύποι API περιλαμβάνουν τα web APIs που χρησιμοποιούνται για την ανταλλαγή δεδομένων μέσω του διαδικτύου, όπως τα RESTful APIs και τα SOAP APIs.

Συνοψίζοντας, τα API είναι εργαλεία που επιτρέπουν την επικοινωνία και την ανταλλαγή δεδομένων μεταξύ εφαρμογών ή υπηρεσιών, επιτρέποντας την ολοκλήρωση, την επεκτασιμότητα και την αποτελεσματικότητα των λογισμικών συστημάτων.

2.5.2 SOAP vs REST

Τόσο το SOAP όσο και το REST είναι μηχανισμοί ανταλλαγής δεδομένων στο διαδίκτυο. Οι διαφορές μεταξύ τους είναι οι εξής:

1. Μορφή Δεδομένων:
    - Το SOAP χρησιμοποιεί μορφή δεδομένων XML, ενώ το REST επιτρέπει την ανταλλαγή δεδομένων σε πολλές μορφές.
2. Σχεδιασμός Κανόνων:
    - Το SOAP είναι υψηλά δομημένο και χρησιμοποιείται για τον καθορισμό κανόνων επικοινωνίας. Αντίθετα, το REST είναι πιο ευέλικτο και επιτρέπει στις εφαρμογές να ανταλλάσσουν δεδομένα με πολλούς τρόπους.

Ως εκ τούτου, τόσο το SOAP όσο και το REST μπορούν να χρησιμοποιηθούν για τη δημιουργία διεπαφών προγραμματισμού εφαρμογών (API) που επιτρέπουν την ανταλλαγή δεδομένων μεταξύ διαφορετικών εφαρμογών. Και τα δύο προσφέρουν τις εξής δυνατότητες:

- Περιγράφουν κανόνες και πρότυπα για τον τρόπο με τον οποίο οι εφαρμογές πραγματοποιούν, επεξεργάζονται και αποκρίνονται σε αιτήματα δεδομένων από άλλες εφαρμογές.
- Χρησιμοποιούν το πρωτόκολλο HTTP για την ανταλλαγή πληροφοριών.
- Υποστηρίζουν SSL/TLS για ασφαλή και κρυπτογραφημένη επικοινωνία.
- Μπορούν να χρησιμοποιηθούν για τη δημιουργία ασφαλών, επεκτάσιμων και ανθεκτικών κατανεμημένων συστημάτων.

2.6 SDK4ED, Τι είναι και ποια η χρησιμότητα[2]

Εικόνα 2: SDK4ED

Η πλατφόρμα SDK4ED αποτελεί ένα εργαλείο για τη διαχείριση του τεχνικού χρέους στον κώδικα. Το τεχνικό χρέος αναφέρεται στον κώδικα που έχει αναπτυχθεί με τρόπο που δυσκολεύει τη συντήρηση και προκαλεί προβλήματα στο μέλλον. Η πλατφόρμα SDK4ED προσφέρει πληροφορίες και λειτουργίες που βοηθούν τους προγραμματιστές να αντιμετωπίσουν το τεχνικό χρέος και να βελτιστοποιήσουν το λογισμικό.

Η πλατφόρμα αυτή έχει αναπτυχθεί μετά από ένα 3ετές έργο σε συνεργασία με πολλές εταιρείες λογισμικού. Τα εργαλεία και οι μέθοδοι που παρέχει η πλατφόρμα έχουν επικεντρωθεί στη μέτρηση, ανάλυση και πρόληψη του τεχνικού χρέους, καθώς και στην αξιολόγηση της οικονομικής επίπτωσης της χρήσης της. Μέσω αυτής της πλατφόρμας, οι προγραμματιστές μπορούν να λάβουν αποφάσεις που θα βελτιώσουν την ποιότητα του κώδικα, προλαμβάνοντας προβλήματα και εξοικονομώντας χρόνο και πόρους στη συντήρηση του λογισμικού.

Η αποτελεσματική διαχείριση του τεχνικού χρέους αποτελεί ένα ζωτικής σημασίας κομμάτι της διαδικασίας ανάπτυξης λογισμικού. Η πλατφόρμα SDK4ED προσφέρει ένα ολοκληρωμένο περιβάλλον που επιτρέπει στους προγραμματιστές να αντιμετωπίσουν τις προκλήσεις που σχετίζονται με το τεχνικό χρέος. Μέσω της πλατφόρμας, οι προγραμματιστές

[2] Πηγή εικόνας

μπορούν να πραγματοποιήσουν μετρήσεις του τεχνικού χρέους, να αναλύσουν την εξέλιξή του, να προλάβουν την εμφάνιση νέου χρέους και να αναλάβουν δράση για την εξάλειψή του.

3 Μεθοδολογία

Παρακάτω παρουσιάζονται όλα τα στάδια, τα εργαλεία και οι τεχνολογίες που χρησιμοποιήθηκαν για την επιτυχή και μεθοδική εκπόνηση της εργασίας, λαμβάνοντας υπόψη τις απαιτήσεις και τις κατευθυντήριες γραμμές των καθηγητών.

Το πρώτο στάδιο περιλάμβανε την καθορισμένη επιλογή των τεχνολογιών που θα χρησιμοποιηθούν για την υλοποίηση του προγραμματιστικού κομματιού της εργασίας. Επιλέχθηκε το Spring Boot framework της Java για την υλοποίηση του server, ο οποίος αναλύει κάθε project, πραγματοποιεί υπολογισμούς μετρικών και αποθηκεύει τα δεδομένα σε μια PostgreSQL βάση δεδομένων. Επίσης, χρησιμοποιήθηκε το Docker για το πακετάρισμα και την ανεξαρτησία της εφαρμογής από λειτουργικά συστήματα και ρυθμίσεις. Όσον αφορά το User Interface (UI), χρησιμοποιήθηκε η πλατφόρμα SDK4ED η οποία είναι υλοποιημένη με το ReactJS framework, ενώ επίσης χρησιμοποιήθηκε το Docker για το πακετάρισμα, τη διανομή και την εκτέλεση της εφαρμογής.

Στην αρχική φάση, παρουσιάστηκε η δυσκολία της κατανόησης του απευθείας τρόπου έναρξης της υλοποίησης. Για τον λόγο αυτό, απαιτήθηκε σημαντική έρευνα σε συνεργασία με τις προτάσεις και το διαδικτυακό υλικό που παρείχαν οι καθηγητές. Διαθέσιμος χρόνος επενδύθηκε για την απόκτηση επαρκούς κατανόησης των προαναφερθέντων τεχνολογιών μέσω ποικίλων πηγών.

Στην φάση της υλοποίησης, τα εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη του εργαλείου, επιλέχθηκαν ήδη γνωστά εργαλεία, εξαιτίας της ήδη υπάρχουσας εμπειρίας μου.
Τα στάδια όπου πραγματοποιήθηκαν σε σειρά:

1. Έρευνα και κατανόηση του Spring Boot Framework της Java.
2. Έρευνα για την βάση δεδομένων PostgreSQL και σύνδεση της με Spring Boot Framework.
3. Αρχική υλοποίηση και μεταφορά της υπάρχουσας εφαρμογής metrics calculator και τροποποίηση της για να υποστηριχθεί η υλοποίηση της ίδιας λογικής αλλά σε service.
4. Δημιουργία API calls και τελική παραμετροποίηση του metrics calculator.
5. Έρευνα για το Docker και το πακετάρισμα της εφαρμογής.
6. Έρευνα για το React Framework της Javascript.
7. Επεξεργασία της πλατφόρμας SDK4ED και τροποποίηση των απαραίτητων κομματιών για την υποστήριξη και την σύνδεση με τον Server.

8. Δημιουργία πινάκων και σχεδιαγραμμάτων σε ReactJS έτσι ώστε να γίνει οπτικοποίηση των αποτελεσμάτων της ανάλησης από τον server στην πλατφόρμα SDK4ED.

Παρακάτω παρουσιάζονται αναλυτικά οι τεχνολογίες, τα εργαλεία και οι υπηρεσίες που χρησιμοποιήθηκαν, με συνοπτικές περιγραφές για να κατανοηθεί η χρησιμότητά τους και ο λόγος επιλογής τους.

3.1 Ανάλυση τεχνολογιών και εργαλείων

3.1.1 Spring Boot Framework[3]

Το Spring Boot είναι ένα ελαφρύ, ανοιχτού κώδικα framework που ανήκει στην οικογένεια των πλατφορμών ανάπτυξης εφαρμογών Java του Spring Framework. Το Spring Boot έχει σχεδιαστεί για να διευκολύνει και να επιταχύνει τη δημιουργία εφαρμογών Java, προσφέροντας μια ευέλικτη και παραγωγική περιβάλλον ανάπτυξης.

Εικόνα 3: Spring Boot Framework

Η ανάγκη για τη δημιουργία του Spring Boot προέκυψε από την ανάγκη των προγραμματιστών να απλοποιήσουν την ανάπτυξη εφαρμογών Java και να μειώσουν τον χρόνο και την πολυπλοκότητα που συνήθως συνοδεύουν την αρχική ρύθμιση και την παραμετροποίηση μιας εφαρμογής. Το Spring Boot παρέχει έναν αυτοματοποιημένο τρόπο για τη ρύθμιση και την επέκταση των εφαρμογών Java, επιτρέποντας στους προγραμματιστές να επικεντρωθούν στην κατασκευή της λογικής της εφαρμογής αντί να ασχοληθούν με λεπτομέρειες συνδεσμολογίας και ρύθμισης.

Το Spring Boot χρησιμεύει ως ένα ευέλικτο και παραγωγικό εργαλείο ανάπτυξης για τη δημιουργία μικρών ή μεγάλης κλίμακας εφαρμογών Java. Με τη χρήση του Spring Boot, οι προγραμματιστές μπορούν να αξιοποιήσουν τις προηγμένες δυνατότητες του Spring Framework και να αναπτύξουν γρήγορα και αποδοτικά εφαρμογές που είναι εύκολο να συντηρηθούν και να διαχειριστούν. Επιπλέον, το Spring Boot προσφέρει ενσωματωμένη υποστήριξη για τη δημιουργία αυτόνομων εφαρμογών με ενσωματωμένους διακομιστές,

[3] Πηγή εικόνας

επιτρέποντας την ευκολότερη ανάπτυξη και αναπαραγωγή των εφαρμογών Java σε ποικίλα περιβάλλοντα εκτέλεσης.

Για να εγκαταστήσετε το Spring Boot Framework, ακολουθήστε τα παρακάτω βήματα:

1. Εγκαταστήστε το Java Development Kit (JDK) στον υπολογιστή σας, αν δεν το έχετε ήδη. Το Spring Boot απαιτεί το JDK για την ανάπτυξη και την εκτέλεση εφαρμογών Java.
2. Κατεβάστε το Spring Boot από την επίσημη ιστοσελίδα του στο https://spring.io/projects/spring-boot. Επιλέξτε την τελευταία έκδοση που επιθυμείτε να χρησιμοποιήσετε.
3. Αποσυμπιέστε το αρχείο λήψης του Spring Boot στον τοπικό σας δίσκο.
4. Εισάγετε ή ανοίξτε τον φάκελο ο οποίος αποσυμπιέστηκε με κάποιο code editor ή κάποιο IDE της επιλογής σας.

3.1.2 Maven

Το Apache Maven είναι ένα εργαλείο διαχείρισης έργων για την ανάπτυξη λογισμικού στην πλατφόρμα Java. Χρησιμοποιείται για τη δημιουργία, τη διαχείριση και την επέκταση έργων Java, καθώς και για τη διαχείριση εξαρτήσεων και την αυτόματη διαχείριση της διαδικασίας χτισίματος και επίδοσης του λογισμικού.

Το Maven χρησιμοποιεί ένα αρχείο διαμόρφωσης με το όνομα pom.xml (Project Object Model) για την περιγραφή της δομής και των εξαρτήσεων του έργου. Μέσω αυτού του αρχείου, μπορείς να καθορίσεις τις εξαρτήσεις του έργου, τις διαδικασίες χτισίματος, τις επιπλέον διαμορφώσεις και πολλά άλλα.[4]

Εικόνα 4:Maven

Η εγκατάσταση του Maven είναι σχετικά απλή. Ακολούθησε τα παρακάτω βήματα για να εγκαταστήσετε το Maven στον υπολογιστή σου:

1. Κατεβάστε την τελευταία έκδοση του Maven από την επίσημη ιστοσελίδα: https://maven.apache.org/download.cgi
2. Αποσυμπιέστε το αρχείο λήψης σε έναν κατάλογο της επιλογής σου.
3. Ορίστε τη μεταβλητή περιβάλλοντος `M2_HOME` στον κατάλογο που αποσυμπίεσατε το Maven.

[4] Πηγή εικόνας

4. Πρόσθεσε τον κατάλογο bin του Maven στη μεταβλητή περιβάλλοντος `PATH`.
5. Ελέγξτε την εγκατάσταση του Maven εκτελώντας την εντολή `mvn -v` στο τερματικό.

3.1.3 PostgreSQL

Η PostgreSQL είναι μια ισχυρή σχεσιακή βάση δεδομένων που αναπτύχθηκε από την ακαδημαϊκή κοινότητα. Είναι μια ανοιχτού κώδικα λύση που παρέχει αξιοπιστία, αντοχή, επεκτασιμότητα και προηγμένα χαρακτηριστικά. Η PostgreSQL λειτουργεί σε πολλές πλατφόρμες, συμπεριλαμβανομένων των Linux, Windows και macOS, και χρησιμοποιείται ευρέως από επιχειρήσεις, οργανισμούς και προγραμματιστές σε όλο τον κόσμο.[5]

Ένα από τα βασικά πλεονεκτήματα της PostgreSQL είναι η ανθεκτικότητά της σε βαριές φορτία και υψηλής διαθεσιμότητας περιβάλλοντα. Μπορεί να υποστηρίξει μεγάλους όγκους δεδομένων και να εκτελεί πολύπλοκες ερωτήσεις με γρήγορο χρόνο απόκρισης. Επιπλέον, η PostgreSQL παρέχει προηγμένες λειτουργίες, όπως υποστήριξη γεωγραφικών δεδομένων, αναγνώριση πλήρους κειμένου, αναδρομικές ερωτήσεις και πολλά άλλα.

Εικόνα 5: PostgreSQL

Επιπλέον, η PostgreSQL υποστηρίζει προηγμένες δυνατότητες ασφάλειας, συμπεριλαμβανομένης της κρυπτογράφησης δεδομένων και των αρχείων καταγραφής. Επιτρέπει επίσης τον παραλληλισμό και τη διαχείριση πολλαπλών συνδέσεων, επιτρέποντας την αποτελεσματική λειτουργία σε πολυνηματικά περιβάλλοντα.

Για να συνδέσετε τη βάση δεδομένων PostgreSQL με το Spring Boot στην Java, μπορείτε να ακολουθήσετε τα παρακάτω βήματα:

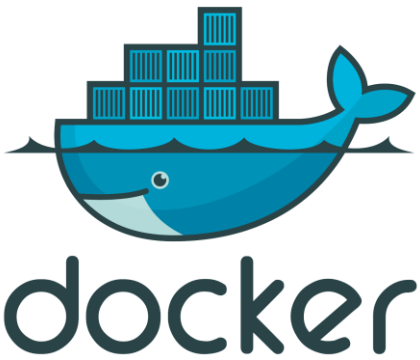
1. Εισαγάγετε την απαιτούμενη εξάρτηση της PostgreSQL στο αρχείο pom.xml του έργου σας. Αυτή η εξάρτηση θα βοηθήσει το Spring Boot να αλληλεπιδράσει με τη βάση δεδομένων PostgreSQL.

[5] Πηγή εικόνας

2. Ορίστε τις απαραίτητες παραμέτρους σύνδεσης στο αρχείο application.properties ή application.yml του έργου σας. Αυτές οι παράμετροι περιλαμβάνουν το URL της βάσης δεδομένων, το όνομα χρήστη, τον κωδικό πρόσβασης και άλλες ρυθμίσεις σύνδεσης.
3. Δημιουργήστε μια κλάση Repository που εκτελεί τις λειτουργίες πρόσβασης στη βάση δεδομένων. Η κλάση αυτή θα είναι υπεύθυνη για την εκτέλεση ερωτημάτων SQL, την ανάκτηση και την ενημέρωση των δεδομένων.
4. Στην κλάση υπηρεσίας ή ελεγκτή Spring, χρησιμοποιήστε την κλάση Repository για να αλληλεπιδράσετε με τη βάση δεδομένων και να εκτελέσετε λειτουργίες πάνω σε αυτήν.

3.1.4 Docker[6]

Το Docker είναι μια ανοικτού κώδικα πλατφόρμα που χρησιμοποιείται για τη δημιουργία, την ανάπτυξη και την εκτέλεση εφαρμογών με χρήση της τεχνολογίας των ελαφριών εικονικών μηχανών, γνωστών ως "containers". Οι εικόνες Docker περιλαμβάνουν όλα τα απαραίτητα εργαλεία και τις εξαρτήσεις που απαιτούνται για την εκτέλεση μιας εφαρμογής, καθιστώντας την εγκατάσταση και την ανάπτυξη εφαρμογών πιο αποτελεσματική και συμβατή μεταξύ διαφορετικών περιβαλλόντων.



Εικόνα 6: Docker

Οι εφαρμογές που τρέχουν σε περιβάλλοντα Docker είναι ελαφριές, απομονωμένες και φορητές, καθώς το Docker προσφέρει ένα ενοποιημένο περιβάλλον εκτέλεσης για τις εφαρμογές. Αυτό σημαίνει ότι μπορούν να εκτελεστούν σε οποιονδήποτε υπολογιστή έχει εγκατεστημένο το Docker, ανεξάρτητα από το λειτουργικό σύστημα ή τις εξαρτήσεις της εφαρμογής.

Οι κύριες έννοιες που σχετίζονται με το Docker είναι οι εξής:

1. Εικόνες (Images): Οι εικόνες Docker είναι πρότυπα που περιέχουν όλα τα απαραίτητα αρχεία και ρυθμίσεις για την εκτέλεση μιας εφαρμογής. Μπορούν να δημιουργηθούν

[6] Πηγή εικόνας

από μια βάση εικόνας ή να ληφθούν από το Docker Hub, το κεντρικό αποθετήριο εικόνων Docker.

2. Ελάχιστη Διανομή (Minimal Runtime Environment): Το Docker παρέχει ένα ελαφρύ και απομονωμένο περιβάλλον εκτέλεσης για τις εφαρμογές, γνωστό ως ελάχιστη διανομή. Αυτή η διανομή περιέχει μόνο τις απαραίτητες εξαρτήσεις για την εκτέλεση της εφαρμογής, επιτρέποντας μια απομονωμένη και ασφαλή εκτέλεση.
3. Containers: Οι ελαφριές εικονικές μηχανές, γνωστές ως containers, είναι ο μηχανισμός με τον οποίο τρέχουν οι εφαρμογές στο περιβάλλον του Docker. Κάθε container είναι απομονωμένο και περιλαμβάνει την εικόνα της εφαρμογής και τα απαραίτητα περιβαλλοντικά χαρακτηριστικά για την εκτέλεσή της.

Η εγκατάσταση του Docker είναι σχετικά απλή και εξαρτάται από το λειτουργικό σύστημα που χρησιμοποιείτε. Υπάρχουν εκδόσεις του Docker για τα περισσότερα λειτουργικά συστήματα, συμπεριλαμβανομένων των Windows, macOS και Linux. Μπορείτε να επισκεφθείτε την επίσημη ιστοσελίδα του Docker (https://www.docker.com/) για να βρείτε τις οδηγίες εγκατάστασης που αφορούν το συγκεκριμένο λειτουργικό σας σύστημα.

3.1.5 React Framework

Το React είναι ένα ανοικτού κώδικα JavaScript πρόγραμμα πλαίσιο που χρησιμοποιείται για την ανάπτυξη διεπαφών χρήστη (UI). Είναι σχεδιασμένο για να επιτρέπει την ανάπτυξη ευέλικτων και ευχρηστών διεπαφών χρήστη, όπου οι αλλαγές στην κατάσταση του UI αντανακλώνται αυτόματα στο περιεχόμενο της σελίδας χωρίς την ανάγκη για ανανέωση ολόκληρης της σελίδας.

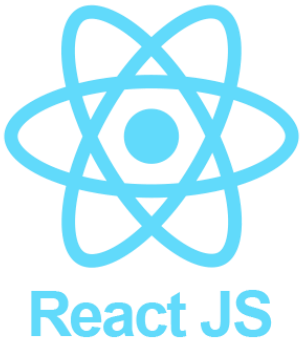
Τα κύρια χαρακτηριστικά του React περιλαμβάνουν:

1. Συνθετικά (Components): Το React χρησιμοποιεί την έννοια των συνθετικών, που είναι ανεξάρτητα κομμάτια κώδικα με δική τους λογική και εμφάνιση. Αυτό επιτρέπει την επαναχρησιμοποίηση και την οργάνωση του κώδικα σε μικρές και αυτόνομες μονάδες.
2. Εικονικό DOM (Virtual DOM): Το React χρησιμοποιεί ένα εικονικό μοντέλο αντικειμένου (Virtual DOM) για την αποδοτική ανανέωση του UI. Αντί να ενημερώνει και ανανεώνει ολόκληρο το DOM κάθε φορά που αλλάζει η κατάσταση, το React συγκρίνει το εικονικό DOM με το πραγματικό DOM και εφαρμόζει μόνο τις αλλαγές που απαιτούνται.
3. Μονοδιάστατη ροή δεδομένων: Το React υποστηρίζει μια μονοδιάστατη ροή δεδομένων, όπου τα δεδομένα ρέουν μόνο από την κορυφή προς τα κάτω. Αυτό

διευκολύνει την αντιμετώπιση και την προβολή της κατάστασης του UI, καθιστώντας τον κώδικα πιο προβλέψιμο και ευκολοδιάβαστο.

4. Αποδοτικός χειρισμός αλλαγών: Το React χρησιμοποιεί τον μηχανισμό "αποδοτικού χειρισμού αλλαγών" για να ελαχιστοποιήσει τις ανανεώσεις του UI και να βελτιστοποιήσει την απόδοση. Με αυτόν τον τρόπο, μόνο οι αλλαγές που αφορούν την κατάσταση ενός συνθετικού αποδίδονται, ενώ οι υπόλοιπες ανανεώσεις αποφεύγονται.[7]

Ο κύριος στόχος του React είναι να δημιουργήσει αποδοτικές και ευέλικτες διεπαφές χρήστη, με έμφαση στην αποδοτικότητα και την ευκολία συντήρησης του κώδικα. Ένα από τα βασικά χαρακτηριστικά του React είναι η επαναχρησιμότητα των συνθετικών (components), που επιτρέπει την οργάνωση του κώδικα σε μικρά και αυτόνομα κομμάτια, επιτρέποντας έτσι την ευκολία στην ανάπτυξη και συντήρηση.



Εικόνα 7: ReactJS Framework

Για να εγκαταστήσετε το React, μπορείς να ακολουθήσεις τα παρακάτω βήματα:

1. Βεβαιώσου ότι έχεις εγκατεστημένο το Node.js στον υπολογιστή σου. Το React απαιτεί το Node.js για τη διαχείριση των απαιτούμενων εργαλείων και εξαρτήσεων.
2. Ανοίξτε ένα τερματικό (command prompt) και πλοηγήσου στον φάκελο όπου θέλεις να δημιουργήσεις το νέο σου React project.
3. Εκτέλεσε την εντολή `npx create-react-app my-app` για να δημιουργήσεις ένα νέο React project. Η εντολή αυτή θα εγκαταστήσει αυτόματα το React και τα απαιτούμενα πακέτα για το έργο σου.
4. Μόλις ολοκληρωθεί η εγκατάσταση, μεταβείς στον φάκελο του νέου σου project με την εντολή `cd my-app`.
5. Τέλος, εκτέλεσε την εντολή `npm start` για να ξεκινήσεις τον τοπικό αναπτυξιακό διακομιστή και να δεις το React project στον περιηγητή σου.

[7] Πηγή εικόνας

3.1.6 MDB Bootstrap for ReactJS[8]

Το MDB Bootstrap είναι ένα ισχυρό framework για την ανάπτυξη του χρήστη που βασίζεται στο ReactJS. Βασίζεται στο Bootstrap, ένα δημοφιλές framework προσαρμογής της διεπαφής χρήστη, και προσθέτει επιπλέον στοιχεία και λειτουργίες που απλοποιούν την ανάπτυξη ευέλικτων και σύγχρονων διαδικτυακών εφαρμογών.

Τα κύρια χαρακτηριστικά του MDB Bootstrap περιλαμβάνουν προδιαμορφωμένα στοιχεία διεπαφής, ενσωματωμένες λειτουργίες απόκρισης, έτοιμα παραδείγματα και πρότυπα, ενσωματωμένη διαχείριση κατάστασης (state management), ένα ισχυρό σύστημα πλοήγησης και πολλά άλλα. Στοιχεία όπως κάρτες, μονάδες, φόρμες, μπάρες πλοήγησης και πολλά άλλα παρέχονται προδιαμορφωμένα, επιτρέποντάς σου να αναπτύξεις γρήγορα και εύκολα επαγγελματικής ποιότητας διεπαφές χρήστη.

Εικόνα 8: MDB for ReactJS

Για την εγκατάσταση του MDB Bootstrap στο έργο σου, ακολούθησε τα παρακάτω βήματα:

1. Εγκατάσταση του Create React App (CRA), αν δεν το έχεις ήδη εγκατεστημένο, με την εντολή `npx create-react-app my-app`.
2. Εγκατάσταση του MDB React με την εντολή `npm install mdbreact`.
3. Έπειτα μπορείς να εισάγεις ένα στοιχείο κάρτας χρησιμοποιώντας για παράδειγμα την εντολή `import { Card } from 'mdbreact';`.

3.1.7 HighCharts for ReactJS[9]

Το Highcharts for ReactJS είναι μια εξαιρετική επιλογή για τη δημιουργία διαγραμμάτων και γραφημάτων στις εφαρμογές ReactJS. Η βιβλιοθήκη προσφέρει μια εύχρηστη διεπαφή για την απεικόνιση δεδομένων με ποικίλους τύπους γραφημάτων, όπως γραμμές, μπάρες, κυκλικά διαγράμματα, περιοχές και άλλα.

Οι βασικές λειτουργίες που παρέχει το Highcharts for ReactJS περιλαμβάνουν:

Εικόνα 9: Highcharts for ReactJS

[8] Πηγή εικόνας
[9] Πηγή εικόνας

1. Δημιουργία διαγραμμάτων: Μπορείτε να δημιουργήσετε διάφορα είδη διαγραμμάτων με ελάχιστον κώδικα. Η βιβλιοθήκη παρέχει πολλές επιλογές στυλ και παραμέτρων για να προσαρμόσετε τα διαγράμματα σύμφωνα με τις απαιτήσεις σας.
2. Δυναμική ανανέωση δεδομένων: Μπορείτε να ενημερώνετε τα δεδομένα του διαγράμματος αυτόματα και δυναμικά χωρίς να απαιτείται ανανέωση της σελίδας.
3. Αλληλεπίδραση και ενεργοποίηση γεγονότων: Μπορείτε να προσθέσετε αλληλεπίδραση στα διαγράμματά σας, όπως κλικ και αιωρούμενα γεγονότα, για να παρέχετε περαιτέρω πληροφορίες ή ενέργειες στους χρήστες.

Για να εγκαταστήσετε το Highcharts for ReactJS, μπορείτε να ακολουθήσετε τα παρακάτω βήματα:

1. Ανοίξτε το τερματικό σας και πλοηγηθείτε στον κατάλογο του έργου σας.
2. Εκτελέστε την εντολή `npm install highcharts-react-official` για να εγκαταστήσετε το πακέτο Highcharts for ReactJS.
3. Μόλις ολοκληρωθεί η εγκατάσταση, μπορείτε να χρησιμοποιήσετε το Highcharts for ReactJS στην εφαρμογή σας.

3.1.8 Git

Το Git είναι ένα διανεμημένο σύστημα διαχείρισης εκδόσεων που χρησιμοποιείται για την παρακολούθηση, την αποθήκευση και τον έλεγχο αλλαγών σε κώδικα και άλλα αρχεία. Επιτρέπει στους προγραμματιστές να συνεργάζονται αποτελεσματικά, να διαχειρίζονται το ιστορικό των αλλαγών και να επιστρέφουν σε προηγούμενες εκδόσεις του κώδικα ανά πάσα στιγμή.[10]

Εικόνα 10: Git

Με το Git, μπορείτε να δημιουργήσετε ένα τοπικό αποθετήριο για τον κώδικά σας, όπου μπορείτε να καταγράφετε τις αλλαγές σας, να δημιουργείτε κλαδιά για να αναπτύξετε διαφορετικά χαρακτηριστικά ή να δοκιμάσετε νέες ιδέες, και να συγχωνεύετε τις αλλαγές σας με άλλους προγραμματιστές. Επιπλέον, μπορείτε να συνεργαστείτε με άλλους προγραμματιστές, να μοιραστείτε τον κώδικά σας και να συνεργαστείτε για την επίλυση προβλημάτων ή την ενσωμάτωση νέων λειτουργιών.

[10] Πηγή εικόνας

Η εγκατάσταση του Git είναι απλή και εξαρτάται από το λειτουργικό σύστημα που χρησιμοποιείτε. Για παράδειγμα, σε ένα σύστημα Linux, μπορείτε να εγκαταστήσετε το Git μέσω του διαχειριστή πακέτων της διανομής σας, ενώ σε ένα σύστημα Windows μπορείτε να κατεβάσετε και να εγκαταστήσετε το Git από την επίσημη ιστοσελίδα του.

3.1.9 Github

Το GitHub είναι μια πλατφόρμα αποθετηρίων κώδικα (code repositories) που βασίζεται στο Git. Λειτουργεί ως μια υπηρεσία φιλοξενίας για τον κώδικά σας, επιτρέποντάς σας να αποθηκεύετε, να διαχειρίζεστε και να συνεργάζεστε σε έργα λογισμικού με άλλους προγραμματιστές. Είναι μια από τις πιο δημοφιλείς πλατφόρμες ανοικτού κώδικα στον κόσμο και προσφέρει πολλές λειτουργίες και εργαλεία για τη διαχείριση των αναπτυσσόμενων έργων.[11]

Εικόνα 11: Github

Για να χρησιμοποιήσετε το GitHub, πρέπει πρώτα να δημιουργήσετε έναν λογαριασμό στην πλατφόρμα. Αφού δημιουργήσετε τον λογαριασμό, μπορείτε να δημιουργήσετε ένα αποθετήριο (repository) για τον κώδικά σας και να ανεβάσετε τον κώδικά σας σε αυτό. Μετά την αποθήκευση του κώδικα, μπορείτε να επιτρέψετε σε άλλους προγραμματιστές να συνεργαστούν μαζί σας στο έργο, να δημιουργήσετε κλαδιά (branches) για να αναπτύξετε νέα χαρακτηριστικά και να προτείνετε αλλαγές (pull requests) στον κώδικά σας.

3.1.10 Intellij IDEA community edition

Το IntelliJ IDEA Community Edition είναι ένα δημοφιλές ενσωματωμένο περιβάλλον ανάπτυξης (IDE) που παρέχει πολλές λειτουργίες και εργαλεία για την ανάπτυξη εφαρμογών Java. Με την υποστήριξη της JetBrains, το IntelliJ IDEA προσφέρει ένα φιλικό περιβάλλον που βοηθά τους προγραμματιστές να αυξήσουν την παραγωγικότητά τους και να δημιουργήσουν υψηλής ποιότητας εφαρμογές.

Μετά την εγκατάσταση, μπορείτε να χρησιμοποιήσετε το IntelliJ IDEA Community Edition για να δημιουργήσετε νέα έργα ή να ανοίξετε υπάρχοντα έργα. Το IDE παρέχει πολλά εργαλεία για τη γρήγορη ανάπτυξη, την αυτόματη συμπλήρωση κώδικα, την αναγνώριση

[11] Πηγή εικόνας

σφαλμάτων και την αποσφαλμάτωση. Μπορείτε να προσαρμόσετε το περιβάλλον εργασίας σας, να προσθέσετε επεκτάσεις και να αξιοποιήσετε τις πολλές δυνατότητες που προσφέρει το IntelliJ IDEA.[12]

Εικόνα 12: Intellij IDEA

Οι βασικές λειτουργίες που προσφέρει το IntelliJ IDEA Community Edition περιλαμβάνουν:

1. Επεξεργασία κώδικα: Παρέχει ευέλικτο περιβάλλον επεξεργασίας κώδικα με σύνταξης επισημάνσεων, αυτόματη συμπλήρωση κώδικα, αναδιάταξη κώδικα και άλλα χαρακτηριστικά που βοηθούν στην αποτελεσματική γραφή κώδικα.
2. Οργάνωση έργων: Μπορείτε να δημιουργήσετε και να οργανώσετε τα έργα σας με τη χρήση των δομών φακέλων, να διαχειριστείτε τις εξαρτήσεις του έργου σας και να εκτελέσετε διάφορες εργασίες όπως την αντιγραφή και τη μετακίνηση αρχείων.
3. Αντίγραφο ασφαλείας και επαναφορά: Το IntelliJ IDEA παρέχει εργαλεία για τη δημιουργία αντιγράφων ασφαλείας του έργου σας και την αποκατάσταση από προηγούμενες εκδόσεις.
4. Αποσφαλμάτωση (Debug): Μπορείτε να εκτελέσετε και να αποσφαλματώσετε τον κώδικά σας με τη χρήση των ενσωματωμένων εργαλείων αποσφαλμάτωσης.
5. Ομαδική εργασία: Μπορείτε να συνεργαστείτε με άλλους προγραμματιστές χρησιμοποιώντας συστήματα ελέγχου εκδόσεων όπως το Git και το GitHub, και να κάνετε εναλλακτικές αλλαγές και συγχώνευση των αλλαγών σας.

Αυτά είναι μερικά από τα βασικά χαρακτηριστικά του IntelliJ IDEA Community Edition. Με την εγκατάσταση του IDE, μπορείτε να ξεκινήσετε να αναπτύσσετε τις εφαρμογές σας και να αξιοποιήσετε τις δυνατότητες του.

3.1.11 Visual Studio Code

Το Visual Studio Code είναι ένα δημοφιλές περιβάλλον ανάπτυξης κειμένου (IDE) που παρέχει πληθώρα λειτουργιών για την ανάπτυξη λογισμικού. Είναι δωρεάν και ανοιχτού κώδικα, και υποστηρίζει διάφορες γλώσσες προγραμματισμού.

Οι βασικές λειτουργίες και χαρακτηριστικά του Visual Studio Code περιλαμβάνουν:

[12] Πηγή εικόνας

1. Υποστήριξη πολλαπλών γλωσσών προγραμματισμού: Το Visual Studio Code παρέχει υποστήριξη για μια ευρεία γκάμα γλωσσών προγραμματισμού, όπως JavaScript, TypeScript, Python, C#, Java και πολλές άλλες.
2. Επεξεργασία κώδικα: Το IDE παρέχει προηγμένες δυνατότητες επεξεργασίας κώδικα, συμπεριλαμβανομένης της αυτόματης συμπλήρωσης κώδικα, της αναδιάταξης κώδικα, της αναζήτησης και αντικατάστασης κώδικα και πολλά άλλα.
3. Ενσωματωμένη αποσφαλμάτωση: Το Visual Studio Code παρέχει ενσωματωμένες δυνατότητες αποσφαλμάτωσης που επιτρέπουν τον έλεγχο και τη διόρθωση σφαλμάτων κατά την ανάπτυξη των εφαρμογών.[13]
4. Επέκταση λειτουργικότητας: Το IDE επιτρέπει την επέκταση της λειτουργικότητάς του μέσω πρόσθετων. Υπάρχουν χιλιάδες πρόσθετα διαθέσιμα στο Marketplace του Visual Studio Code που προσθέτουν επιπλέον χαρακτηριστικά και εργαλεία.

Εικόνα 13: Visual Studio Code

Για να εγκαταστήσετε το Visual Studio Code, ακολουθήστε τα εξής βήματα:

1. Μεταβείτε στην επίσημη ιστοσελίδα του Visual Studio Code στον ακόλουθο σύνδεσμο: https://code.visualstudio.com/.
2. Κάντε κλικ στο κουμπί "Download" για να κατεβάσετε το αρχείο εγκατάστασης για το λειτουργικό σύστημα που χρησιμοποιείται (Windows, macOS ή Linux).
3. Ακολουθήστε τις οδηγίες εγκατάστασης που παρέχονται από τον εγκαταστάτη του Visual Studio Code.
4. Αφού ολοκληρωθεί η εγκατάσταση, μπορείτε να ανοίξετε το Visual Studio Code και να ξεκινήσετε να το χρησιμοποιείτε.

[13] Πηγή εικόνας

3.2 Ανάλυση Backend Εφαρμογής[14]

Κατά τη μεταφορά της υπάρχουσας εφαρμογής στο νέο web service, αντιμετωπίστηκαν προκλήσεις στον σχεδιασμό και την υλοποίηση του νέου project. Αρχικά, μεταφέρθηκαν οι καίριες κλάσεις που υπολογίζουν τις μετρικές και οι κλάσεις που αποθηκεύουν τα δεδομένα ανάλυσης και αλληλεπιδρούν με τη βάση δεδομένων. Η σχεδίαση και η υλοποίηση της βάσης δεδομένων απαιτούσαν πρόσθετη κατανόηση και σχεδιαστική σκέψη. Έπειτα από έρευνα και ανάλυση του τρόπου σύνδεσης της εφαρμογής με τη βάση, έγινε αλλαγή των αρχικών κλάσεων προκειμένου να υποστηρίζουν τη σύνδεση με τη βάση.

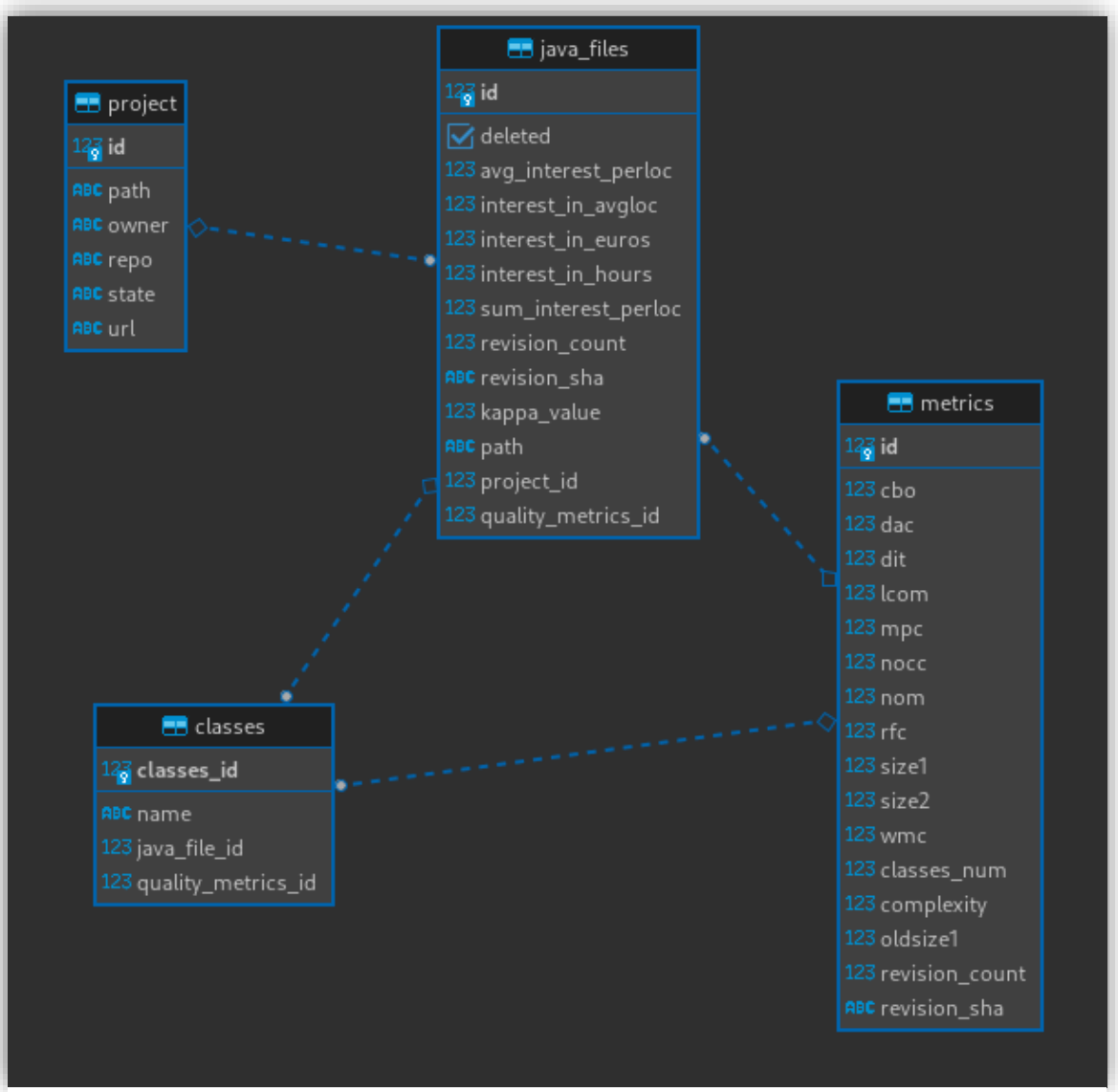
Μελετώντας τις κλάσεις και τις συσχετίσεις που απαιτούνταν για την ευκολία κατανόησης και διαχείρισης της βάσης, δημιουργήθηκε το παρακάτω διάγραμμα που αναπαριστά τις σχέσεις μεταξύ των πινάκων.

Η κλάση Project αντιπροσωπεύει ένα έργο και έχει μια σχέση με πολλές κλάσεις CalculatedJavaFile μέσω του πεδίου javaFiles. Κάθε CalculatedJavaFile ανήκει σε ένα συγκεκριμένο Project.

- Η κλάση CalculatedJavaFile αντιπροσωπεύει έναν υπολογισμένο αρχείο Java και ανήκει σε ένα συγκεκριμένο Project. Έχει μια σχέση με πολλές κλάσεις CalculatedClass μέσω του πεδίου classes.
- Η κλάση QualityMetrics αντιπροσωπεύει τις ποιοτικές μετρήσεις ενός αρχείου Java ή μιας κλάσης. Έχει μια σχέση με το CalculatedJavaFile μέσω του πεδίου qualityMetrics. Περιέχει πολλά πεδία για να αποθηκεύσει διάφορα μετρήσιμα χαρακτηριστικά του αρχείου ή της κλάσης.
- Η κλάση CalculatedClass αντιπροσωπεύει μια υπολογισμένη κλάση στο αρχείο Java. Έχει μια σχέση με ένα μόνο CalculatedJavaFile μέσω του πεδίου javaFile και μια σχέση με ένα QualityMetrics μέσω του πεδίου qualityMetrics.

Έτσι έχοντας τις συσχετίσεις και έτοιμες τις κλάσεις repositories φτιαγμένες ώστε να υπάρχει η σύνδεση της εφαρμογής με την βάση, ξεκίνησε η υλοποίηση των πρώτων request,

[14] Πηγαίος Κώδικας



Εικόνα 14: Συσχετήσεις Βάσης Δεδομένων

δημιουργώντας την κλάση ProjectController όπου αναλύεται παρακάτω. Ο ProjectController διαχειρίζεται τα ακόλουθα αιτήματα (requests):

- GET /api/project: Επιστρέφει μια λίστα με όλα τα projects. Το αίτημα αυτό δεν απαιτεί παραμέτρους.
- GET /api/project/owner: Επιστρέφει μια λίστα με τα projects βάσει του ιδιοκτήτη (owner). Απαιτεί την παράμετρο owner, η οποία είναι το όνομα του ιδιοκτήτη των projects.
- GET /api/project/repo: Επιστρέφει μια λίστα με τα projects βάσει του αποθετηρίου (repo). Απαιτεί την παράμετρο repo, η οποία είναι το όνομα του αποθετηρίου των projects.

- GET /api/project/url: Επιστρέφει ένα project βάσει του URL του. Απαιτεί την παράμετρο url, η οποία είναι το URL του project.
- GET /api/project/owner_and_repo: Επιστρέφει ένα project βάσει του ιδιοκτήτη (owner) και του αποθετηρίου (repo). Απαιτεί τις παραμέτρους owner και repo, που είναι αντίστοιχα το όνομα του ιδιοκτήτη και το όνομα του αποθετηρίου του project.
- GET /api/project/{id}: Επιστρέφει ένα project βάσει του αναγνωριστικού ID του. Απαιτεί την παράμετρο {id}, που είναι το αναγνωριστικό ID του project.
- GET /api/project/state: Επιστρέφει την κατάσταση ενός project βάσει του URL του. Απαιτεί την παράμετρο url, η οποία είναι το URL του project.
- DELETE /api/project/url: Διαγράφει ένα project βάσει του URL του. Απαιτεί την παράμετρο url, η οποία είναι το URL του project.
- DELETE /api/project/{id}: Διαγράφει ένα project βάσει του αναγνωριστικού ID του. Απαιτεί την παράμετρο {id}, που είναι το αναγνωριστικό ID του project.

Κάθε αίτημα επιστρέφει μια απόκριση (response) σε μορφή JSON ή XML, ανάλογα με τον τύπο αποδοχής (Accept) που ορίζεται στην κεφαλίδα (header) του αιτήματος. Η απόκριση περιλαμβάνει τα αντίστοιχα projects ή μηνύματα σφάλματος, εάν κάτι πάει στραβά ή εάν το αναζητούμενο project δεν υπάρχει.

Ο τελευταίος και πιο συμαντικός controller που διαχειρίζεται την ανάλυση, τον υπολογισμό και σερβίρισμα των διάφορων μετρικών είναι ο AnalysisController. Ουσιαστικά, αυτός ο ελεγκτής διαχειρίζεται αιτήσεις για μετρήσεις και αναλύσεις μετρικών για ένα συγκεκριμένο έργο. Περιλαμβάνει διάφορες μέθοδους HTTP GET και POST που επιστρέφουν αποτελέσματα μετρήσεων για το έργο, όπως τα ενδιαφέροντα ανά αλλαγή, τα αρχεία υψηλού ενδιαφέροντος, τις μετρικές επαναχρησιμοποίησης του έργου και πολλά άλλα.

Η κάθε μέθοδος στον controller είναι σημείο εισόδου για μια συγκεκριμένη ανάλυση ή μέτρηση και επιστρέφει τα αποτελέσματα ως απάντηση σε αιτήματα HTTP. Οι παράμετροι των μεθόδων καθορίζουν τα διάφορα φίλτρα και περιορισμούς που μπορούν να εφαρμοστούν στα αποτελέσματα. Αυτός ο ελεγκτής διαχειρίζεται τα παρακάτω αιτήματα (requests) για την ανάλυση μετρικών ενός έργου:

- POST /api/analysis: Αυτό το αίτημα δημιουργεί μια νέα ανάλυση για ένα συγκεκριμένο αποθετήριο. Παίρνει ένα αντικείμενο NewAnalysisDTO ως σώμα αιτήματος, το οποίο αποτελείται από ένα git url και έναν git user με ένα username και ένα password η κάποιο access token, και εκκινεί μια νέα ανάλυση με βάση αυτό το αντικείμενο, δημιουργώντας και αποθηκεύοντας αυτόματα το project από το git url στην βάση. Όσο

για τον git user και τα crendetials του, μπορούν να είναι κενά εφόσον τα αποθετήρια τα οποία θέλουμε να αναλύσουμε είναι δημόσια. Αν είναι ιδιωτικά θα χρειαστούν τα credentials, μόνο όμως για να προσπελαστούν τα commits, δεν αποθηκεύεται κάποιο ιδιωτικό στοιχείο στο server ή στην βάση.

- GET /api/analysis/cumulativeInterest: Αυτό το αίτημα επιστρέφει το συγκεντρωτικό ενδιαφέρον ανά αναθεώρηση (commit) για ένα συγκεκριμένο αποθετήριο. Παίρνει την παράμετρο url ως υποχρεωτική παράμετρο και την παράμετρο sha ως προαιρετική παράμετρο.
- GET /api/analysis/interestPerCommitFile: Αυτό το αίτημα επιστρέφει το ενδιαφέρον ανά αναθεώρηση (commit) για ένα συγκεκριμένο αρχείο σε ένα αποθετήριο. Παίρνει τις παραμέτρους url (υποχρεωτική), sha (προαιρετική) και filePath (προαιρετική).
- GET /api/analysis/interestChange: Αυτό το αίτημα επιστρέφει τη μεταβολή του ενδιαφέροντος ανά αναθεώρηση (commit) για ένα συγκεκριμένο αποθετήριο. Παίρνει τις παραμέτρους url (υποχρεωτική) και sha (προαιρετική).
- GET /api/analysis/fileInterestChangeByCommitAndFile: Αυτό το αίτημα επιστρέφει τη μεταβολή του ενδιαφέροντος ενός συγκεκριμένου αρχείου για ένα συγκεκριμένο αναθεωρητήριο και commit. Παίρνει τις παραμέτρους url (υποχρεωτική), sha (υποχρεωτική) και filePath (υποχρεωτική).
- GET /api/analysis/fileInterestChange: Αυτό το αίτημα επιστρέφει τη μεταβολή του ενδιαφέροντος αρχείων για ένα συγκεκριμένο αποθετήριο. Παίρνει την παράμετρο url ως υποχρεωτική παράμετρο.
- GET /api/analysis/normalizedInterest: Αυτό το αίτημα επιστρέφει το κανονικοποιημένο ενδιαφέρον ανά αναθεώρηση (commit) για ένα συγκεκριμένο αποθετήριο. Παίρνει την παράμετρο url ως υποχρεωτική παράμετρο και την παράμετρο sha ως προαιρετική παράμετρο.
- GET /api/analysis/normalizedInterestPerFile: Αυτό το αίτημα επιστρέφει το κανονικοποιημένο ενδιαφέρον ανά αρχείο για ένα συγκεκριμένο αποθετήριο. Παίρνει την παράμετρο url ως υποχρεωτική παράμετρο.
- GET /api/analysis/highInterestFiles: Αυτό το αίτημα επιστρέφει τα αρχεία με υψηλό ενδιαφέρον για ένα συγκεκριμένο αποθετήριο και commit. Παίρνει την παράμετρο url ως υποχρεωτική παράμετρο, την παράμετρο sha ως προαιρετική παράμετρο και την παράμετρο limit ως προαιρετική παράμετρο.

- GET /api/analysis/projectReusabilityMetrics: Αυτό το αίτημα επιστρέφει τις μετρικές επαναχρησιμοποίησης του έργου για ένα συγκεκριμένο αποθετήριο. Παίρνει την παράμετρο url ως υποχρεωτική παράμετρο και την παράμετρο limit ως προαιρετική παράμετρο.
- GET /api/analysis/fileReusabilityMetrics: Αυτό το αίτημα επιστρέφει τις μετρικές επαναχρησιμοποίησης αρχείων για ένα συγκεκριμένο αποθετήριο. Παίρνει την παράμετρο url ως υποχρεωτική παράμετρο και την παράμετρο limit ως προαιρετική παράμετρο.
- GET /api/analysis/fileReusabilityMetricsByCommit: Αυτό το αίτημα επιστρέφει τις μετρικές επαναχρησιμοποίησης αρχείων για ένα συγκεκριμένο αποθετήριο και commit. Παίρνει την παράμετρο url ως υποχρεωτική παράμετρο και την παράμετρο limit ως προαιρετική παράμετρο.
- GET /api/analysis/fileReusabilityMetricsByCommitAndFile: Αυτό το αίτημα επιστρέφει τις μετρικές επαναχρησιμοποίησης ενός συγκεκριμένου αρχείου για ένα συγκεκριμένο αποθετήριο και commit. Παίρνει τις παραμέτρους url (υποχρεωτική), sha (υποχρεωτική) και filePath (υποχρεωτική).

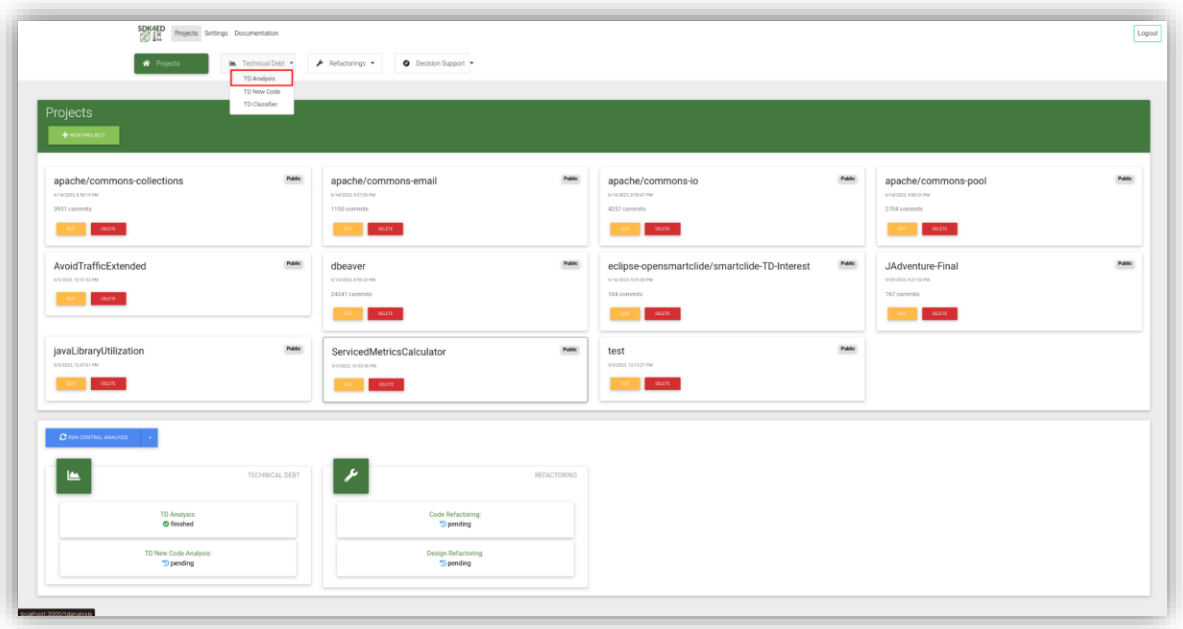
Αυτά είναι τα διαθέσιμα αιτήματα που μπορεί να διαχειριστεί ο AnalysisController στην εφαρμογή ανάλυσης αποθετηρίων. Ο ελεγκτής επεξεργάζεται αυτά τα αιτήματα και επιστρέφει τα αποτελέσματα μετά από ανάλυση των δεδομένων του έργου.

Συνοψίζοντας, προηγουμένως παρουσιάστηκαν και εξηγήθηκαν όλα τα αιτήματα που μπορούν να γίνουν από το εργαλείο μέτρησης ποιότητας λογισμικού που αναπτύχθηκε. Στο επόμενο κεφάλαιο, θα δούμε ότι δεν χρησιμοποιούνται όλα αυτά στο πρόγραμμα πελάτη (frontend). Ορισμένα από αυτά είναι περιττά ή δεν καλούνται ποτέ. Ωστόσο, λόγω της δυνατότητας μελλοντικής επέκτασης και της πιθανής προσωπικής χρήσης του εργαλείου μέτρησης ποιότητας, δημιουργήθηκαν οι παραπάνω δυνατότητες. Αυτό οφείλεται στην ήδη αποθηκευμένη πληροφορία στη βάση δεδομένων της εφαρμογής. Τέλος, τα αιτήματα προσαρμόστηκαν για τη χρήση των δεδομένων στο πρόγραμμα πελάτη, κυρίως για απλούστευση των πληροφοριών. Δεν ήταν αναγκαία η οπτικοποίηση όλων των αποτελεσμάτων, αλλά κυρίως για να μην υπερφορτωθεί η διεπαφή χρήστη και να διατηρηθεί η ευχρηστία, ειδικά σε περιπτώσεις μεγάλων αναλύσεων για commits σε ένα έργο.

3.3 Ανάλυση Frontend Εφαρμογής[15]

Φτάνοντας σε ένα σημείο όπου απαιτούνταν η διασύνδεση του frontend με το backend, αποφασίστηκε να επιδειχθεί ο απαραίτητος χρόνος για την κατανόηση των αλλαγών που θα έπρεπε να γίνουν καθώς και των σημείων του UI της SDK4ED πλατφόρμας που θα επηρεάζονταν.

Στην εργασία αντιμετωπίζεται το τεχνικό χρέος και ο υπολογισμός του επιτοκίου και άλλων μετρικών τεχνικού χρέους. Για τον λόγο αυτό, απαιτήθηκαν κυρίως αλλαγές στη σελίδα TD Analysis, η οποία είναι προσβάσιμη μέσω του μενού "Technical Debt", όπως φαίνεται στην παρακάτω εικόνα. Αυτή η σελίδα αποτελεί την κεντρική σελίδα της εφαρμογής.

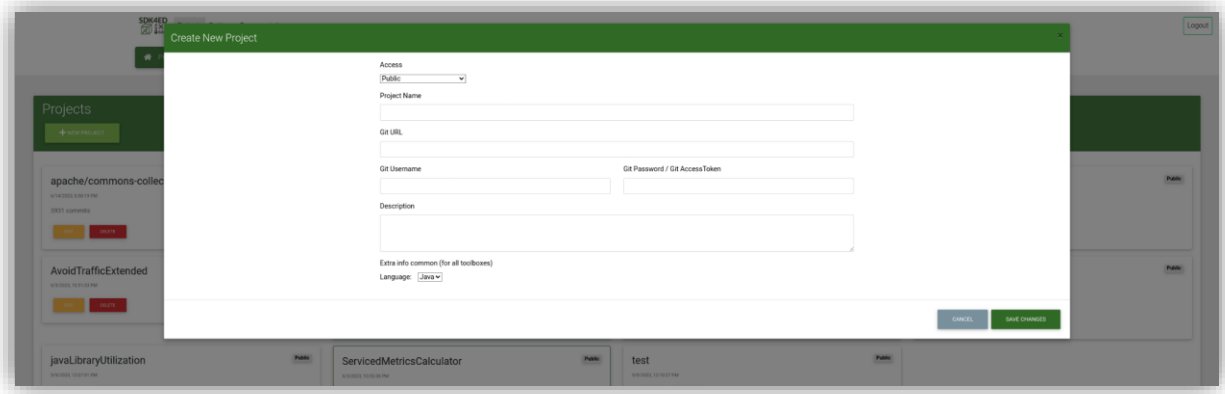


Εικόνα 15: Αρχική σελίδα SDK4ED

Όπως φαίνεται στο κεντρικό μενού εμφανίζονται όλα τα project τα οποία έχουν εισαχθεί στην εφαρμογή από τον χρήστη, ανεξάρτητα από το αν είναι ανελλημένα ή όχι. Η εισαγωγή γίνεται από το κουμπί το οποίο υπάρχει μέσα στην κεφαλίδα του panel, Projects. Πατώντας το κουμπί εμφανίζεται η Create New Project φόρμα, όπου μπορείς να επιλέξεις αν το αποθετήριο git είναι δημόσιο ή ιδιωτικό, το όνομα του project που ο χρήστης θέλει να ορίσει, το url από οποιαδήποτε υπηρεσία κάνει host git repositories. Τα username και password or access token σε περίπτωση που το αποθετήριο είναι ιδιωτικό. Μία περιγραφή που θέλει ο χρήστης να εισάγει για το συγκεκριμένο project και τέλος η γλώσσα του project. Προς το

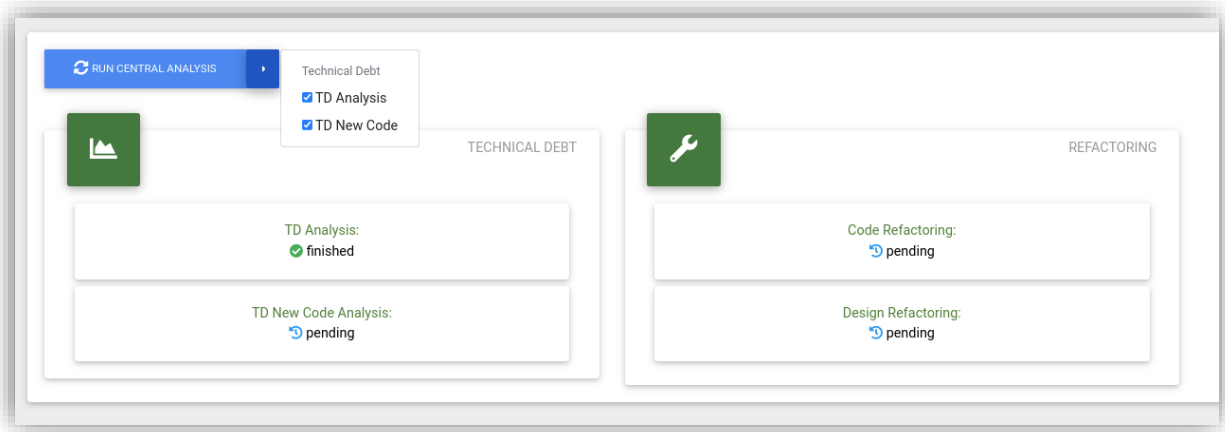
[15] Πηγαίος Κώδικας

παρών, υποστηρίζεται μόνο η Java αλλά μπήκε σαν επιλογή και η C για λόγους εμφάνισης του πεδίου επιλογής.



Εικόνα 16: Δημιουργία νέου project

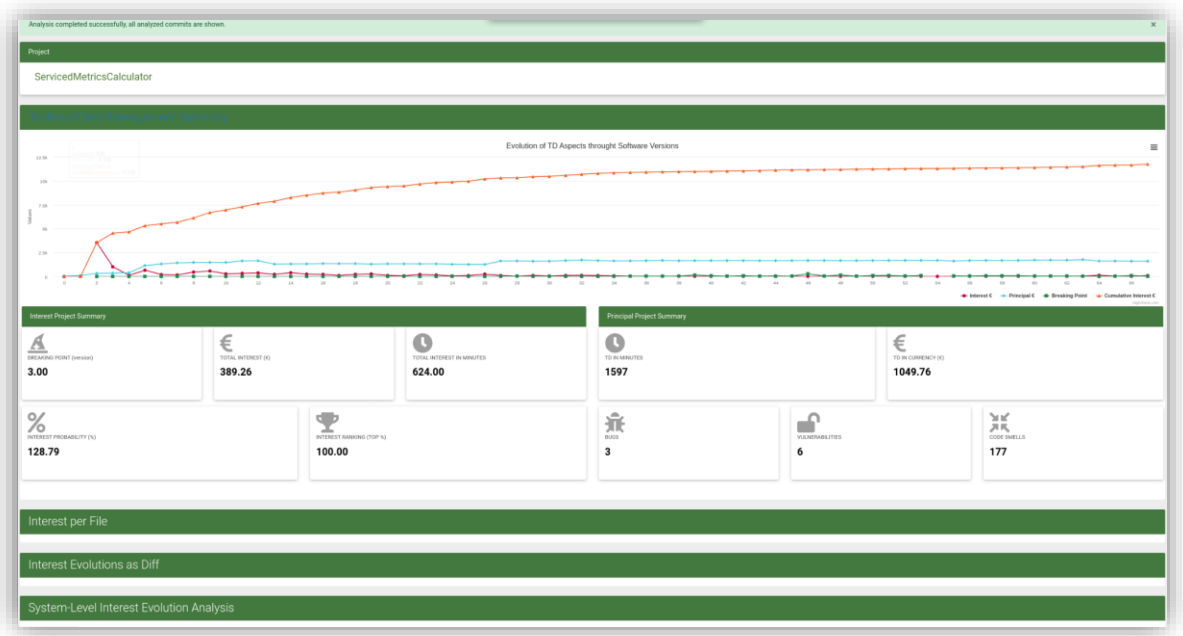
Όταν προστίθεται ένα έργο, εμφανίζεται ένα μικρό πλαίσιο με έναν τίτλο που αντιστοιχεί στο όνομα που έχει δοθεί, το αν είναι δημόσιο ή ιδιωτικό το αποθετήριο, την ημερομηνία δημιουργίας και την περιγραφή, εάν υπάρχει. Υπάρχουν επίσης δύο κουμπιά: το κουμπί "Επεξεργασία" επιτρέπει τη διαχείριση και την τροποποίηση του συγκεκριμένου έργου, και το κουμπί "Delete" επιτρέπει την αφαίρεσή του. Το παράθυρο που εμφανίζεται όταν πατηθεί το κουμπί "Edit" είναι το ίδιο με αυτό που περιγράφηκε παραπάνω, δηλαδή το "Create New Project". Το κουμπί "Delete" εμφανίζει ένα παράθυρο διαλόγου για την επιβεβαίωση της διαγραφής.



Εικόνα 17: Πλαίσιο κεντρικής ανάλυσης

Πιο κάτω στη σελίδα εμφανίζεται ένα ακόμα πλαίσιο όπου βρίσκονται το κουμπί "Run Central Analysis" με ένα πτυσσόμενο πεδίο δίπλα του. Όταν η σελίδα φορτώνεται για πρώτη φορά, το κουμπί "Run Central Analysis" είναι απενεργοποιημένο (disabled) και εμφανίζεται

σε γκρι χρώμα. Για να ενεργοποιηθεί και να επιτραπεί η εκκίνηση της ανάλυσης, πρέπει να επιλεγεί ένα από τα έργα στο πρώτο πλαίσιο. Το πτυσσόμενο πεδίο έχει προεπιλεγμένες όλες τις συνδεδεμένες εφαρμογές που πραγματοποιούν ανάλυση του κώδικα για πιθανές μετρήσεις. Η πρώτη επιλογή, TD Analysis, αντιστοιχεί στο backend που αναπτύχθηκε για την παρούσα εργασία. Η δεύτερη επιλογή (TD New Code) είναι επίσης σημαντική αναφοράς για αυτήν την εργασία, καθώς χρησιμοποιείται για τον υπολογισμό ορισμένων τιμών, όπως το breaking point, ή για την προβολή ορισμένων πεδίων στη σελίδα TD Analysis. Προκειμένου να γίνει η ανάλυση, το συγκεκριμένο έργο πρέπει να έχει αναλυθεί ή να γίνεται ανάλυση ταυτόχρονα με την ανάλυση που ξεκινά από το backend της εργασίας. Καθώς προχωράμε παρακάτω, βλέπουμε τις καταστάσεις (statuses) που είναι δυναμικές και καθώς η ανάλυση είναι σε εξέλιξη, το εικονίδιο αλλάζει σε έναν ανεξάντλητο loader. Όταν ολοκληρωθεί, εμφανίζεται ένα πράσινο σημάδι επιτυχίας (tick), ενώ σε περίπτωση αποτυχίας εμφανίζεται ένα κόκκινο ερωτηματικό.



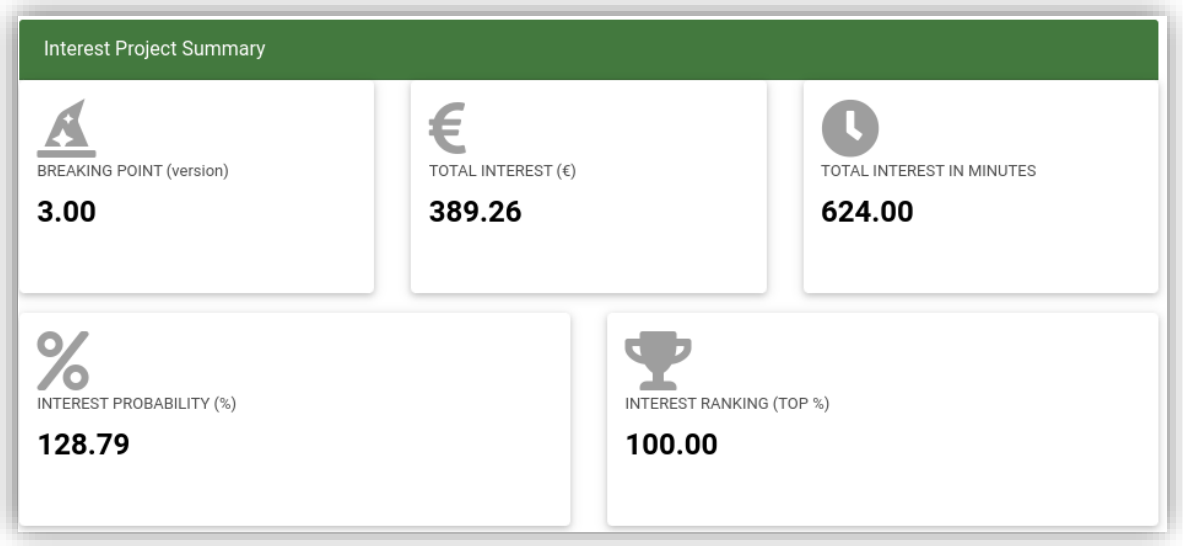
Εικόνα 18: Σελίδα ανάλυσης τεχνικού χρέους

Εισερχόμενος πλέον ο χρήστης στην σελίδα "TD Analysis" η οποία άλλαξε κατά βάση, αντικρίζει την παραπάνω εικόνα. Όπου έχουμε κατα σειρά τα εξής πλαίσια. Το πρώτο πλαίσιο, είναι ένα "Alert" το οποίο ενημέρωνει τον χρήστη αν η ανάλυση του project στο οποίο έχει επιλέξει να του παρουσιαστούν τα δεδομένα έχει ολοκληρωθεί ή είναι σε εξέλιξη ή δεν έχει ξεκινήσει καθόλου ή υπήρξε κάποιο πρόβλημα και η ανάλυση διακόπηκε. Σε αυτό το σημείο να αναφέρουμε ότι τα πεδία δεν αλλάζουν σε πραγματικό χρόνο αλλά για οποιαδήποτε

ενημέρωση ο χρήστης θα χρειαστεί να ξαναφορτώσει την σελίδα για να πάρει τις τελευταίες αλλαγές.

Επόμενο πλαίσιο όπως φαίνεται και στην παραπάνω εικόνα είναι το όνομα το οποίο ο χρήστης επέλεξε το project το οποίο θα εμφανίζονται τα δεδομένα ανάλυσης. Παρακάτω, παρατηρείτε ένα πλαίσιο το οποίο περιλαμβάνει τέσσερις γραφικές παραστάσεις μέσα σε ένα διάγραμμα "HighCharsJS for React" όπως προαναφέρθηκε ότι χρησιμοποιήθηκε. Επίσης υπάρχουν και 2 πλαίσια ακόμα το αριστερά το οποίο αναφέρει γενικές πληροφορίες για το interest το οποίο έχει αναλυθεί στο project μέχρι εκείνη την στιγμή και το δεξιά εμφανίζει γενικές πληροφορίες για το principal του project. Στο διάγραμμα παρουσιάζονται οι εξής γραφικές παραστάσεις για κάθε ανελημένο commit του project:

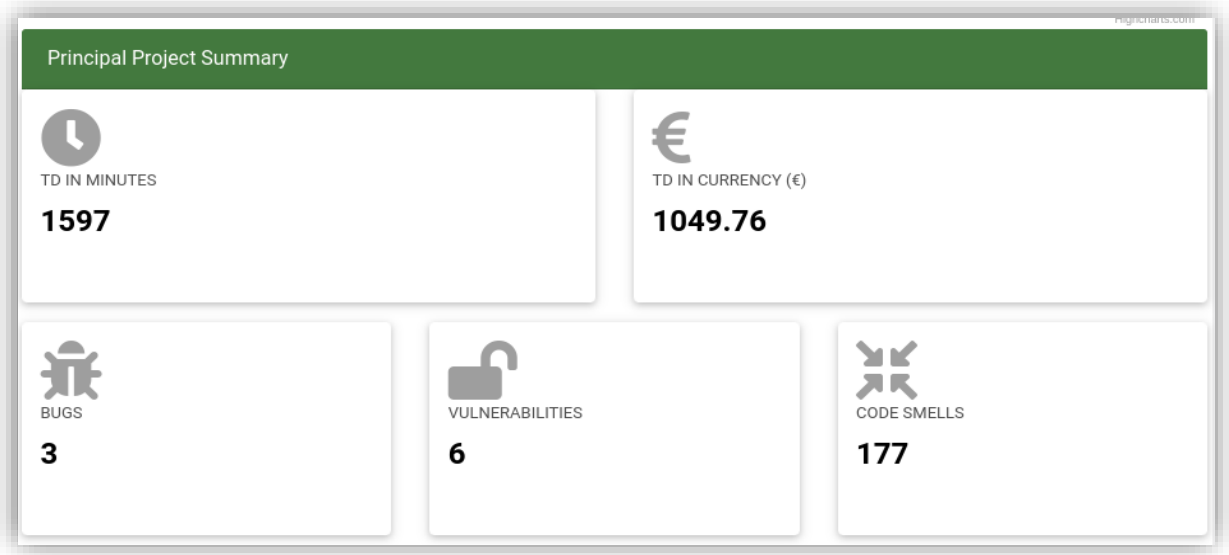
- Interest σε Ευρώ
- Principal σε Ευρώ
- Breaking Point
- Σωρευτικό (Cummulative) Interest σε Ευρώ



Εικόνα 19: Πλαίσιο Interest Project Summary

Το αριστερό πλαίσιο το οποίο εμφανίζεται στην σελίδα περιέχει όπως ειπώθηκε γενικές πληροφορίες για το interest, όπως το "breaking point" το οποίο υπολογίζεται στρογγυλοποιόντας το αποτέλεσμα της διαίρεσης της τιμής του principal σε λεπτά από το τελευταίο commit με το συνολικό interest σε λεπτά. Δηλαδή έχουμε, $\left\lceil \frac{last\ Principal\ in\ Minutes}{total\ Interest\ in\ Minutes} \right\rceil$. Το "Total Interest (€)" το οποίο είναι το άθροισμα του "Interest (In €)" κάθε κλάσης που υπάρχει

στο τελευταίο commit του project. Το “Total Interest in Minutes” όπου είναι το άθροιμα του “Interest (In Hours)” επί 60 λεπτά της ώρας έτσι ώστε να υπολογιστεί το αποτέλεσμα σε λεπτά. Το “Interest Probability (%)” υποδεικνύει το πόσο άλλαξε το πρότζεκτ κατά την διάρκεια από το πρώτο μέχρι το τελευταίο commit. Ο υπολογισμός γίνεται ως εξής: $\frac{\left(\sum_{0}^{lastRevisionCount} "Change\ Between\ Revisions\ (In\ \%)"\right)}{lastRevisionCount}$. Τέλος, το “Interest Ranking (Top %)” το οποίο υποδεικνύει τι ποσοστό κατέχει το συγκεκριμένο project σε interest σε σχέση με όλα τα project που υπάρχουν ανελημένα εκείνη την στιγμή, $\frac{totalInterestForCalculatedProject}{totalInterestForAllProjects} \times 100$.



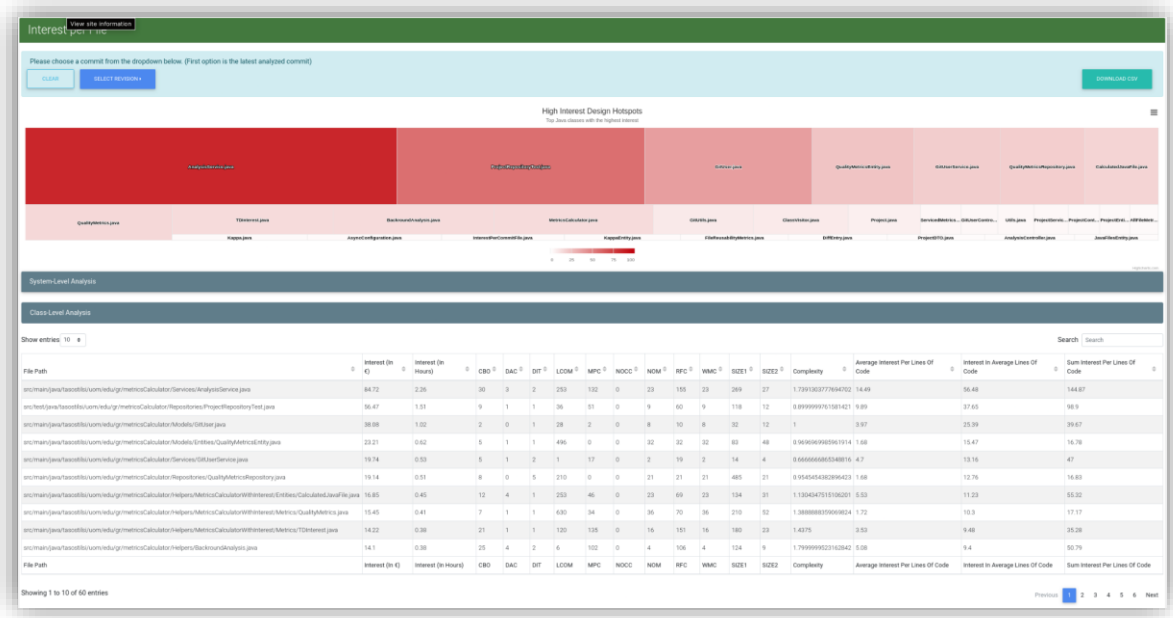
Εικόνα 20: Πλαίσιο Principal Project Summary

Το δεξί πάνελ, όπως γράφτηκε παραπάνω εμφανίζεται μια εικόνα του principal του συγκεκριμένου project όπου η ανάλυση του principal γίνεται από ξεχωριστή εφαρμογή η οποία εκτελείτε από την πρώτη σελίδα στην επιλογή “TD New Code” από την επιλογή που υπάρχει δίπλα από το κουμπί “Run Central Analysis”. Επομένως, έχουμε συνοπτικά τις παρακάτω ανελυμένες τιμές, οι οποίες είναι ονομαστικά:

- Principal in Minutes
- Principal in Euros
- Bugs
- Vulnerabilities
- Code Smells

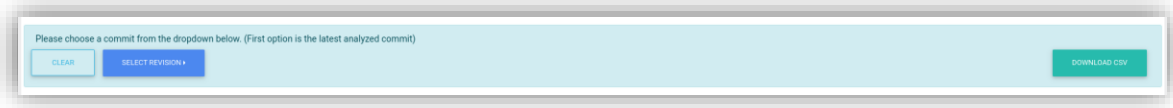
Οι παραπάνω τιμές εμφανίζονται στην τελευταία κατάσταση του αναλυμένου project.

Στην συνέχεια, έχουμε 3 πτυσσόμενα πεδία τα οποία περιλαμβάνουν τους πίνακες με τις μετρικές οι οποίες παρουσιάστηκαν στην προηγούμενη ενότητα, παράλληλα με κάποιες γραφικές παραστάσεις οι οποίες αναδεικνύουν τις αναλύσεις και οπτικοποιούν τα αποτελέσματα δίνοντας ένα πιο ισχυρό εργαλείο στον χρήστη να καταλάβει την πληροφορία που αποτυπώνεται στον πίνακα ακόμα περισσότερο και πιο άμεσα. Επίσης, σε αυτά τα πτυσσόμενα πλαίσια έχουμε κάποια κοινά πεδία τα οποία έχουν την ίδια ακριβώς λειτουργικότητα θα παραλειφθούν όπου επαναλαμβάνονται.



Εικόνα 21: Πλαίσιο Interest per File

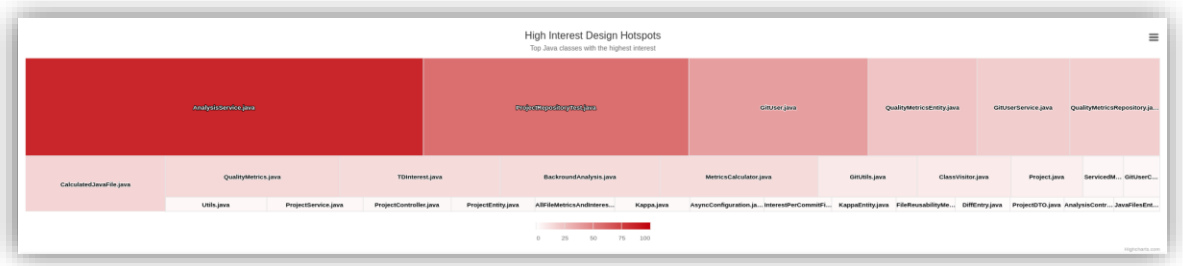
Το πρώτο κατά σειρά panel, “Interest per File”, παρουσιάζει τις όλες τις μετρικές οι οποίες παρουσιάζονται στην προηγούμενη ενότητα για κάθε αρχείο το οποίο είναι στην τελευταία κατάσταση της αναλυμένης στιγμής στην οποία βρίσκεται το project.



Εικόνα 22: Πλαίσιο επιλογής revision count

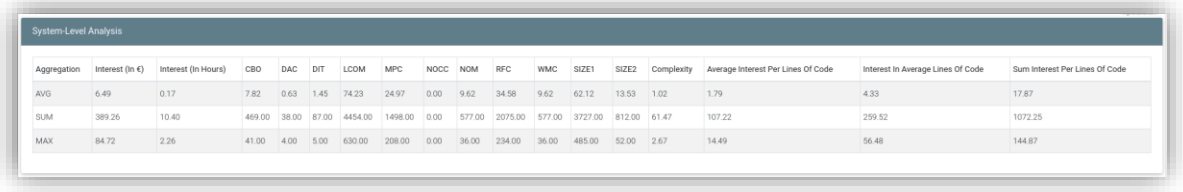
Όπως παρατηρείται όμως μπορεί ο χρήστης να θέλει να δει ιστορικά δεδομένα, από προηγούμενα commits τα οποία είναι αναλημένα, οπότε για αυτό τον λόγο έχει ενσωματωθεί ένα alert box το οποίο περιέχει μέσα του 3 κουμπιά. Το πρώτο είναι το κουμπί “Clear”, το

οποίο επαναφέρει τον πίνακα στην αρχική κατάσταση που είναι όταν όπως φορτώνεται η σελίδα. Δεξιά του “Clear”, το κουμπί “Select Revision”, όπου μπορεί κάποιος να δει πόσα και ποια revision counts είναι αναλημένα έτσι ώστε να έχει την επιλογή να δει τα δεδομένα από την εκάστωτε χρονική στιγμή. Πατώντας αυτό το κουμπί παρατηρείτε ένα search box στην αρχή έτσι ώστε να μπορεί ο χρήστης να αναζητήσει τον αριθμό που θέλει σε περίπτωση που τα commits είναι πάρα πολλά και με αυτόν το τρόπο δεν θα χρειάζεται να κάνει scroll για να επιλέξει κάποιο συγκεκριμένο revision count. Από την άλλη μεριά, προστέθηκε το κουμπί “Download CSV”, με το οποίο πατώντας το κατεβαίνει ένα αρχείο csv όπου περιλαμβάνει τα δεδομένα του πίνακα. Σε αυτό το σημείο να σημειωθεί, ότι πατώντας το κουμπί “Download CSV” τα δεδομένα που θα παράξει το csv αρχείο είναι αυτά που υπάρχουν εκείνη την στιγμή στον πίνακα, κατά αυτόν τον τρόπο, επιτυγχάνεται η παραγωγή αρχείων csv και για παλαιότερα commits τα οποία μπορεί ο χρήστης να επιθυμεί να παράξει ένα αρχείο με δεδομένα.



Εικόνα 23: Διάγραμμα High Interest Hotspots

Παρακάτω το διάγραμμα το οποίο διαφαίνεται παρουσιάζει τις 30 πιο μεγάλες σε Interest κλάσεις. Οι κλάσεις με το πιο έντονο χρώμα είναι και αυτές με την μεγαλύτερη τιμή “Interest (In €)”.

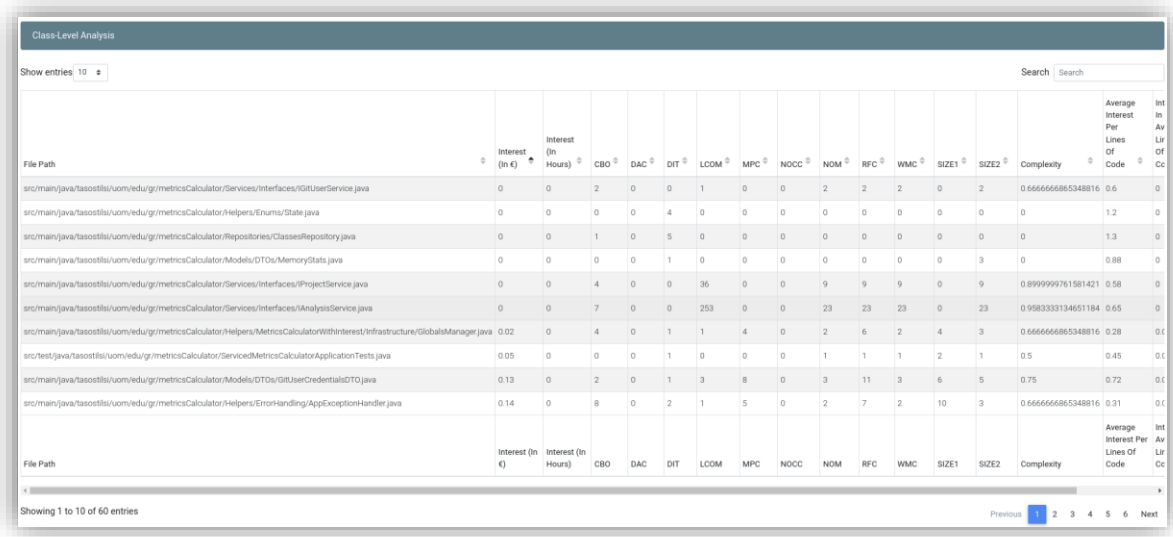

System-Level Analysis

Aggregation	Interest (In €)	Interest (In Hours)	CBO	DAC	DIT	LCOM	MPC	NOCC	NOM	RFC	WMC	SIZE1	SIZE2	Complexity	Average Interest Per Lines Of Code	Interest In Average Lines Of Code	Sum Interest Per Lines Of Code
AVG	6.49	0.17	7.82	0.63	1.45	74.23	24.97	0.00	9.62	34.58	9.62	62.12	13.53	1.02	1.79	4.33	17.87
SUM	389.26	10.40	469.00	38.00	87.00	4454.00	1498.00	0.00	577.00	2075.00	577.00	3727.00	812.00	61.47	107.22	259.52	1072.25
MAX	84.72	2.26	41.00	4.00	5.00	630.00	208.00	0.00	36.00	234.00	36.00	485.00	52.00	2.67	14.49	56.48	144.87

Εικόνα 24: Πίνακας System-Level Analysis Interest per File

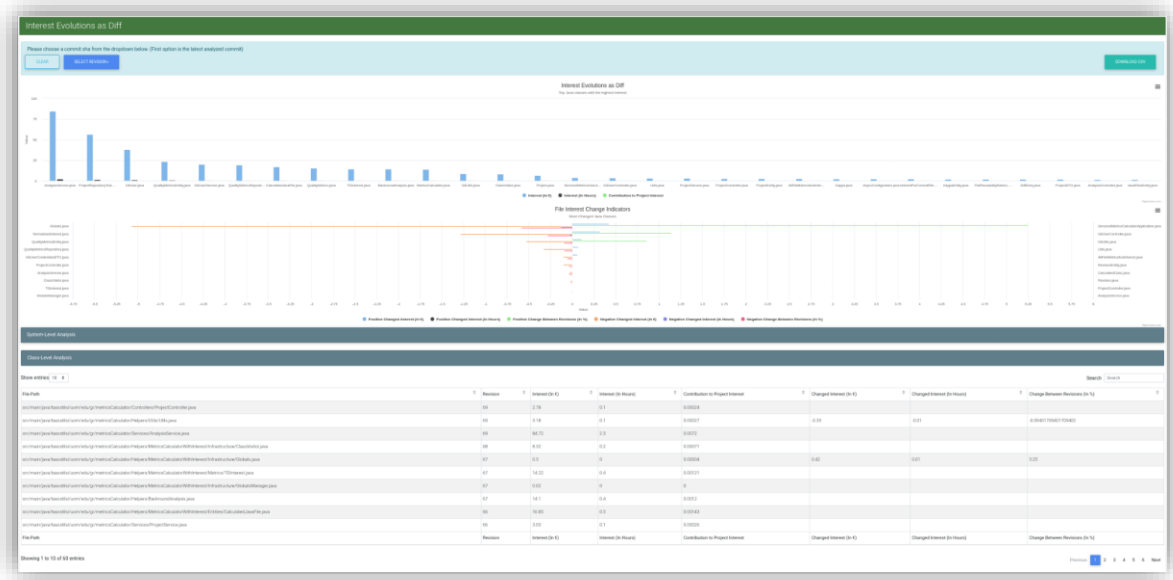
Το επόμενο επεκτυνόμενο panel “System-Level Analysis” είναι ένας πινακας ο οποίος παρουσιάζει την μέγιστη τιμή, το άθροισμα και τον μέσο κάθε μετρικής που υπάρχει στον πίνακα. Άξιο αναφοράς ό,τι υπολογίζεται εκ νέου κάθε φορά που ο χρήστης επιλέγει να

εμφανίσει παλαιότερο αναλυμένο commit, καθώς επίσης ότι συμπεριλαμβάνονται οι τιμές του μέσα στο csv αρχείο.



Εικόνα 25: Πίνακας Class-Level Analysis Interest per File

Τέλος, ο πίνακας όπου εμπεριέχει τις τιμές για κάθε αρχείο το οποίο περιλαμβάνεται στο εκάστοτε commit, το οποίο θα επιλέξει ο χρήστης να παρουσιάσει. Στον πίνακα αυτό υπάρχει και πλαίσιο αναζήτησης, σε περίπτωση που ο χρήστης χρειαστεί να αναζητήσει κάποιο αρχείο ή κάποια συγκεκριμένη τιμή του πίνακα, καθώς φυσικά και κάθε στήλη του έχει την επιλογή ταξινόμησης.



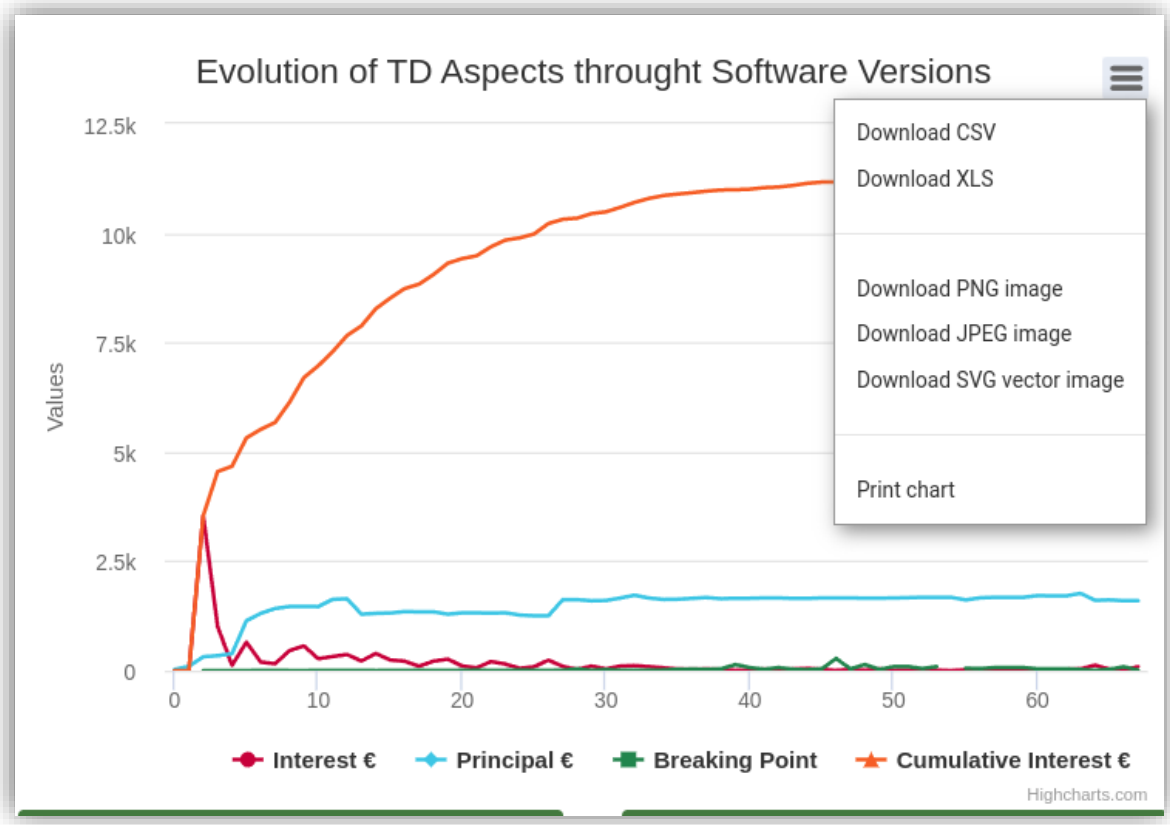
Εικόνα 26: Πλαίσιο Interest Evolution as Diff

Το τελευταίο κατά σειρά πλαίσιο, με τίτλο “System-Level Interest Evolution Analysis”, μας παρουσιάζει τις μετρικές ανά αναλυμένου commit συνολικά.

Changed Interest (In €)	Η μεταβολή του Interest από το προηγούμενο ακριβώς αναλυμένο commit σε ευρώ
Changed Interest (In Hours)	Η μεταβολή του Interest από το προηγούμενο ακριβώς αναλυμένο commit σε ώρες
Change Between Revisions (In %)	Η μεταβολή του Interest από το προηγούμενο ακριβώς αναλυμένο commit σε ποσοστό
Normalized Interest (In €)	Είναι η κανονικοποίηση της τιμής Interest (in €) σε σχέση με τις γραμμές κώδικα του κάθε commit. Η τιμή αυτή υπολογίζεται ως εξής: $\frac{\sum_0^{lastRevisionCount} Interest\ (in\ €)}{\sum_0^{lastRevisionCount} SIZE1\ (or\ LoC)}$
Normalized Interest (In Hours)	Είναι η κανονικοποίηση της τιμής Interest (in Hours) σε σχέση με τις γραμμές κώδικα του κάθε commit. Η τιμή αυτή υπολογίζεται ως εξής: $\frac{\sum_0^{lastRevisionCount} Interest\ (in\ Hours)}{\sum_0^{lastRevisionCount} SIZE1\ (or\ LoC)}$

Πίνακας 2: Παρουσίαση Μετρικών “System-Level Interest Evolution Analysis” πίνακα.

Ο χρήστης σε αυτό το panel, μπορεί να δει μια συνοπτική εικόνα της μεταβολής του interest για ολόκληρο το commit και για κάθε commit ξεχωριστά. Με τον ίδιο τρόπο όπως και με τα προηγούμενα πλαίσια μπορεί να παραχθεί csv αρχείο το οποίο περιλαμβάνει τις τιμές του πίνακα αυτού.



Εικόνα 27: Επιλογές διαγραμμάτων

Κλείνοντας τις αλλαγές οι οποίες πραγματοποιήθηκαν στο frontend, επισημαίνεται ότι σε όλα τα διαγράμματα υπάρχει ένα κουμπί το με τις επιλογές οι οποίες φαίνονται στην παραπάνω εικόνα. Όπου μπορούν να παράξουν αρχεία, csv και xls με τις τιμές οι οποίες έχει μέσα κάθε γραφική παράσταση, καθώς επίσης να παραχθούν διάφορα αρχεία εικόνων αλλά δίνεται και η επιλογή να εκτυπωθεί απευθείας η γραφική παράσταση.

4 Επίλογος

4.1 Σύνοψη και συμπεράσματα

Η διπλωματική εργασία προσφέρει σημαντικά συμπεράσματα για τα προβλήματα που προέκυψαν και παρουσιάστηκαν στην εισαγωγή. Καταρχάς, ανιχνεύσαμε τη σημασία των μετρικών λογισμικού για την αξιολόγηση της ποιότητας και της απόδοσης του κώδικα. Η ανάλυση των μετρικών προσφέρει πολύτιμες πληροφορίες για την πρόοδο των προγραμματιστών και την βελτίωση του κώδικα. Επιπλέον, διαπιστώσαμε ότι η αξιολόγηση μετρικών λογισμικού συμβάλλει στην προετοιμασία και τη διαχείριση προγραμματιστικών έργων.

Συμπεραίνουμε ότι το εργαλείο υπολογισμού μετρικών που αναπτύξαμε αποτελεί ένα σημαντικό βήμα προς τη βελτίωση της ποιότητας και της απόδοσης του κώδικα. Επιπλέον, αναφέρουμε τον προοπτικό επεκτασιμότητας του εργαλείου μας για να υποστηρίζει περισσότερες γλώσσες προγραμματισμού και να ανιχνεύει αρχιτεκτονικές παραβιάσεις. Συνολικά, η διπλωματική εργασία μας προσφέρει σημαντική συνεισφορά στην προαγωγή της ποιότητας και της απόδοσης του κώδικα, καθώς και στη βελτίωση της διαδικασίας ανάπτυξης λογισμικού.

4.2 Όρια και περιορισμοί της έρευνας

Ως μέρος της αξιολόγησης της εργασίας και του αναπτυγμένου εργαλείου, προκύπτουν κάποια πιθανά όρια και περιορισμοί που θα πρέπει να ληφθούν υπόψη. Ανάμεσα σε αυτούς μπορούν να περιλαμβάνονται:

1. Περιορισμένη κάλυψη γλωσσών προγραμματισμού: Το εργαλείο μπορεί να είναι περιορισμένο σε συγκεκριμένες γλώσσες προγραμματισμού και να μην υποστηρίζει όλες τις γλώσσες.
2. Περιορισμένη δυνατότητα ανίχνευσης σύνθετων προβλημάτων: Ενδέχεται να υπάρχουν περιορισμοί στην ανίχνευση ορισμένων σύνθετων προβλημάτων κώδικα, ιδίως όταν απαιτείται περαιτέρω ανάλυση ή κατανόηση του περιβάλλοντος εκτέλεσης.

3. Ανάγκη για προσαρμογή και εξατομίκευση: Ενδέχεται να απαιτείται προσαρμογή του εργαλείου σε συγκεκριμένα περιβάλλοντα ανάπτυξης ή προτιμήσεις των προγραμματιστών. Αυτό μπορεί να απαιτεί πρόσθετο χρόνο και πόρους.
4. Περιορισμένη ακρίβεια: Οι μετρήσεις και οι αναλύσεις που παρέχονται από το εργαλείο ενδέχεται να μην είναι απόλυτα ακριβείς και να απαιτείται ανθρώπινη κρίση για την εκτίμηση των αποτελεσμάτων.

Αυτοί οι περιορισμοί και όρια πρέπει να ληφθούν υπόψη κατά την αξιολόγηση και την εφαρμογή της εργασίας και του εργαλείου. Επιπλέον, μπορεί να είναι απαραίτητος περαιτέρω πειραματισμός και βελτιώσεις για την αντιμετώπιση αυτών των περιορισμών και τη βελτίωση της απόδοσης του εργαλείου.

4.3 Μελλοντικές Επεκτάσεις

Όλα τα έργα λογισμικού έχουν περιθώρια επέκτασης, εξέλιξης και αύξηση της ποιότητας, παρακάτω παρουσιάζονται κάποιες ιδέες για μελλοντικές επεκτάσεις, εξέλιξης αλλά και δημιουργίας test ώστε να να ανιχνεύονται και να διορθώνονται πιθανά προβλήματα πιο εύκολα και γρήγορα.

- Υποστήριξη για περισσότερες γλώσσες προγραμματισμού: Επεκτείνετε το εργαλείο σας για να υποστηρίζει μετρήσεις μετρικών και ανάλυσης ποιότητας για περισσότερες γλώσσες προγραμματισμού, όχι μόνο για την τρέχουσα γλώσσα που υποστηρίζετε.
- Ανάλυση ομοιότητας κώδικα: Ενσωματώστε λειτουργίες ανάλυσης ομοιότητας κώδικα για την εντοπισμό παρόμοιων τμημάτων κώδικα. Αυτό μπορεί να βοηθήσει στον εντοπισμό ανακυκλωμένου κώδικα, ανιχνεύοντας παρόμοιες λογικές δομές και μοτίβα σε διάφορα μέρη του έργου σας.
- Ανάλυση κώδικα ανα release version ή git tag: Η ανάλυση κώδικα ανά release version ή git tag παρέχει αναλυτικές πληροφορίες σχετικά με τον κώδικα για κάθε έκδοση του λογισμικού, ανιχνεύει αλλαγές, προσθήκες και διαγραφές κώδικα μεταξύ των διαφορετικών εκδόσεων ή git tags, εμφανίζει τις διαφορές σε επίπεδο γραμμής κώδικα για καλύτερη κατανόηση των αλλαγών και παρουσιάζει αναλυτικές μετρήσεις και μετρικές για κάθε έκδοση, όπως πλήθος γραμμών κώδικα, πολυπλοκότητα και κάλυψη δοκιμών. Επιπλέον, ανιχνεύει πιθανά προβλήματα ή σφάλματα που προκύπτουν σε συγκεκριμένες εκδόσεις και παρέχει ιστορικό κώδικα για εύκολη πρόσβαση σε παλαιότερες εκδόσεις και ανάκτηση πληροφοριών. Αυτές οι δυνατότητες επιτρέπουν

την ανάπτυξη καλύτερων πρακτικών, τη βελτίωση της ποιότητας του κώδικα και την ευκολότερη ανίχνευση και διόρθωση προβλημάτων.

- Ανίχνευση αρχιτεκτονικών παραβιάσεων: Προσθέστε δυνατότητες ελέγχου της συμμόρφωσης του κώδικα σε σχέση με προκαθορισμένες αρχιτεκτονικές προτάσεις. Αυτό μπορεί να περιλαμβάνει τον έλεγχο του σεβασμού των οδηγιών αρχιτεκτονικής και την αναγνώριση παραβιάσεων στον κώδικα.
- Η δημιουργία unit tests, integration tests και API tests είναι σημαντικές επεκτάσεις για το εργαλείο υπολογισμού μετρικών λογισμικού. Αυτές οι δοκιμές επιτρέπουν τον έλεγχο της λειτουργικότητας, της ορθότητας και της απόδοσης του κώδικα. Η προσθήκη αυτών των ειδών των δοκιμών θα ενισχύσει την αξιοπιστία, την ασφάλεια και την απόδοση του εργαλείου υπολογισμού μετρικών. Επίσης, θα δώσει τη δυνατότητα στους προγραμματιστές να εντοπίζουν και να διορθώνουν πιθανά προβλήματα πιο εύκολα και γρήγορα.

5 Βιβλιογραφία

Βιβλία:

Pressman, Roger S. Software Engineering: A Practitioner's Approach. 8th ed., McGraw-Hill Education, 2014.

Martin, Robert C. Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall, 2009.

Brooks Jr., Frederick P. The Mythical Man-Month: Essays on Software Engineering. Anniversary ed., Addison-Wesley Professional, 1995.

Sipser, Michael. Introduction to the Theory of Computation. 3rd ed., Cengage Learning, 2012.

van Vliet, Hans. Software Engineering: Principles and Practice. 3rd ed., Wiley, 2008.

Martin, Robert C. Agile Software Development, Principles, Patterns, and Practices. Pearson Education, 2003.

Schach, Stephen R. Object-Oriented and Classical Software Engineering. 8th ed., McGraw-Hill Education, 2010.

Sommerville, Ian. Software Engineering. 10th ed., Pearson Education Limited, 2016.

Witten, Ian H., and David Bainbridge. How to Build a Digital Library. 2nd ed., Morgan Kaufmann Publishers, 2010.

Άρθρα:

Ampatzoglou, A., Michailidis, A., Sarikyriakidis, C., Ampatzoglou, A., Chatzigeorgiou, A., & Avgeriou, P. (2018). A Framework for Managing Interest in Technical Debt: An Industrial Validation. In Proceedings of the 2018 International Conference on Technical Debt (pp. 115–

124). Association for Computing Machinery.

Marija Jankovic, Dionysios Kehagias, Miltiadis Siavvas, Dimitrios Tsoukalas, Alexandros Chatzigeorgiou, “The SDK4ED Approach to Software Quality Optimization and Interplay Calculation”, 15th China-Europe International Symposium on Software Engineering Education (CEISEE ’19)

Bakota, Tibor, et al. “A Probabilistic Software Quality Model.” IEEE Transactions on Software Engineering, vol. 42, no. 1, Jan. 2016, pp. 89-110.

Basili, Victor R., et al. “The Empirical Investigation of Perspective-Based Reading.” Empirical Software Engineering: An International Journal, vol. 1, no. 2, June 1996, pp. 133-164.

Bouwers, Eric, et al. “Quantifying the Analyzability of Software Architectures.” Information and Software Technology, vol. 54, no. 11, Nov. 2012, pp. 1219-1236.

Cunningham W (1992) The WyCash portfolio management system. In: OOPSLA ’92 Addendum to the Proceedings on Object-oriented Programming Systems Languages and Applications (Addendum), ACM Press New York NY USA pp 29–30

Fowler M (2009) Technical debt quadrant: balancing short-term gain and long-term pain [online]. Available at https://martinfowler.com/bliki/TechnicalDebtQuadrant.html

McConnell S (2007) Technical debt: 7 ways to manage it [online]. Available at https://www.construx.com/resources/technical-debt-7-ways-to-manage-it/

Moschou A (2019) Technical debt management in software engineering: a systematic mapping study [online]. Available at https://dspace.lib.uom.gr/bitstream/2159/24570/4/MoschouAthanasiaMsc2019.pdf

Sarikyriakidis C (2020) Technical debt measurement in software engineering [online]. Available at

https://dspace.uowm.gr/xmlui/bitstream/handle/123456789/641/ChristosSarikyriakidis_finalthesis.pdf;jsessionid=AA99DAD8AB4253938C791BA827F49D9B?sequence=1

Ιστοσελίδες:

SDK4ED (2020) SDK4ED - software development toolkit for energy optimization and technical debt elimination [online]. Available at https://sdk4ed.eu/

Singh S et al (2020) A systematic literature review on technical debt management in agile software development process [online]. Available at https://link.springer.com/article/10.1007/s42979-020-00406-6

Software Quality (2020) ASQ - American Society for Quality [online]. Available at https://asq.org/quality-resources/software-quality

What is technical debt? (2020) [online]. Available at https://el.myservername.com/what-is-technical-debt

Εγχειρίδιο των web services της W3C. "Το World Wide Web Consortium (W3C) παρέχει ένα εγχειρίδιο για τα web services." https://www.w3.org/TR/ws-arch/

Εγχειρίδιο των web services της Microsoft. "Η Microsoft παρέχει ένα εγχειρίδιο με πληροφορίες για την ανάπτυξη και τη χρήση web services με την τεχνολογία της."https://docs.microsoft.com/en-us/dotnet/core/extensions/developing-web-apis

"Ανασκόπηση των web services." Wikipedia. https://en.wikipedia.org/wiki/Web_service

Amazon Web Services. "The Difference Between SOAP and REST." https://aws.amazon.com/compare/the-difference-between-soap-rest/

TTMind. "Web Services: REST vs gRPC." https://www.ttmind.com/techpost/WEB-SERVICES-REST-VS-gRPC

Docker. https://www.docker.com/.

Docker Documentation. https://docs.docker.com/.

Docker Hub. https://hub.docker.com/.

Docker YouTube channel. https://www.youtube.com/docker.

Git. https://git-scm.com/.

Highcharts. https://www.highcharts.com/.

IntelliJ IDEA Community Edition. https://www.jetbrains.com/idea/.

MDB React. https://mdbootstrap.com/docs/react/.

MDB React GitHub repository. https://github.com/mdbootstrap/mdb-react-ui-kit.

Maven. https://maven.apache.org/.

Maven GitHub repository. https://github.com/apache/maven.

Pro Git Book. https://git-scm.com/book/en/v2.

Atlassian Git Tutorial. https://www.atlassian.com/git/tutorials.

Git Official Documentation. https://git-scm.com/doc.

Highcharts for ReactJS GitHub repository. https://github.com/highcharts/highcharts-react.

Highcharts for ReactJS Documentation. https://www.highcharts.com/docs/index.

Maven User Guide. https://maven.apache.org/guides/index.html.

Maven Examples. https://maven.apache.org/examples/index.html.

MDB React Examples and Tutorials. https://mdbootstrap.com/docs/react/getting-started/quick-start/.

MDB React Community on Stack Overflow. https://stackoverflow.com/questions/tagged/mdbreact.