

Ατομική Διπλωματική Εργασία

**ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ ΑΝΑΛΥΣΗΣ
ΑΚΑΔΗΜΑΪΚΩΝ ΕΡΓΑΣΙΩΝ ΓΙΑ ΕΝΤΟΠΙΣΜΟ ΑΙ-GENERATED
ΠΕΡΙΕΧΟΜΕΝΟΥ**

Μιχάλης Παπακώστας

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ



ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2026

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

**Σχεδιασμός και Υλοποίηση Συστήματος Ανάλυσης Ακαδημαϊκών Παραδοτέων για
Εντοπισμό AI-Generated Περιεχομένου**

Μιχάλης Παπακώστας

Επιβλέπων Καθηγητής
Κα. Ελένη Κωνσταντίνου

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2026

Δηλώνω ότι η παρούσα διατριβή και οποιοδήποτε συνοδευτικό λογισμικό αποτελούν δικό μου πρωτότυπο έργο, εκπονημένο σε πλήρη συμμόρφωση με τον Κώδικα Δεοντολογίας για την Έρευνα του Πανεπιστημίου Κύπρου (<https://www.ucy.ac.cy/legislation/kodikas-deontologias-kai-politikes/>), και με πλήρη ευθύνη εκ μέρους μου για την ακρίβεια, την ακεραιότητα και την εγκυρότητα όλου του περιεχομένου.

Ευχαριστίες

Πρωτίστως, θα ήθελα να εκφράσω τις θερμές μου ευχαριστίες στην επιβλέπουσα καθηγήτριά μου, κα. Ελένη Κωνσταντίνου, για την πολύτιμη καθοδήγηση, την εμπιστοσύνη και τη συνεχή υποστήριξή της κατά τη διάρκεια εκπόνησης της διπλωματικής μου εργασίας. Οι χρήσιμες υποδείξεις της και το ευχάριστο κλίμα συνεργασίας που δημιούργησε συνέβαλαν καθοριστικά στην ολοκλήρωση αυτού του συστήματος.

Επιπλέον, θα ήθελα να ευχαριστήσω όλους όσους συνέβαλαν με οποιονδήποτε τρόπο στην ολοκλήρωση της εργασίας αυτής, είτε μέσω επιστημονικής καθοδήγησης είτε μέσω παροχής δεδομένων και τεχνικής υποστήριξης.

Ιδιαίτερες ευχαριστίες οφείλω στην οικογένειά μου, για τη συνεχή στήριξη και την ενθάρρυνση που μου προσέφεραν καθ' όλη τη διάρκεια των φοιτητικών μου χρόνων. Ήταν πάντα δίπλα μου σε κάθε δύσκολη στιγμή, στηρίζοντάς με, με κάθε δυνατό τρόπο και δίνοντάς μου τη δύναμη να συνεχίσω. Χωρίς τη δική τους βοήθεια, κατανόηση και πίστη σε εμένα, η πορεία αυτή θα ήταν πολύ πιο δύσκολη.

Τέλος, ευχαριστώ τους φίλους και τους συμφοιτητές μου, που ήταν πάντα δίπλα μου και με στήριζαν καθ' όλη τη διάρκεια των σπουδών μου. Η στήριξή τους ήταν αναπόσπαστο κομμάτι της ολοκλήρωσης αυτής της προσπάθειας και τους οφείλω ένα μεγάλο ευχαριστώ.

Περίληψη

Η παρούσα διπλωματική εργασία αφορά τον σχεδιασμό και την ανάπτυξη ενός διαδικτυακού συστήματος με τίτλο « Σχεδιασμός και Υλοποίηση Συστήματος Ανάλυσης Ακαδημαϊκών Εργασιών για Εντοπισμό AI-Generated Περιεχομένου», το οποίο έχει ως στόχο την υποστήριξη στους καθηγητές στη διαδικασία αξιολόγησης εργασιών που ενδέχεται να περιέχουν περιεχόμενο παραγόμενο από εργαλεία Τεχνητής Νοημοσύνης. Η εργασία υλοποιήθηκε στο πλαίσιο της γενικότερης ανάγκης για αναβάθμιση των μηχανισμών ακαδημαϊκής αξιολόγησης, σε μια εποχή κατά την οποία η χρήση τέτοιων εργαλείων από φοιτητές γίνεται ολοένα και πιο διαδεδομένη [9].

Η ανάπτυξη του συστήματος χωρίστηκε σε δύο φάσεις. Στην πρώτη φάση μελετήθηκαν σύγχρονες τεχνικές για την ανίχνευση AI-generated περιεχομένου, με έμφαση στις διαφορετικές μεθόδους που εφαρμόζονται για κείμενο και για κώδικα. Παράλληλα σχεδιάστηκε η αρχιτεκτονική και η δομή των δεδομένων του συστήματος, η οποία οργανώνει τις πληροφορίες σε μαθήματα, ακαδημαϊκά εξάμηνα, παραδοτέα και ομάδες φοιτητών.

Στη δεύτερη φάση υλοποιήθηκε η εφαρμογή ως διαδικτυακή πλατφόρμα. Το σύστημα επιτρέπει στους καθηγητές να διαχειρίζονται τα μαθήματα και τα παραδοτέα των φοιτητών τους και να λαμβάνουν αυτόματα αποτελέσματα ανάλυσης για τα αρχεία που ανεβαίνουν. Η ανάλυση γίνεται διαφορετικά ανάλογα με τον τύπο του αρχείου, καλύπτοντας τόσο έγγραφα κειμένου όσο και πηγαίο κώδικα σε πολλές γλώσσες προγραμματισμού. Τα αποτελέσματα παρουσιάζονται με τη μορφή ποσοστών, ώστε ο καθηγητής να μπορεί να λάβει τεκμηριωμένες αποφάσεις, χωρίς το σύστημα να αντικαθιστά την κρίση του. Σημαντικό μέρος της εργασίας αποτέλεσε και η εγκατάσταση του συστήματος σε server του Τμήματος, ώστε να μην παραμείνει στο επίπεδο του πρωτοτύπου, αλλά να αποτελεί λειτουργικό εργαλείο που μπορεί να χρησιμοποιηθεί στην πράξη. Για την υλοποίηση χρησιμοποιήθηκαν, μεταξύ άλλων, οι τεχνολογίες FastAPI [5] για το backend, Next.js [6] για το frontend,

MariaDB [7] ως σύστημα διαχείρισης βάσεων δεδομένων, καθώς και ο εξυπηρετητής Apache ως reverse proxy για την εξυπηρέτηση της διαδικτυακής πλατφόρμας μέσω HTTPS.

Η πρόσβαση γίνεται μέσω δύο ρόλων, του διαχειριστή και του καθηγητή. Οι χρήστες δημιουργούν λογαριασμό, επιβεβαιώνουν το email τους με κωδικό μιας χρήσης (OTP) και η ενεργοποίηση γίνεται μετά από έγκριση του διαχειριστή. Για τα έγγραφα κειμένου (DOCX) η ανάλυση γίνεται σε επίπεδο παραγράφου, ενώ για τα αρχεία πηγαίου κώδικα χρησιμοποιείται το DetectCodeGPT, μια μέθοδος που αξιοποιεί τεχνικές perturbation-based detection μαζί με tree-sitter parsers για την υποστήριξη πολλών γλωσσών προγραμματισμού.

Στόχος του συστήματος δεν είναι να αντικαταστήσει την κρίση του καθηγητή ούτε να αποτελέσει μηχανισμό τιμωρίας των φοιτητών. Πρόκειται για ένα υποστηρικτικό εργαλείο που δίνει ενδείξεις και βοηθάει σε μια πιο τεκμηριωμένη αξιολόγηση. Συνολικά, η εργασία προτείνει μια πρακτική και επεκτάσιμη λύση για ακαδημαϊκά περιβάλλοντα, ενώ ταυτόχρονα αναδεικνύει τις δυσκολίες και τους περιορισμούς που υπάρχουν στην ανίχνευση AI-generated περιεχομένου.

Περιεχόμενα

1. Εισαγωγή.....	1
1.1 Στόχος της διπλωματικής	1
1.2 Πρόβλημα και κίνητρο.....	3
1.3 Συνεισφορά της εργασίας	3
1.4 Μεθοδολογία Ανάπτυξης και Χρονοδιαγράμματα	4
1.5 Οργάνωση Κειμένου.....	9
1.6 Ορισμοί, Ακρωνύμια και Συντομογραφίες	12
2. Υπόβαθρο και Σχετικές Εργασίες	15
2.1 Τεχνητή Νοημοσύνη και παραγωγή περιεχομένου.....	15
2.2 AI-generated κείμενο και AI-generated κώδικας.....	16
2.3 Ανάλυση και εντοπισμός AI-generated περιεχομένου.....	17
2.4 Συστήματα διαχείρισης ακαδημαϊκών υποβολών.....	18
2.5 Σχετικές Εφαρμογές και Εργαλεία.....	19
3. Απαιτήσεις Συστήματος.....	23
3.1 Απαιτήσεις και Γενική Περιγραφή Συστήματος.....	23
3.1.1 Απαιτήσεις Διεπαφών.....	26
3.1.2 Λειτουργικές Απαιτήσεις.....	29
3.1.3 Μη Λειτουργικές Απαιτήσεις.....	30
3.2 Κατηγορίες Χρηστών.....	32
3.3 Περιορισμοί Συστήματος	33
3.4 Σενάρια Χρήσης.....	34
4. Σχεδίαση και Υλοποίηση	37
4.1 Αρχιτεκτονική Συστήματος	37
4.2 Εργαλεία Ανάπτυξης.....	39
4.2.1 Visual Studio Code.....	39
4.2.2 Γλώσσα Προγραμματισμού Python.....	40
4.2.3 FastAPI.....	40
4.2.4 Γλώσσα Περιγραφής HTML	40
4.2.5 Γλώσσα Μορφοποίησης CSS.....	41
4.2.6 Γλώσσα Προγραμματισμού JavaScript / TypeScript	41
4.2.7 React.....	41
4.2.8 Next.js.....	42
4.2.9 SQLAlchemy και Alembic	42
4.2.10 MariaDB	43

4.2.11 Σύστημα Διαχείρισης Βάσεων Δεδομένων phpMyAdmin.....	43
4.2.12 DetectCodeGPT.....	43
4.2.13 Tree-sitter.....	44
4.3 Ασφάλεια Συστήματος.....	44
4.4 Βάση Δεδομένων	45
4.4.1 Πίνακες Βάσης Δεδομένων	46
4.4.2 Διάγραμμα Οντοτήτων-Σχέσεων.....	51
4.5 Δομή Αρχείων.....	52
4.6 Βιβλιοθήκες.....	52
4.6.1 Backend (Python)	52
4.6.2 Frontend.....	52
5. Έλεγχος Λειτουργικότητας του Συστήματος.....	53
5.1 Μεθοδολογία Ελέγχου	53
5.2 Δοκιμές ανά Λειτουργία	54
5.2.1 Εγγραφή και Σύνδεση.....	54
5.2.2 Μαθήματα, Εξάμηνα και Παραδοτέα.....	54
5.2.3 Μαζική Υποβολή μέσω ZIP	55
5.2.4 Ανάλυση Αρχείων Κειμένου	55
5.2.5 Ανάλυση Αρχείων Κώδικα.....	56
5.2.6 Αναφορές Ανάλυσης	56
5.2.7 Στατιστικά και Analytics	57
5.2.8 Dashboard Διαχειριστή.....	57
5.2.9 Ρυθμίσεις Κατωφλίων	57
5.3 Συνοπτικά Αποτελέσματα.....	58
5.4 Αξιολόγηση Ευχρηστίας	59
6. Συζήτηση και Μελλοντικές Προεκτάσεις	62
6.1 Συζήτηση.....	62
6.2 Μελλοντικές Προεκτάσεις	64
7. Συμπεράσματα	67
7.1 Συνοπτική Εκτίμηση	67
Βιβλιογραφία.....	69
Παράρτημα Α.....	1
Παράρτημα Β.....	1
Παράρτημα Γ	1
Παράρτημα Δ.....	1

Κεφάλαιο 1

1. Εισαγωγή

1.1 Στόχος της διπλωματικής	1
1.2 Πρόβλημα και κίνητρο	3
1.3 Συνεισφορά της εργασίας	3
1.4 Μεθοδολογία Ανάπτυξης και Χρονοδιαγράμματα	4
1.5 Οργάνωση Κειμένου	9
1.6 Ορισμοί, Ακρωνύμια και Συντομογραφίες	12

1.1 Στόχος της διπλωματικής

Τα τελευταία χρόνια, τα εργαλεία Τεχνητής Νοημοσύνης χρησιμοποιούνται όλο και πιο συχνά από φοιτητές, καθώς μπορούν να παράγουν κείμενο, κώδικα και ολόκληρες εργασίες μέσα σε πολύ λίγα δευτερόλεπτα. Στον ακαδημαϊκό χώρο, τέτοια εργαλεία μπορούν να βοηθήσουν τους φοιτητές να κατανοήσουν δύσκολες έννοιες, να βελτιώσουν τη δομή ενός κειμένου ή να εξερευνήσουν λύσεις σε προγραμματιστικά προβλήματα. Η χρήση τους δεν είναι από μόνη της κάτι αρνητικό. Αντιθέτως, πολλά εκπαιδευτικά ιδρύματα ενθαρρύνουν τη χρήση τους, εφόσον γίνεται με διαφάνεια και μέσα στα όρια που θέτει ο κάθε καθηγητής [9, 23]. Ωστόσο, υπάρχουν περιπτώσεις όπου η χρήση τους ξεπερνά τα όρια της βοήθειας και αντικαθιστά ουσιαστικά την εργασία του φοιτητή. Το ερώτημα δεν είναι αν οι φοιτητές χρησιμοποιούν AI, αλλά σε τι βαθμό και αν ο καθηγητής έχει τρόπο να το εκτιμήσει.

Στόχος της διπλωματικής είναι η ανάπτυξη ενός συστήματος που παρέχει στους καθηγητές μια πρώτη ένδειξη για το κατά πόσο μια εργασία περιέχει περιεχόμενο που παράχθηκε από

εργαλεία Τεχνητής Νοημοσύνης. Το σύστημα εξυπηρετεί δύο λειτουργίες. Αρχικά, προσφέρει στον καθηγητή ένα περιβάλλον οργάνωσης των μαθημάτων που διδάσκει και των υποβολών των εργασιών των φοιτητών του. Ακολούθως, αναλύει αυτόματα τα αρχεία που υποβάλλονται, τόσο έγγραφα κειμένου όσο και πηγαίο κώδικα, και παρουσιάζει ποσοστά πιθανότητας χρήσης ΑΙ ανά παράγραφο ή ανά αρχείο. Τα αποτελέσματα αυτά λειτουργούν ως βοηθητική ένδειξη και όχι ως οριστική κατηγοριοποίηση των αρχείων. Σκοπός δεν είναι να αντικατασταθεί η αξιολόγηση του καθηγητή, αλλά να του δοθεί ένα εργαλείο που μπορεί να χρησιμοποιήσει ως αφετηρία για περαιτέρω διερεύνηση, αν το κρίνει απαραίτητο.

Πιο συγκεκριμένα, η εργασία στοχεύει στα εξής:

- Στη δημιουργία ενός κεντρικού περιβάλλοντος όπου ο καθηγητής μπορεί να διαχειρίζεται τα μαθήματα, τα παραδοτέα και τις υποβολές των φοιτητών του.
- Στην υλοποίηση μηχανισμών αυτόματης ανάλυσης που να καλύπτουν διαφορετικούς τύπους αρχείων, τόσο για έγγραφα κειμένου όσο και για πηγαίο κώδικα σε πολλές γλώσσες προγραμματισμού. Κάθε τύπος αρχείου αξιολογείται με τη μέθοδο που του ταιριάζει καλύτερα.
- Στην παρουσίαση των αποτελεσμάτων με τρόπο κατανοητό και χρήσιμο, ώστε ο καθηγητής να μπορεί να βγάλει τα δικά του συμπεράσματα.
- Στον σχεδιασμό μιας αρχιτεκτονικής που να είναι επεκτάσιμη, ώστε στο μέλλον να μπορούν να προστεθούν νέοι τύποι αρχείων ή πιο εξελιγμένες τεχνικές ανίχνευσης.
- Στην εγκατάσταση του συστήματος σε server του Τμήματος, ώστε να μην παραμείνει στο επίπεδο του πρωτοτύπου αλλά να μπορεί να χρησιμοποιηθεί στην πράξη.

Πέρα από τα παραπάνω, η εργασία επιδιώκει να συμβάλει σε μια πιο διαφανή και τεκμηριωμένη διαδικασία αξιολόγησης. Το σύστημα δίνει ενδείξεις και υποστηρικτική πληροφόρηση, αλλά η τελική απόφαση παραμένει πάντα στον καθηγητή. Δεν προορίζεται να λειτουργήσει ως μηχανισμός τιμωρίας των φοιτητών, αλλά ως εργαλείο που βοηθάει τους καθηγητές σε ένα συνεχώς εξελισσόμενο ακαδημαϊκό περιβάλλον.

1.2 Πρόβλημα και κίνητρο

Η αυξανόμενη χρήση εργαλείων Τεχνητής Νοημοσύνης για παραγωγή κειμένου και πηγαίου κώδικα έχει δημιουργήσει νέα δεδομένα στον χώρο της εκπαίδευσης [9, 10]. Οι φοιτητές έχουν πλέον πρόσβαση σε εργαλεία που μπορούν να παράγουν απαντήσεις, αναλύσεις, κώδικα και τεχνικά κείμενα με μεγάλη ευκολία και ταχύτητα. Τα εργαλεία αυτά μπορούν να βοηθήσουν στη μαθησιακή διαδικασία, αλλά ταυτόχρονα δημιουργούν προβλήματα σε θέματα ακαδημαϊκής ακεραιότητας και αυθεντικότητας των εργασιών.

Στα περισσότερα πανεπιστήμια ο έλεγχος των παραδοτέων είναι κατακερματισμένος. Η διαχείριση των υποβολών γίνεται συχνά με αποσπασματικό τρόπο και η αξιολόγηση για πιθανό AI-generated περιεχόμενο βασίζεται είτε σε χειροκίνητη παρατήρηση είτε σε εργαλεία που δεν είναι σχεδιασμένα για ακαδημαϊκή χρήση. Επίσης, τα διαθέσιμα εργαλεία συνήθως ασχολούνται μόνο με κείμενο, χωρίς να καλύπτουν τα διαφορετικά είδη παραδοτέων που συναντάμε στην πράξη [3, 10].

Έτσι, προκύπτει ένα σαφές κενό. Η απουσία ενός ολοκληρωμένου συστήματος που να συνδυάζει τη διαχείριση των εργασιών με την ανάλυση περιεχομένου σε ένα ενιαίο περιβάλλον. Οι καθηγητές καλούνται να διαχειριστούν μεγάλο όγκο αρχείων και να λάβουν αποφάσεις χωρίς πάντα να έχουν στη διάθεσή τους τα κατάλληλα εργαλεία. Το κίνητρο για την παρούσα εργασία είναι η ανάπτυξη ενός συστήματος που να ενοποιεί τις παραπάνω λειτουργίες σε ένα ενιαίο σύστημα. Με τον τρόπο αυτό, η διαδικασία διαχείρισης και αξιολόγησης των εργασιών γίνεται πιο οργανωμένη, πιο διαφανής και καλύτερα προσαρμοσμένη στις σύγχρονες ανάγκες της εκπαίδευσης.

1.3 Συνεισφορά της εργασίας

Η παρούσα διπλωματική συνεισφέρει στον σχεδιασμό και την υλοποίηση ενός ολοκληρωμένου συστήματος για τη διαχείριση και ανάλυση ακαδημαϊκών εργασιών, το οποίο ανταποκρίνεται στις σύγχρονες προκλήσεις που προκύπτουν από τη χρήση εργαλείων

Τεχνητής Νοημοσύνης στην εκπαίδευση. Οι κύριες συνεισφορές της εργασίας συνοψίζονται ως εξής:

- Σχεδιασμός και υλοποίηση ενός πληροφοριακού συστήματος που συνδυάζει τη διαχείριση μαθημάτων, παραδοτέων και ομάδων με τη δυνατότητα αυτόματης ανάλυσης περιεχομένου.
- Ανάπτυξη αρχιτεκτονικής client-server βασισμένης σε σύγχρονες τεχνολογίες (FastAPI [5] και Next.js [6]), σχεδιασμένη ώστε να μπορεί να επεκταθεί και να εξελιχθεί στο μέλλον.
- Υλοποίηση μηχανισμών ανάλυσης που αντιμετωπίζουν διαφορετικά τα αρχεία κειμένου και τα αρχεία πηγαίου κώδικα, με τη μέθοδο που ταιριάζει σε κάθε περίπτωση.
- Ενσωμάτωση του DetectCodeGPT για την ανίχνευση AI-generated κώδικα, μιας μεθόδου βασισμένης σε perturbation-based detection [3], σε συνδυασμό με tree-sitter parsers που υποστηρίζουν πολλαπλές γλώσσες προγραμματισμού.
- Σχεδίαση μηχανισμού παρουσίασης αποτελεσμάτων με τη μορφή ποσοστών και βοηθητικών πληροφοριών, με στόχο την υποστήριξη της διαδικασίας αξιολόγησης χωρίς την αντικατάσταση της ανθρώπινης κρίσης [11, 12].

Συνολικά, η εργασία προσφέρει μια πρακτική και ολοκληρωμένη λύση για την υποστήριξη της αξιολόγησης των εργασιών των φοιτητών.

1.4 Μεθοδολογία Ανάπτυξης και Χρονοδιαγράμματα

Η ανάπτυξη ενός πληροφοριακού συστήματος απαιτεί μια οργανωμένη διαδικασία που καλύπτει όλα τα στάδια, από την κατανόηση του προβλήματος μέχρι την εγκατάσταση και τη συντήρηση. Στη βιβλιογραφία της Τεχνολογίας Λογισμικού περιγράφονται διάφορα μοντέλα κύκλου ζωής που καθορίζουν πώς οργανώνονται αυτά τα στάδια [1]. Τα πιο γνωστά είναι τα εξής:

Το μοντέλο καταρράκτη (waterfall) ακολουθεί μια αυστηρά γραμμική ροή: κάθε στάδιο ολοκληρώνεται πριν ξεκινήσει το επόμενο, χωρίς δυνατότητα επιστροφής. Αυτό λειτουργεί καλά όταν οι απαιτήσεις είναι πλήρως γνωστές εξ αρχής, αλλά στην πράξη σπάνια ισχύει αυτό.

Το αυξητικό μοντέλο (incremental) βασίζεται στη σταδιακή ανάπτυξη. Μια πρώτη λειτουργική έκδοση κυκλοφορεί νωρίς, και κάθε επόμενη έκδοση προσθέτει νέες λειτουργίες πάνω στην προηγούμενη. Με τον τρόπο αυτό, το σύστημα γίνεται χρησιμοποιήσιμο πολύ πριν ολοκληρωθεί πλήρως [1].

Το εξελικτικό μοντέλο (evolutionary) βασίζεται στην ιδέα ότι το σύστημα διαμορφώνεται σταδιακά μέσα από συνεχείς βελτιώσεις. Σε αντίθεση με πιο σταθερές προσεγγίσεις, οι απαιτήσεις δεν θεωρούνται πλήρως καθορισμένες από την αρχή. Καθώς οι χρήστες αλληλοεπιδρούν με το σύστημα, εντοπίζουν νέες ανάγκες, προτείνουν αλλαγές και επηρεάζουν την πορεία της ανάπτυξης. Έτσι, το τελικό σύστημα προσαρμόζεται καλύτερα στις πραγματικές ανάγκες των χρηστών [1].

Υπάρχουν και πιο σύγχρονες προσεγγίσεις, όπως οι ευέλικτες μεθοδολογίες (agile), που δίνουν έμφαση σε μικρούς κύκλους ανάπτυξης, συνεχή ανατροφοδότηση και στενή συνεργασία με τον τελικό χρήστη. Ωστόσο, αυτές εφαρμόζονται συνήθως σε ομάδες πολλών ατόμων και δεν ταίριαζαν στο πλαίσιο μιας ατομικής διπλωματικής εργασίας.

Για την παρούσα εργασία επιλέχθηκε ένας συνδυασμός αυξητικού και εξελικτικού μοντέλου. Η επιλογή αυτή έγινε για δύο λόγους. Πρώτον, το σύστημα αποτελείται από πολλά ανεξάρτητα υποσυστήματα (αυθεντικοποίηση, διαχείριση μαθημάτων, ανάλυση κειμένου, ανάλυση κώδικα, αναφορές, στατιστικά) που μπορούσαν να αναπτυχθούν σταδιακά. Δεύτερον, ορισμένες απαιτήσεις δεν ήταν γνωστές εξ αρχής. Για παράδειγμα, η ανάγκη για ανάλυση πηγαίου κώδικα προέκυψε αφού είχε ήδη ολοκληρωθεί η ανάλυση κειμένου, και η εγκατάσταση σε πραγματικό server ανέδειξε τεχνικές απαιτήσεις που δεν είχαν προβλεφθεί αρχικά.

Στην πράξη, η ανάπτυξη εξελίχθηκε ως εξής:

Στο πρώτο στάδιο μελετήθηκε το πρόβλημα και αξιολογήθηκε αν το σύστημα μπορεί να υλοποιηθεί με τις διαθέσιμες τεχνολογίες. Η μηχανική απαιτήσεων περιλαμβάνει την εύρεση, καταγραφή, ανάλυση και έλεγχο των απαιτήσεων ενός συστήματος [1]. Οι απαιτήσεις διαχωρίστηκαν σε λειτουργικές (οργάνωση παραδοτέων, υποβολή αρχείων, εκτέλεση ανάλυσης, προβολή αναφορών) και μη λειτουργικές (ασφάλεια, χρηστικότητα, αξιοπιστία, δυνατότητα λειτουργίας σε πραγματικό περιβάλλον).

Σημαντικό είναι ότι οι απαιτήσεις δεν παρέμειναν σταθερές. Καθώς προχωρούσε η υλοποίηση, προέκυψαν νέες ανάγκες: η υποστήριξη ανάλυσης πηγαίου κώδικα [3, 10], η οργάνωση δεδομένων σε ιεραρχική δομή (μαθήματα → ακαδημαϊκά εξάμηνα → παραδοτέα → ομάδες φοιτητών) και η ανάγκη εγκατάστασης σε server του Τμήματος. Αυτές οι αλλαγές ήταν αναμενόμενες στο πλαίσιο του εξελικτικού μοντέλου.

Μετά τον καθορισμό των απαιτήσεων, ακολούθησε ο σχεδιασμός του συστήματος. Σε αυτό το στάδιο καθορίστηκε η αρχιτεκτονική client-server, όπου το backend αναλαμβάνει τη λογική της εφαρμογής, τη βάση δεδομένων, τη διαχείριση των αρχείων και την ανάλυση των υποβολών, ενώ το frontend παρέχει την διεπαφή χρήστη. Η σχεδίαση έγινε με στόχο τον διαχωρισμό των βασικών υποσυστημάτων, ώστε η εφαρμογή να είναι πιο κατανοητή, επεκτάσιμη και συντηρήσιμη. Στο στάδιο αυτό καθορίστηκε και το μοντέλο δεδομένων του συστήματος. Τα ακαδημαϊκά δεδομένα οργανώθηκαν ιεραρχικά σε μαθήματα, ακαδημαϊκά εξάμηνα, παραδοτέα, και ομάδες. Η επιλογή αυτής της δομής έγινε ώστε το σύστημα να αντικατοπτρίζει τον πραγματικό τρόπο με τον οποίο οργανώνονται οι εργασίες σε ένα πανεπιστημιακό περιβάλλον. Παράλληλα, σχεδιάστηκαν οι βασικές οντότητες της βάσης δεδομένων, όπως χρήστες, μαθήματα, παραδοτέα, ομάδες και αναφορές ανάλυσης.

Ακολούθως ξεκίνησε η υλοποίηση της πρώτης λειτουργικής έκδοσης του συστήματος. Η πρώτη έκδοση περιλάμβανε τις βασικές λειτουργίες που ήταν απαραίτητες ώστε το σύστημα να αρχίσει να λειτουργά. Συγκεκριμένα, υλοποιήθηκαν οι μηχανισμοί αυθεντικοποίησης, η διαχείριση χρηστών, η δημιουργία μαθημάτων, η δημιουργία ακαδημαϊκών εξαμήνων,

παραδοτέων και ομάδων, καθώς και η βασική λειτουργία υποβολής αρχείων. Η υλοποίηση του backend πραγματοποιήθηκε με FastAPI, ενώ το frontend αναπτύχθηκε με Next.js.

Για τη διαχείριση της βάσης δεδομένων χρησιμοποιήθηκε SQLAlchemy [8] και Alembic. Αμέσως μετά, προστέθηκε η δυνατότητα ανάλυσης αρχείων κειμένου τύπου DOCX. Το σύστημα επεξεργάζεται το περιεχόμενο του αρχείου και παράγει αποτελέσματα σε επίπεδο παραγράφου, παρέχοντας ποσοστό πιθανής χρήσης AI, τελική κατηγοριοποίηση και αναλυτική παρουσίαση των αποτελεσμάτων. Η λειτουργία αυτή αποτέλεσε βασικό κομμάτι του συστήματος, καθώς συνδέεται άμεσα με τον αρχικό στόχο της εργασίας, δηλαδή, την υποβοήθηση της αξιολόγησης ακαδημαϊκών εργασιών που ενδέχεται να περιέχουν AI-generated περιεχόμενο.

Στην επόμενη φάση προστέθηκε η δυνατότητα ανάλυσης πηγαίου κώδικα. Η προσθήκη αυτή αποτέλεσε εξέλιξη των αρχικών απαιτήσεων, καθώς το σύστημα δεν περιορίστηκε πλέον μόνο σε ακαδημαϊκά κείμενα, αλλά επεκτάθηκε ώστε να μπορεί να αναλύει και υποβολές που περιέχουν κώδικα. Για τον σκοπό αυτό ενσωματώθηκαν μηχανισμοί ανίχνευσης που εξετάζουν χαρακτηριστικά του κώδικα και παράγουν ενδείξεις σχετικά με το κατά πόσο ο κώδικας μπορεί να έχει δημιουργηθεί από εργαλεία Τεχνητής Νοημοσύνης.

Παράλληλα με την υλοποίηση πραγματοποιήθηκε προγραμματισμός και παρακολούθηση του έργου. Στην Τεχνολογία Λογισμικού, ο προγραμματισμός έργου περιλαμβάνει τη διάσπαση της εργασίας σε δραστηριότητες, την ανάθεση εργασιών, την πρόβλεψη πιθανών προβλημάτων και την αξιολόγηση της προόδου. Στην παρούσα εργασία, η ανάπτυξη οργανώθηκε σε επιμέρους στάδια όπως την αρχική ανάλυση, τον σχεδιασμό βάσης δεδομένων, την υλοποίηση backend και frontend, την ενσωμάτωση ανάλυσης DOCX και ανάλυσης κώδικα, τη δημιουργία αναφορών, τις δοκιμές και την εγκατάσταση του συστήματος σε server του Τμήματος. Αυτά τα στάδια χρησιμοποιήθηκαν ως σημεία ελέγχου της προόδου, σύμφωνα με την πρακτική της διαχείρισης έργων λογισμικού.

Κατά τη διάρκεια της ανάπτυξης εφαρμόστηκαν διαδικασίες επαλήθευσης και επικύρωσης. Η επαλήθευση εξετάζει αν το σύστημα υλοποιείται σωστά σύμφωνα με τις προδιαγραφές,

ενώ η επικύρωση εξετάζει αν το σύστημα καλύπτει πραγματικά τις ανάγκες του χρήστη [1]. Στο πλαίσιο της παρούσας εργασίας, η επαλήθευση εφαρμόστηκε μέσω ελέγχων στα API endpoints, στη σωστή αποθήκευση δεδομένων, στη διαχείριση αρχείων και στη λειτουργία των μοντέλων ανάλυσης. Η επικύρωση εφαρμόστηκε ελέγχοντας αν οι τελικές λειτουργίες εξυπηρετούν τον σκοπό του συστήματος, δηλαδή αν βοηθούν τον καθηγητή να οργανώνει υποβολές και να λαμβάνει αναλυτικές ενδείξεις για πιθανή χρήση Τεχνητής Νοημοσύνης.

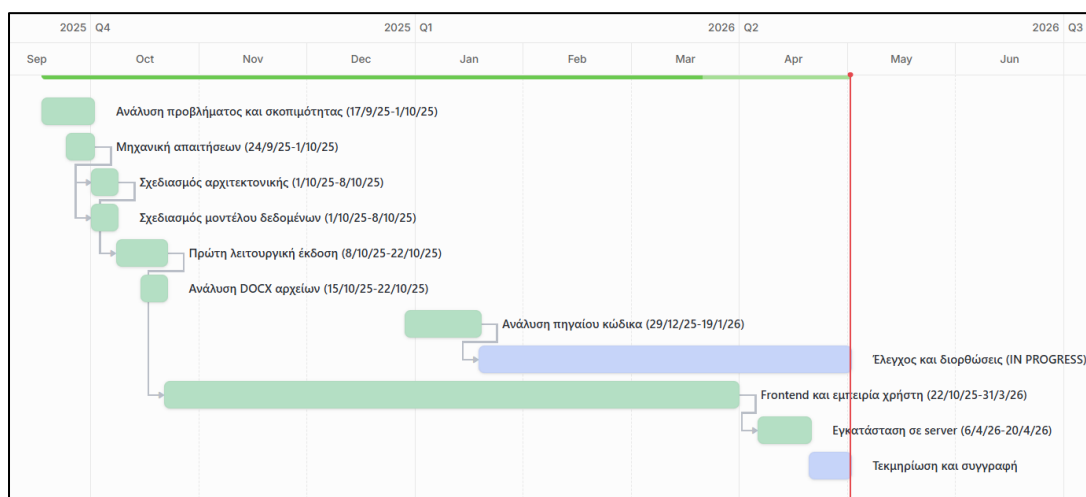
Σημαντικό μέρος της μεθοδολογίας αποτέλεσαν και οι δοκιμές λογισμικού. Ο έλεγχος λογισμικού έχει ως σκοπό να δείξει ότι το πρόγραμμα έχει την αναμενόμενη συμπεριφορά και να ανακαλύψει σφάλματα πριν τη χρήση του από τελικούς χρήστες [1]. Στο σύστημα πραγματοποιήθηκαν δοκιμές για τη σύνδεση χρηστών, την πρόσβαση με βάση ρόλους [13], τη δημιουργία και επεξεργασία μαθημάτων, την υποβολή αρχείων, την εκτέλεση ανάλυσης, την εμφάνιση αναφορών και τη σωστή συμπεριφορά του συστήματος σε περιπτώσεις λανθασμένων ή μη υποστηριζόμενων αρχείων. Οι δοκιμές αυτές βοήθησαν στον εντοπισμό και τη διόρθωση σφαλμάτων τόσο στο backend όσο και στο frontend.

Στο τελικό στάδιο πραγματοποιήθηκε η εγκατάσταση του συστήματος σε πραγματικό περιβάλλον. Το σύστημα μεταφέρθηκε σε server Rocky Linux και ρυθμίστηκε ώστε να λειτουργεί μέσω Apache, HTTPS και systemd services. Η φάση αυτή αποτέλεσε σημαντικό μέρος της εργασίας, επειδή το σύστημα δεν παρέμεινε σε τοπικό περιβάλλον ανάπτυξης, αλλά εγκαταστάθηκε σε πραγματικό περιβάλλον. Η εγκατάσταση αυτή απαίτησε επιπλέον ελέγχους σε θέματα διαδρομών αρχείων, μεταβλητών περιβάλλοντος, εκκίνησης υπηρεσιών και επικοινωνίας frontend με backend.

Τέλος, λήφθηκε υπόψη και η έννοια της συντήρησης και εξέλιξης λογισμικού. Η συντήρηση αφορά τη διαδικασία τροποποίησης ενός συστήματος μετά την παράδοσή του, με στόχο τη διόρθωση σφαλμάτων, τη βελτίωση απόδοσης και την προσαρμογή σε νέες απαιτήσεις [1]. Για τον λόγο αυτό, το σύστημα σχεδιάστηκε με τρόπο που να επιτρέπει μελλοντικές επεκτάσεις, όπως υποστήριξη περισσότερων τύπων αρχείων, βελτίωση των μοντέλων ανίχνευσης και προσθήκη νέων αναλυτικών στατιστικών.

Συνολικά, η μεθοδολογία ανάπτυξης της διπλωματικής εργασίας βασίστηκε σε μια οργανωμένη διαδικασία λογισμικού, η οποία ξεκίνησε από την ανάλυση του προβλήματος και των απαιτήσεων, προχώρησε στον σχεδιασμό και την υλοποίηση του συστήματος, συνέχισε με δοκιμές και επαλήθευση, και ολοκληρώθηκε με την εγκατάσταση σε πραγματικό server. Η προσέγγιση αυτή επέτρεψε την ανάπτυξη ενός λειτουργικού, επεκτάσιμου και πρακτικά αξιοποιήσιμου συστήματος για τη διαχείριση και ανάλυση εργασιών.

Το συνολικό χρονοδιάγραμμα της διπλωματικής εργασίας παρατίθεται στο Σχήμα 1.



Σχήμα 1 - Χρονοδιάγραμμα Υλοποίησης Εργασίας

1.5 Οργάνωση Κειμένου

Η παρούσα διπλωματική εργασία είναι οργανωμένη σε 7 κεφάλαια και τέσσερα παραρτήματα, με στόχο την αναλυτική παρουσίαση του προβλήματος, του θεωρητικού υποβάθρου, της σχεδίασης, της υλοποίησης και της αξιολόγησης του συστήματος.

Στο Κεφάλαιο 1 παρουσιάζεται η εισαγωγή της εργασίας. Περιγράφεται ο στόχος της διπλωματικής, το πρόβλημα που επιχειρεί να αντιμετωπίσει, το κίνητρο για την ανάπτυξη του συστήματος και η βασική συνεισφορά της εργασίας. Επιπλέον, παρουσιάζεται η

μεθοδολογία ανάπτυξης, το χρονοδιάγραμμα και οι βασικοί όροι που χρησιμοποιούνται στο κείμενο.

Στο Κεφάλαιο 2 παρουσιάζεται το θεωρητικό υπόβαθρο. Αναλύεται η έννοια της Τεχνητής Νοημοσύνης καθώς και η διάκριση ανάμεσα σε AI-generated κείμενο και AI-generated κώδικα. Επίσης, γίνεται αναφορά στις βασικές προσεγγίσεις εντοπισμού τέτοιου περιεχομένου και στη σημασία των συστημάτων διαχείρισης ακαδημαϊκών εργασιών.

Στο Κεφάλαιο 3 παρουσιάζονται οι απαιτήσεις του συστήματος. Αναλύονται οι λειτουργικές και μη λειτουργικές απαιτήσεις, οι κατηγορίες χρηστών, οι περιορισμοί του συστήματος και τα βασικά σενάρια χρήσης.

Στο Κεφάλαιο 4 παρουσιάζεται η σχεδίαση και η υλοποίηση του συστήματος. Περιγράφεται η αρχιτεκτονική client-server, τα εργαλεία ανάπτυξης που χρησιμοποιήθηκαν, οι μηχανισμοί ασφάλειας και η δομή της βάσης δεδομένων. Στο ίδιο κεφάλαιο παρουσιάζεται η δομή των αρχείων του project, οι βιβλιοθήκες που χρησιμοποιήθηκαν και αντιπροσωπευτικά παραδείγματα κώδικα τόσο από το backend όσο και από το frontend.

Στο Κεφάλαιο 5 παρουσιάζεται ο έλεγχος της λειτουργικότητας του συστήματος. Περιγράφεται η μεθοδολογία ελέγχου, οι δοκιμές που πραγματοποιήθηκαν για κάθε βασική λειτουργία και τα αποτελέσματα που προέκυψαν.

Στο Κεφάλαιο 6 παρουσιάζεται η συζήτηση πάνω στα ευρήματα της εργασίας και οι μελλοντικές προεκτάσεις. Αναφέρονται οι περιορισμοί του συστήματος, τονίζοντας ότι τα αποτελέσματα ανίχνευσης λειτουργούν ως βοηθητικές ενδείξεις και όχι ως απόλυτες αποφάσεις, και προτείνονται πιθανές βελτιώσεις.

Τέλος, στο Κεφάλαιο 7 παρουσιάζονται τα συμπεράσματα της εργασίας. Συνοψίζεται η συνολική συνεισφορά και καταγράφονται οι εμπειρίες που αποκτήθηκαν κατά τη διάρκεια της ανάπτυξης.

Στο Παράρτημα Α περιλαμβάνεται το ερωτηματολόγιο που σχεδιάστηκε για την αξιολόγηση του συστήματος.

Στο Παράρτημα Β περιλαμβάνεται το Εγχειρίδιο Χρήστη. Περιγράφονται αναλυτικά οι βασικές οθόνες της εφαρμογής, όπως η σύνδεση και η εγγραφή, η διαχείριση μαθημάτων και παραδοτέων, η υποβολή αρχείων, η προβολή αποτελεσμάτων ανάλυσης και το περιβάλλον διαχείρισης. Η παρουσίαση συνοδεύεται από στιγμιότυπα οθόνης, ώστε να γίνεται κατανοητός ο τρόπος λειτουργίας της πλατφόρμας στην πράξη.

Στο Παράρτημα Γ περιλαμβάνεται το Τεχνικό Εγχειρίδιο, με οδηγίες εγκατάστασης και ρύθμισης του συστήματος σε server.

Στο Παράρτημα Δ παρουσιάζονται παραδείγματα υλοποίησης. Περιλαμβάνονται αντιπροσωπευτικά αποσπάσματα κώδικα από βασικά σημεία του συστήματος, τόσο από το backend όσο και από το frontend, που αναδεικνύουν τον τρόπο λειτουργίας και συνεργασίας των επιμέρους υποσυστημάτων.

1.6 Ορισμοί, Ακρωνύμια και Συντομογραφίες

Ορισμοί, Ακρωνύμια και Συντομογραφίες	Επεξήγηση
ΠΚ	Πανεπιστήμιο Κύπρου.
ΑΙ	Artificial Intelligence, Τεχνητή Νοημοσύνη.
AI-generated content	Περιεχόμενο που έχει παραχθεί πλήρως ή μερικώς από εργαλεία Τεχνητής Νοημοσύνης.
AI-generated text	Κείμενο που έχει παραχθεί από εργαλείο Τεχνητής Νοημοσύνης.
AI-generated code	Πηγαίος κώδικας που έχει παραχθεί από εργαλείο Τεχνητής Νοημοσύνης.
Χρήστης	Οποιοσδήποτε έχει λογαριασμό και χρησιμοποιεί το σύστημα για παρακολούθηση και διαχείριση διαδικασιών.
Καθηγητής	Χρήστης του συστήματος που έχει τη δυνατότητα να διαχειρίζεται μαθήματα, παραδοτέα, ομάδες και υποβολές εργασιών.
Διαχειριστής	Χρήστης με αυξημένα δικαιώματα, ο οποίος μπορεί να εγκρίνει λογαριασμούς και να διαχειρίζεται βασικές ρυθμίσεις του συστήματος.
Υποβολή	Αρχείο ή σύνολο αρχείων που ανεβαίνει στο σύστημα και συνδέεται με συγκεκριμένο παραδοτέο ή ομάδα.
Παραδοτέο	Εργασία ή μέρος εργασίας που ανήκει σε συγκεκριμένο μάθημα και ακαδημαϊκό εξάμηνο.
Backend	Το τμήμα του συστήματος που εκτελείται στον server και διαχειρίζεται τη λογική, τα δεδομένα, τα αρχεία και τα API endpoints.

Frontend	Το τμήμα του συστήματος που εμφανίζεται στον χρήστη μέσω του φυλλομετρητή και επιτρέπει την αλληλεπίδραση με την εφαρμογή.
API	Application Programming Interface. Μηχανισμός επικοινωνίας μεταξύ διαφορετικών τμημάτων λογισμικού.
REST API	Τύπος API που βασίζεται σε HTTP αιτήματα για την ανταλλαγή δεδομένων μεταξύ client και server.
FastAPI	Framework της Python που χρησιμοποιήθηκε για την ανάπτυξη του backend.
Next.js	Framework βασισμένο στην React, το οποίο χρησιμοποιήθηκε για την ανάπτυξη του frontend.
SQLAlchemy	Βιβλιοθήκη της Python που χρησιμοποιείται για την επικοινωνία του backend με τη βάση δεδομένων.
Alembic	Εργαλείο που χρησιμοποιείται για τη διαχείριση αλλαγών στο σχήμα της βάσης δεδομένων.
MariaDB	Σύστημα διαχείρισης σχεσιακής βάσης δεδομένων που χρησιμοποιείται στο production περιβάλλον.
JWT	JSON Web Token. Μηχανισμός αυθεντικοποίησης που χρησιμοποιείται για την ασφαλή αναγνώριση χρηστών [13].
OTP	One-Time Password. Κωδικός μίας χρήσης που χρησιμοποιείται για επιβεβαίωση email.
DetectCodeGPT	Μέθοδος ανίχνευσης πιθανώς AI-generated πηγαίου κώδικα, βασισμένη σε perturbation-based detection.
tree-sitter	Εργαλείο parsing πηγαίου κώδικα που χρησιμοποιείται για την ανάλυση της δομής προγραμμάτων σε διάφορες γλώσσες.
Apache	Web server που χρησιμοποιείται ως reverse proxy για την πρόσβαση στο σύστημα [15].
Reverse Proxy	Μηχανισμός μέσω του οποίου ένας web server προωθεί αιτήματα προς εσωτερικές υπηρεσίες.

systemd	Σύστημα διαχείρισης υπηρεσιών σε Linux, το οποίο χρησιμοποιείται για αυτόματη εκκίνηση και παρακολούθηση υπηρεσιών.
React.js	Βιβλιοθήκη JavaScript για την κατασκευή διαδραστικών διεπαφών χρήστη (user interfaces).
phpMyAdmin	Διεπαφή διαχείρισης βάσεων δεδομένων MySQL μέσω web περιβάλλοντος.
Components	Επαναχρησιμοποιήσιμα κομμάτια διεπαφής χρήστη στο React, που υλοποιούν λειτουργικές μονάδες του frontend.
UI	Διεπαφή χρήστη (User Interface)
UX	Εμπειρία χρήστη (User Experience)

Κεφάλαιο 2

2. Υπόβαθρο και Σχετικές Εργασίες

2.1 Τεχνητή Νοημοσύνη και παραγωγή περιεχομένου	15
2.2 AI-generated κείμενο και AI-generated κώδικας	16
2.3 Ανάλυση και εντοπισμός AI-generated περιεχομένου	17
2.4 Συστήματα διαχείρισης ακαδημαϊκών υποβολών	18
2.5 Σχετικές Εφαρμογές και Εργαλεία	19

2.1 Τεχνητή Νοημοσύνη και παραγωγή περιεχομένου

Τα τελευταία χρόνια, η Τεχνητή Νοημοσύνη έχει παρουσιάσει ραγδαία εξέλιξη [9, 10], ιδιαίτερα στον τομέα των συστημάτων που είναι ικανά να παράγουν περιεχόμενο με μορφή φυσικής γλώσσας, τεχνικού κειμένου ή ακόμη και πηγαίου κώδικα. Η ανάπτυξη μεγάλων γλωσσικών μοντέλων και εργαλείων παραγωγής περιεχομένου έχει επιτρέψει την αυτόματη δημιουργία απαντήσεων, αναλύσεων, περιγραφών, προγραμμάτων και άλλων μορφών πολύ γρήγορα και εύκολα.

Η δυνατότητα αυτή έχει επηρεάσει σημαντικά πολλούς τομείς, όπως την εκπαίδευση, την ανάπτυξη λογισμικού και τη διαχείριση πληροφοριών. Ειδικότερα στο ακαδημαϊκό περιβάλλον, η χρήση τέτοιων εργαλείων μπορεί να λειτουργήσει υποστηρικτικά για τον φοιτητή, βοηθώντας τον στην κατανόηση δύσκολων εννοιών, στη δημιουργία ενός αρχικού προσχεδίου, στη βελτίωση της δομής ενός κειμένου και στη διερεύνηση πιθανών λύσεων σε τεχνικά ή προγραμματιστικά προβλήματα. Παράλληλα όμως δημιουργεί και νέα ερωτήματα σχετικά με την αυθεντικότητα, τη διαφάνεια και τα όρια της χρήσης αυτών των τεχνολογιών [9, 12].

Η πρόκληση γίνεται ακόμη μεγαλύτερη όταν το παραγόμενο περιεχόμενο είναι αρκετά κοντά σε ανθρώπινη εργασία. Σε τέτοιες περιπτώσεις, η απλή οπτική επιθεώρηση από έναν καθηγητή δεν είναι πάντοτε αρκετή, ιδιαίτερα όταν υπάρχει μεγάλος αριθμός υποβολών ή όταν τα παραδοτέα περιλαμβάνουν διαφορετικούς τύπους αρχείων, όπως κείμενα και πηγαίο κώδικα. Για τον λόγο αυτό, δημιουργείται η ανάγκη για εργαλεία που μπορούν να παρέχουν βοηθητικές ενδείξεις σχετικά με την πιθανή χρήση Τεχνητής Νοημοσύνης στις εργασίες των φοιτητών.

2.2 AI-generated κείμενο και AI-generated κώδικας

Το AI-generated περιεχόμενο μπορεί να εμφανίζεται σε διαφορετικές μορφές, με πιο συχνές το κείμενο και τον πηγαίο κώδικα [3, 10]. Στην πρώτη περίπτωση, τα εργαλεία Τεχνητής Νοημοσύνης μπορούν να παράγουν απαντήσεις σε ερωτήσεις, περιγραφές και γενικότερα γραπτό κείμενο που ακολουθεί συγκεκριμένο ύφος και δομή. Στη δεύτερη περίπτωση, μπορούν να δημιουργούν συναρτήσεις, κλάσεις, βοηθητικές μεθόδους ή ακόμη και ολοκληρωμένα προγράμματα.

Παρότι το AI-generated κείμενο και ο AI-generated κώδικας έχουν κοινό χαρακτηριστικό την αυτόματη παραγωγή τους από κάποιο μοντέλο, παρουσιάζουν και ουσιαστικές διαφορές. Το κείμενο αξιολογείται συνήθως με βάση τη συνοχή, τη δομή, την ομοιομορφία και την οργάνωση των παραγράφων. Αντίθετα, ο πηγαίος κώδικας έχει επιπλέον ιδιότητες, όπως η μορφοποίηση, τα ονόματα μεταβλητών και συναρτήσεων, ο τρόπος διάσπασης του προβλήματος σε μικρότερες λειτουργίες, η χρήση σχολίων, η επαναχρησιμοποίηση μοτίβων και η γενικότερη δομή του προγράμματος [3].

Η ύπαρξη αυτών των δύο μορφών περιεχομένου είναι ιδιαίτερα σημαντική, επειδή πολλά παραδοτέα δεν περιορίζονται μόνο σε ένα έγγραφο κειμένου, αλλά μπορεί να περιλαμβάνουν και αρχεία κώδικα. Επομένως, ένα σύστημα που στοχεύει στην ανάλυση ακαδημαϊκών υποβολών πρέπει να μπορεί να χειριστεί διαφορετικούς τύπους αρχείων και να εφαρμόζει κατάλληλη μέθοδο αξιολόγησης ανάλογα με το είδος του.

2.3 Ανάλυση και εντοπισμός AI-generated περιεχομένου

Ο εντοπισμός περιεχομένου που έχει παραχθεί από εργαλεία Τεχνητής Νοημοσύνης αποτελεί μια ιδιαίτερα δύσκολη διαδικασία, καθώς δεν υπάρχει απόλυτα αξιόπιστη μέθοδος που να μπορεί να αποδείξει με βεβαιότητα την προέλευση ενός κειμένου ή ενός κώδικα [10, 11, 12]. Για τον λόγο αυτό, οι περισσότερες προσεγγίσεις δεν βασίζονται σε μία μοναδική ένδειξη, αλλά σε συνδυασμό χαρακτηριστικών, μοτίβων και στατιστικών παρατηρήσεων, από τα οποία προκύπτει μια εκτίμηση [2].

Στην περίπτωση του κειμένου, συχνά εξετάζονται στοιχεία όπως η ομοιομορφία του ύφους, η δομή των παραγράφων, η σταθερότητα της γλώσσας, η επανάληψη συγκεκριμένων εκφράσεων και η συνολική ροή του κειμένου [3]. Ωστόσο, τέτοια στοιχεία δεν αποτελούν από μόνα τους απόδειξη, καθώς μπορούν να εμφανιστούν και σε ανθρώπινα γραπτά.

Στην περίπτωση του πηγαίου κώδικα, η ανάλυση μπορεί να βασιστεί σε διαφορετικά χαρακτηριστικά. Εξετάζονται στοιχεία όπως η μορφοποίηση, τα ονόματα μεταβλητών και συναρτήσεων, η χρήση γενικών σχολίων, η ομοιομορφία στον τρόπο γραφής και η επαναλαμβανόμενη χρήση βοηθητικών συναρτήσεων. Παράλληλα, ο κώδικας μπορεί να αναλυθεί και ως προς τη δομή του, αφού η οργάνωση των συναρτήσεων, των κλάσεων και των επιμέρους τμημάτων του προγράμματος μπορεί να δώσει επιπλέον ενδείξεις [3].

Ιδιαίτερα σημαντικό στην υλοποίηση τέτοιων συστημάτων είναι η επεξήγηση των αποτελεσμάτων. Οι καθηγητές χρειάζεται να έχουν πρόσβαση σε πληροφορίες που εξηγούν γιατί ένα αρχείο παρουσιάζει ενδείξεις πιθανής χρήσης Τεχνητής Νοημοσύνης. Για τον λόγο αυτό, τα αποτελέσματα είναι προτιμότερο να παρουσιάζονται με ποσοστά, σχόλια και αναλυτικές πληροφορίες, αντί να περιορίζονται σε ένα τελικό αποτέλεσμα, όπως «AI» ή «Human».

Επιπλέον, είναι σημαντικό να ξεκαθαριστεί ότι τα συστήματα ανίχνευσης AI-generated περιεχομένου δεν μπορούν να δώσουν απόλυτες απαντήσεις. Τα αποτελέσματά τους πρέπει να αντιμετωπίζονται ως ενδείξεις και όχι ως τελικές αποδείξεις [11, 12]. Αυτό είναι ιδιαίτερα

σημαντικό, γιατί μια λανθασμένη εκτίμηση μπορεί να αδικήσει έναν φοιτητή. Για τον λόγο αυτό, τέτοια συστήματα πρέπει να χρησιμοποιούνται ως βοηθητικά εργαλεία για τον καθηγητή, προσφέροντας επιπλέον πληροφορίες που μπορούν να τον βοηθήσουν να σχηματίσει μια πιο ολοκληρωμένη και δίκαιη αξιολόγηση.

2.4 Συστήματα διαχείρισης ακαδημαϊκών υποβολών

Η διαχείριση των εργασιών είναι πολύ σημαντική λειτουργία σε πολλά πανεπιστημιακά και εκπαιδευτικά περιβάλλοντα. Συνήθως περιλαμβάνει τη δημιουργία μαθημάτων, τον ορισμό παραδοτέων, τη διαχείριση ομάδων ή μεμονωμένων φοιτητών, την καταχώριση υποβολών και την αποθήκευση αρχείων. Τέτοιες λειτουργίες βοηθούν τους καθηγητές να οργανώνουν καλύτερα τις εργασίες και να έχουν συγκεντρωμένα τα παραδοτέα των φοιτητών.

Η εμφάνιση των εργαλείων Τεχνητής Νοημοσύνης αυξάνει την ανάγκη για συστήματα που δεν περιορίζονται μόνο στην αποθήκευση των αρχείων, αλλά παρέχουν και επιπλέον υποστήριξη στον διδάσκοντα κατά τη διαδικασία αξιολόγησης [9, 10]. Ένα τέτοιο σύστημα πρέπει να μπορεί να οργανώνει τα δεδομένα ανά μάθημα, ακαδημαϊκό εξάμηνο, παραδοτέο και ομάδα και ταυτόχρονα να μπορεί να αναλύσει το περιεχόμενο των εργασιών.

Επομένως, ένα σύγχρονο σύστημα διαχείρισης εργασιών μπορεί να επεκταθεί πέρα από την απλή αποθήκευση των αρχείων. Μπορεί να συνδυάζει λειτουργίες οργάνωσης, ανάλυσης και παρουσίασης αποτελεσμάτων, με στόχο να υποστηρίξει τον καθηγητή στη διαχείριση μεγάλου αριθμού εργασιών και στη λήψη πιο τεκμηριωμένων αποφάσεων. Σε αυτό το πλαίσιο, ο εντοπισμός AI-generated περιεχομένου αποτελεί μια πρόσθετη λειτουργία που μπορεί να ενισχύσει τη διαδικασία αξιολόγησης, χωρίς όμως να αντικαθιστά την ανθρώπινη κρίση [9, 11, 12].

2.5 Σχετικές Εφαρμογές και Εργαλεία

Πριν από τον σχεδιασμό του συστήματος, κρίθηκε απαραίτητη η μελέτη των υπαρχόντων εργαλείων και εφαρμογών που σχετίζονται με τον εντοπισμό AI-generated περιεχομένου και τη διαχείριση των εργασιών των φοιτητών. Σκοπός ήταν να κατανοηθεί τι υπάρχει ήδη στον χώρο, ποιες ανάγκες καλύπτονται και ποιες όχι, ώστε το σύστημα που θα αναπτυχθεί να καλύπτει ένα πραγματικό κενό.

- Εργαλεία ανίχνευσης AI-generated κειμένου:

Το πιο γνωστό εργαλείο στον χώρο της ακαδημαϊκής ακεραιότητας είναι το Turnitin, το οποίο χρησιμοποιείται εδώ και χρόνια από πανεπιστήμια για ανίχνευση λογοκλοπής. Από το 2023 πρόσθεσε δυνατότητα εντοπισμού AI-generated κειμένου, βασισμένη σε δικό του μοντέλο με αρχιτεκτονική transformer. Η ανίχνευση υποστηρίζει κυρίως αγγλικά, ισπανικά και ιαπωνικά, ενώ η πλατφόρμα ενσωματώνεται με δημοφιλή LMS όπως το Moodle, το Canvas και το Blackboard. Παρά την ευρεία χρήση του, το Turnitin δεν αναλύει πηγαίο κώδικα και η πρόσβαση γίνεται μόνο μέσω άδειας.

Ένα ακόμα εργαλείο που έχει αποκτήσει μεγάλη δημοτικότητα είναι το GPTZero, το οποίο δημιουργήθηκε από φοιτητή του Princeton και πλέον χρησιμοποιείται σε περισσότερα από 3.500 εκπαιδευτικά ιδρύματα. Η λογική του βασίζεται σε δύο μετρικές: την perplexity, η οποία μετράει πόσο απρόβλεπτο είναι ένα κείμενο για ένα γλωσσικό μοντέλο, και τη burstiness, που εξετάζει κατά πόσο υπάρχει ποικιλία στο μήκος και στη δομή των προτάσεων. Το GPTZero παρέχει χρωματική σήμανση σε επίπεδο πρότασης, ενσωμάτωση με Canvas και Google Classroom, και δυνατότητα μαζικού ελέγχου εργασιών. Ωστόσο, η υποστήριξή του αφορά κυρίως αγγλικό κείμενο, δεν αναλύει κώδικα και δεν μπορεί να εγκατασταθεί τοπικά.

Η πλατφόρμα Copyleaks ακολουθεί μια πιο ολοκληρωμένη προσέγγιση, συνδυάζοντας ανίχνευση λογοκλοπής και AI-generated περιεχομένου στο ίδιο εργαλείο. Υποστηρίζει πάνω από 30 γλώσσες και προσφέρει ανάλυση σε επίπεδο πρότασης. Το ιδιαίτερο χαρακτηριστικό

της Copyleaks είναι ότι μέσω του εργαλείου αυτού μπορεί να εντοπίσει AI-generated πηγαίο κώδικα, αν και αυτή η λειτουργία στοχεύει περισσότερο σε εταιρείες παρά σε πανεπιστήμια.

Τέλος, αξίζει να αναφερθεί και το Originality.ai, ένα εργαλείο που απευθύνεται κυρίως σε εκδότες και δημιουργούς περιεχομένου, αλλά έχει αρχίσει να χρησιμοποιείται και σε πανεπιστήμια. Σε ανεξάρτητες μελέτες σύγκρισης έχει δείξει υψηλή ακρίβεια, ιδιαίτερα σε κείμενο που έχει υποστεί παράφραση.

- Ερευνητικές προσεγγίσεις:

Εκτός από τα εμπορικά εργαλεία, στη βιβλιογραφία υπάρχουν και ερευνητικές μέθοδοι που αξίζει να αναφερθούν, καθώς αποτελούν τη θεωρητική βάση πολλών συστημάτων ανίχνευσης. Το DetectGPT [2] είναι μια μέθοδος που λειτουργεί χωρίς να χρειάζεται εκπαίδευση σε δεδομένα. Η ιδέα του είναι σχετικά απλή: εφαρμόζει μικρές αλλαγές σε ένα κείμενο και παρατηρεί πώς αλλάζει η πιθανότητά του σύμφωνα με ένα γλωσσικό μοντέλο. Αν η πιθανότητα πέφτει με κάθε αλλαγή, αυτό σημαίνει ότι το κείμενο βρίσκεται σε ένα τοπικό μέγιστο, κάτι που είναι χαρακτηριστικό AI-generated κειμένου. Η ίδια αρχή, δηλαδή η perturbation-based detection, είναι αυτή που χρησιμοποιεί και το DetectCodeGPT [3] που ενσωματώθηκε στο παρόν σύστημα, αλλά προσαρμοσμένη για πηγαίο κώδικα.

Το GLTR (Giant Language Model Test Room) ήταν ένα από τα πρώτα εργαλεία οπτικοποίησης AI-generated κειμένου. Χρησιμοποιεί χρωματική κωδικοποίηση για να δείξει ποιες λέξεις ενός κειμένου ήταν πιθανές επιλογές για το μοντέλο GPT-2. Παρόλο που η ιδέα ήταν καινοτόμα για την εποχή του, το εργαλείο βασίζεται σε παλαιότερα μοντέλα και δεν ανταποκρίνεται πλέον στα νεότερα LLMs. Μια πιο σύγχρονη ερευνητική μέθοδος είναι το Binoculars, η οποία συγκρίνει τις εξόδους δύο γλωσσικών μοντέλων ταυτόχρονα. Αν και αποτελεσματική σε AI κείμενο, η ακρίβειά της πέφτει σημαντικά σε κείμενο που έχει τροποποιηθεί.

- Συστήματα διαχείρισης ακαδημαϊκών εργασιών:

Στον χώρο της διαχείρισης ακαδημαϊκών εργασιών, τα πιο διαδεδομένα εργαλεία είναι τα Learning Management Systems. Το Moodle χρησιμοποιείται και από το Πανεπιστήμιο Κύπρου για τη διαχείριση μαθημάτων, την υποβολή εργασιών και τη βαθμολόγηση. Αποτελεί μια πλατφόρμα ανοιχτού κώδικα με πολύ μεγάλη βάση χρηστών παγκοσμίως. Παρά τις εκτεταμένες λειτουργίες του στη διαχείριση ακαδημαϊκού περιεχομένου, το Moodle δεν περιλαμβάνει ενσωματωμένη δυνατότητα ανίχνευσης AI-generated περιεχομένου. Κάτι τέτοιο είναι εφικτό μόνο μέσω plugins ή ενσωμάτωσης με εξωτερικά εργαλεία, όπως το Turnitin.

Αντίστοιχα, το GitHub Classroom χρησιμοποιείται σε μαθήματα προγραμματισμού για τη διαχείριση repositories, τον αυτόματο έλεγχο κώδικα μέσω CI/CD pipelines και τη βαθμολόγηση. Ωστόσο, δεν προσφέρει κανέναν είδος ανάλυσης για πιθανή χρήση AI στον κώδικα που υποβάλλεται.

Αυτό που γίνεται αντιληπτό από τα υπάρχοντα εργαλεία ανίχνευσης και τα συστήματα διαχείρισης εργασιών είναι ότι το κάθε εργαλείο λειτουργάει από μόνο του ξεχωριστά. Δηλαδή, ένας καθηγητής που θέλει να ελέγξει αν μια εργασία περιέχει AI-generated περιεχόμενο πρέπει πρώτα να τη συλλέξει από κάποιο σύστημα (π.χ. Moodle), μετά να ανεβάσει τα αρχεία σε ξεχωριστό εργαλείο ανίχνευσης (π.χ. Turnitin ή GPTZero), και τέλος να συνδυάσει τα αποτελέσματα χειροκίνητα. Αυτή η διαδικασία είναι χρονοβόρα, ιδιαίτερα σε μαθήματα με μεγάλο αριθμό υποβολών.

Επιπλέον, σχεδόν κανένα εργαλείο δεν προσφέρει ανάλυση πηγαίου κώδικα σε ακαδημαϊκό πλαίσιο. Το σύστημα της παρούσας εργασίας σχεδιάστηκε ακριβώς για να καλύψει αυτά τα κενά. Συγκεκριμένα, ενοποιεί τη διαχείριση ακαδημαϊκών δεδομένων (μαθήματα, εξάμηνα, παραδοτέα, ομάδες) με τη δυνατότητα ανάλυσης τόσο κειμένου όσο και πηγαίου κώδικα σε πολλαπλές γλώσσες προγραμματισμού, μέσω του DetectCodeGPT [3] και tree-sitter parsers [4].

Παράλληλα, λειτουργεί ως self-hosted εφαρμογή στον server του Τμήματος, χωρίς εξάρτηση από εξωτερικές υπηρεσίες ή κόστος αδειών, κάτι που είναι σημαντικό για ένα τμήμα του πανεπιστημίου που θέλει να έχει πλήρη έλεγχο των δεδομένων του.

Βέβαια, πρέπει να τονιστεί ότι τα εμπορικά εργαλεία υπερτερούν σε πολλά σημεία. Τα μοντέλα τους εκπαιδεύονται συνεχώς σε τεράστια datasets, υποστηρίζουν πολλές γλώσσες, ενσωματώνονται εύκολα με υπάρχουσες πλατφόρμες και αναβαθμίζονται τακτικά για να αντιμετωπίζουν νεότερα μοντέλα AI.

Η σύγκριση αυτή δεν έχει σκοπό να δείξει ότι το παρόν σύστημα αντικαθιστά αυτά τα εργαλεία, αλλά ότι προσφέρει μια εναλλακτική λύση για μια συγκεκριμένη ανάγκη: ένα ενιαίο, ανεξάρτητο σύστημα που μπορεί να χρησιμοποιηθεί άμεσα σε πανεπιστημιακά ιδρύματα χωρίς εξωτερικές εξαρτήσεις.

Κεφάλαιο 3

3. Απαιτήσεις Συστήματος

3.1 Απαιτήσεις και Γενική Περιγραφή Συστήματος	23
3.2 Κατηγορίες Χρηστών	32
3.3 Περιορισμοί Συστήματος	33
3.4 Σενάρια Χρήσης	34

3.1 Απαιτήσεις και Γενική Περιγραφή Συστήματος

Ο καθορισμός των απαιτήσεων αποτελεί σημαντικό στάδιο σε κάθε έργο ανάπτυξης λογισμικού, καθώς καθορίζει τις λειτουργίες που πρέπει να παρέχει το σύστημα, τους χρήστες που θα το αξιοποιούν, καθώς και τους περιορισμούς που πρέπει να ληφθούν υπόψη κατά τον σχεδιασμό και την υλοποίηση. Στο πλαίσιο της παρούσας εργασίας, στόχος ήταν η ανάπτυξη ενός πληροφοριακού συστήματος για τη διαχείριση ακαδημαϊκών εργασιών και την ανάλυση του περιεχομένου τους ως προς πιθανή χρήση εργαλείων Τεχνητής Νοημοσύνης [9, 10].

Ο προσδιορισμός των απαιτήσεων έγινε μέσα από την ανάλυση του προβλήματος και τη συνεχή επικοινωνία με την επιβλέπουσα καθηγήτρια, κα. Ελένη Κωνσταντίνου. Σκοπός της διαδικασίας ήταν να καταγραφούν οι βασικές ανάγκες που προκύπτουν από τη διαχείριση ακαδημαϊκών υποβολών και από την αυξανόμενη χρήση εργαλείων Τεχνητής Νοημοσύνης στην εκπόνηση εργασιών. Μέσα από αυτή τη διαδικασία καθορίστηκαν οι βασικές λειτουργίες του συστήματος, όπως η οργάνωση μαθημάτων, ακαδημαϊκών εξαμήνων, παραδοτέων, ομάδων και υποβολών, καθώς και η ανάγκη για ανάλυση διαφορετικών τύπων αρχείων, όπως έγγραφα κειμένου και πηγαίος κώδικας. Ιδιαίτερη έμφαση δόθηκε στη

δημιουργία ενός περιβάλλοντος απλού, κατανοητού και πρακτικού, το οποίο θα μπορεί να υποστηρίξει τον καθηγητή στη διαδικασία αξιολόγησης χωρίς να αντικαθιστά την κρίση του.

Το σύστημα απευθύνεται σε καθηγητές, οι οποίοι χρειάζονται έναν οργανωμένο τρόπο για να διαχειρίζονται μαθήματα, παραδοτέα, ομάδες και αρχεία φοιτητών. Παράλληλα, το σύστημα παρέχει μηχανισμούς ανάλυσης των αρχείων που υποβάλλονται, ώστε ο διδάσκων να μπορεί να λαμβάνει βοηθητικές ενδείξεις σχετικά με το αν ένα παραδοτέο περιέχει περιεχόμενο που ενδέχεται να έχει παραχθεί από εργαλεία ΑΙ.

Το τελικό σύστημα αποτελεί μια διαδικτυακή πλατφόρμα, προσβάσιμη μέσω browser, η οποία βασίζεται σε αρχιτεκτονική client-server. Το frontend παρέχει το γραφικό περιβάλλον χρήστη, ενώ το backend αναλαμβάνει τη διαχείριση των δεδομένων, την αυθεντικοποίηση χρηστών, την αποθήκευση αρχείων και την εκτέλεση των μηχανισμών ανάλυσης. Η βάση δεδομένων αποθηκεύει τις πληροφορίες που αφορούν χρήστες, μαθήματα, ακαδημαϊκά εξάμηνα, παραδοτέα, ομάδες, υποβολές και αναφορές ανάλυσης.

Το σύστημα υποστηρίζει δύο βασικές κατηγορίες χρηστών: τον διαχειριστή και τον καθηγητή. Ο διαχειριστής έχει τη δυνατότητα να ελέγχει και να εγκρίνει λογαριασμούς χρηστών, ενώ ο καθηγητής μπορεί να δημιουργεί και να διαχειρίζεται τη δική του περιβάλλον. Με αυτόν τον τρόπο, κάθε καθηγητής έχει πρόσβαση μόνο στα δικά του δεδομένα και στις δικά του μαθήματα.

Οι βασικές λειτουργίες του συστήματος περιλαμβάνουν:

- Διαχείριση χρηστών: Το σύστημα υποστηρίζει εγγραφή νέων χρηστών, επιβεβαίωση email μέσω κωδικού μιας χρήσης (OTP), έγκριση λογαριασμών από τον διαχειριστή, σύνδεση, αποσύνδεση και επαναφορά κωδικού πρόσβασης.
- Διαχείριση μαθημάτων: Ο καθηγητής μπορεί να δημιουργεί, να επεξεργάζεται και να διαγράφει μαθήματα.

- Διαχείριση ακαδημαϊκών εξαμήνων: Κάθε μάθημα μπορεί να συνδεθεί με διαφορετικά ακαδημαϊκά εξάμηνα, όπως για παράδειγμα Fall 2025 ή Spring 2026, ώστε οι υποβολές των εργασιών να οργανώνονται ανά περίοδο διδασκαλίας.
- Διαχείριση παραδοτέων: Ο καθηγητής μπορεί να δημιουργεί παραδοτέα για κάθε ακαδημαϊκό εξάμηνο. Κάθε παραδοτέο αντιστοιχεί σε μια εργασία ή σε μέρος μιας εργασίας που ανατίθεται στους φοιτητές.
- Διαχείριση ομάδων: Το σύστημα υποστηρίζει την οργάνωση των εργασιών σε ομάδες. Οι ομάδες μπορούν να δημιουργούνται αυτόματα κατά τη μαζική υποβολή αρχείων μέσω ZIP, με βάση τη δομή των φακέλων που περιέχονται στο αρχείο.
- Υποβολή αρχείων: Ο καθηγητής μπορεί να ανεβάζει αρχεία είτε μεμονωμένα είτε μαζικά μέσω ZIP. Στην περίπτωση του ZIP, κάθε φάκελος αντιστοιχεί σε μία ομάδα και τα αρχεία που περιέχει καταχωρούνται ως υποβολές της συγκεκριμένης ομάδας.
- Ανάλυση αρχείων: Το σύστημα μπορεί να αναλύει τα αρχεία που υποβάλλονται, είτε αυτόματα κατά την υποβολή είτε χειροκίνητα σε μεταγενέστερο στάδιο. Η διαδικασία ανάλυσης διαφοροποιείται ανάλογα με τον τύπο του αρχείου, δηλαδή εφαρμόζεται διαφορετική προσέγγιση για έγγραφα κειμένου και διαφορετική για πηγαίο κώδικα.
- Παραγωγή αναφορών: Για κάθε αρχείο που αναλύεται, δημιουργείται αναφορά με τα αποτελέσματα της ανάλυσης. Στα αρχεία DOCX η αναφορά περιλαμβάνει ανάλυση σε επίπεδο παραγράφου, ενώ στα αρχεία πηγαίου κώδικα παρουσιάζονται ενδείξεις και ποσοστά που σχετίζονται με πιθανή χρήση Τεχνητής Νοημοσύνης.
- Στατιστικά ανά παραδοτέο: Το σύστημα παρέχει αποτελέσματα για κάθε παραδοτέο, επιτρέποντας στον διδάσκοντα να αποκτά μια συνολική εικόνα για τις υποβολές που έχουν αναλυθεί.

Το σύστημα έχει σχεδιαστεί με τρόπο επεκτάσιμο, ώστε να μπορεί στο μέλλον να προσαρμοστεί σε νέες ανάγκες. Για παράδειγμα, μπορεί να επεκταθεί ώστε να υποστηρίζει περισσότερους τύπους αρχείων, νέες μεθόδους ανάλυσης και νέους τύπους χρηστών όπως τους φοιτητές, οι οποίοι θα μπορούν να παραδίδουν τις εργασίες τους μέσω αυτού του συστήματος. Ιδιαίτερη έμφαση δόθηκε στη λειτουργικότητα, στην ασφάλεια, στην απλότητα χρήσης και στην κατανοητή παρουσίαση των αποτελεσμάτων, έτσι ώστε το σύστημα να λειτουργεί ως βοηθητικό εργαλείο για τον καθηγητή και όχι ως μηχανισμός αυτόματης λήψης αποφάσεων.

3.1.1 Απαιτήσεις Διεπαφών

Το γραφικό περιβάλλον χρήστη του συστήματος σχεδιάστηκε με γνώμονα την απλότητα, την καθαρή οργάνωση της πληροφορίας και την εύκολη πρόσβαση στις βασικές λειτουργίες. Η εφαρμογή λειτουργεί μέσω browser και δεν απαιτείται η εγκατάσταση του λογισμικού από τον χρήστη. Ο καθηγητής μπορεί να έχει πρόσβαση στο σύστημα από σταθερό ή φορητό υπολογιστή, χρησιμοποιώντας έναν σύγχρονο browser. Για λόγους ασφάλειας, η πρόσβαση γίνεται μόνο μέσα από το δίκτυο του Τμήματος Πληροφορικής ή απομακρυσμένα μέσω VPN.

Η διεπαφή οργανώνεται με τρόπο που επιτρέπει στον χρήστη να ακολουθεί μια φυσική ροή. Αρχικά ο χρήστης συνδέεται στο σύστημα, στη συνέχεια δημιουργεί ή επιλέγει μάθημα, ορίζει ακαδημαϊκό εξάμηνο, δημιουργεί παραδοτέο, ανεβάζει αρχεία και τέλος βλέπει τα αποτελέσματα της ανάλυσης. Η δομή αυτή ακολουθεί τον τρόπο με τον οποίο οργανώνονται πρακτικά οι ακαδημαϊκές εργασίες.

Η επικοινωνία μεταξύ του frontend και του backend γίνεται μέσω REST API πάνω από HTTPS. Το frontend αποστέλλει αιτήματα προς το backend για ενέργειες όπως σύνδεση χρήστη, δημιουργία μαθημάτων, υποβολή αρχείων, ανάκτηση αναφορών και εμφάνιση στατιστικών. Το backend επιστρέφει τα αντίστοιχα δεδομένα σε κατάλληλη μορφή για προβολή στη διεπαφή χρήστη.

Τα βασικά στοιχεία διεπαφής (UI elements) που χρησιμοποιούνται στο σύστημα είναι:

- Κουμπιά (Buttons): Χρησιμοποιούνται για βασικές λειτουργίες όπως δημιουργία μαθήματος, προσθήκη παραδοτέου, υποβολή αρχείων, εκκίνηση ανάλυσης, προβολή αναφοράς και διαγραφή εγγραφών.
- Φόρμες εισαγωγής δεδομένων (Input Fields): Χρησιμοποιούνται για την εισαγωγή στοιχείων όπως τίτλος μαθήματος, ακαδημαϊκό εξάμηνο, τίτλος παραδοτέου και στοιχεία λογαριασμού όπως το email, τον κωδικό του χρήστη και τον κωδικό μιας χρήσης (OTP).
- Πίνακες και λίστες δεδομένων: Χρησιμοποιούνται για την παρουσίαση μαθημάτων, παραδοτέων, ομάδων και εργασιών με οργανωμένο τρόπο, ώστε ο χρήστης να μπορεί να εντοπίζει εύκολα τις πληροφορίες που χρειάζεται.
- Φίλτρα: Επιτρέπουν στον χρήστη να φιλτράρει τις υποβολές με βάση μάθημα, ακαδημαϊκό εξάμηνο, παραδοτέο ή ομάδα.
- Κάρτες πληροφορίας: Χρησιμοποιούνται για τη συνοπτική παρουσίαση στατιστικών, αποτελεσμάτων ανάλυσης και βασικών πληροφοριών για κάθε υποβολή.
- Αναφορές αποτελεσμάτων: Παρουσιάζουν τα αποτελέσματα της ανάλυσης με κατανοητό τρόπο, χρησιμοποιώντας ποσοστά, κατηγοριοποιήσεις και αναλυτικές πληροφορίες.
- Μηνύματα επιβεβαίωσης και σφαλμάτων: Ενημερώνουν τον χρήστη για την επιτυχία ή αποτυχία μιας ενέργειας, όπως επιτυχής υποβολή αρχείου, αποτυχία ανάλυσης ή μη υποστηριζόμενος τύπος αρχείου.
- Γραφήματα και στατιστικές απεικονίσεις: Χρησιμοποιούνται για την παρουσίαση συγκεντρωτικών αποτελεσμάτων ανά παραδοτέο, δίνοντας στον καθηγητή μια πιο συνολική εικόνα των υποβολών

Η διεπαφή έχει σχεδιαστεί ώστε να είναι όσο το δυνατόν πιο ευανάγνωστη και φιλική προς τον χρήστη. Ο χρήστης δεν χρειάζεται να γνωρίζει τεχνικές λεπτομέρειες για τον τρόπο λειτουργίας των μηχανισμών ανάλυσης. Αντίθετα, το σύστημα παρουσιάζει τα

αποτελέσματα σε μορφή που μπορεί να αξιοποιηθεί εύκολα κατά τη διαδικασία αξιολόγησης των εργασιών των φοιτητών.

Διεπαφές Υλικού

Για τη χρήση του συστήματος απαιτείται μια βασική υπολογιστική συσκευή με δυνατότητα σύνδεσης στο διαδίκτυο και πρόσβαση στο δίκτυο του Τμήματος. Η εφαρμογή είναι διαδικτυακή και εκτελείται μέσω browser, επομένως δεν απαιτείται εγκατάσταση τοπικής εφαρμογής. Για λόγους ασφάλειας, η πρόσβαση στο σύστημα πραγματοποιείται είτε μέσα από το δίκτυο του Τμήματος είτε απομακρυσμένα μέσω VPN.

Οι βασικές διεπαφές υλικού που απαιτούνται είναι:

- Οθόνη προβολής, για την εμφάνιση του γραφικού περιβάλλοντος, των εργασιών και των αναφορών.
- Πληκτρολόγιο, για την εισαγωγή στοιχείων στις φόρμες του συστήματος.
- Ποντίκι ή touchpad, για την πλοήγηση, την επιλογή ενεργειών και την υποβολή αρχείων.
- Σύνδεση στο διαδίκτυο, για την πρόσβαση στην εφαρμογή και την επικοινωνία με τον server.
- Πρόσβαση στο δίκτυο του Τμήματος ή σύνδεση μέσω VPN, για την ασφαλή χρήση της πλατφόρμας εκτός του δικτύου του πανεπιστημίου.

Το σύστημα μπορεί να χρησιμοποιηθεί από σταθερούς υπολογιστές, φορητούς υπολογιστές και συσκευές με σύγχρονο browser. Ωστόσο, για την καλύτερη εμπειρία χρήσης, ιδιαίτερα κατά τη διαχείριση πολλών αρχείων και την προβολή αναφορών, προτείνεται η χρήση σε desktop ή laptop περιβάλλον.

3.1.2 Λειτουργικές Απαιτήσεις

Οι λειτουργικές απαιτήσεις περιγράφουν τις βασικές λειτουργίες που πρέπει να παρέχει το σύστημα στους χρήστες του. Στην παρούσα εργασία, οι λειτουργικές απαιτήσεις προέκυψαν από τον στόχο της δημιουργίας μιας πλατφόρμας που να συνδυάζει τη διαχείριση ακαδημαϊκών εργασιών με την ανάλυση περιεχομένου τους.

Οι βασικές λειτουργικές απαιτήσεις είναι οι ακόλουθες:

- Το σύστημα επιτρέπει την εγγραφή νέων χρηστών.
- Το σύστημα υποστηρίζει την επιβεβαίωση του email μέσω κωδικού μιας χρήσης (OTP).
- Το σύστημα επιτρέπει στον διαχειριστή να εγκρίνει ή να απορρίπτει λογαριασμούς.
- Το σύστημα υποστηρίζει την σύνδεση και την αποσύνδεση των χρηστών.
- Το σύστημα προστατεύει την πρόσβαση στις λειτουργίες μέσω JWT token.
- Το σύστημα επιτρέπει στον καθηγητή να δημιουργεί, να επεξεργάζεται και να διαγράφει μαθήματα.
- Το σύστημα επιτρέπει τη δημιουργία ακαδημαϊκών εξαμήνων για κάθε μάθημα.
- Το σύστημα επιτρέπει τη δημιουργία παραδοτέων μέσα σε κάθε ακαδημαϊκό εξάμηνο.
- Το σύστημα επιτρέπει την αυτόματη δημιουργία ομάδων για κάθε παραδοτέο.
- Το σύστημα επιτρέπει τη μαζική υποβολή αρχείων μέσω ZIP.
- Το σύστημα αναγνωρίζει τον τύπο κάθε αρχείου και εφαρμόζει την κατάλληλη διαδικασία ανάλυσης.
- Το σύστημα υποστηρίζει ανάλυση DOCX αρχείων σε επίπεδο παραγράφου.
- Το σύστημα υποστηρίζει ανάλυση αρχείων πηγαίου κώδικα.
- Το σύστημα δημιουργεί αναφορά ανάλυσης για κάθε αρχείο.
- Το σύστημα παρέχει στατιστικά ανά παραδοτέο.
- Το σύστημα μπορεί να χειριστεί περιπτώσεις μη υποστηριζόμενων αρχείων χωρίς να επηρεαστεί η συνολική λειτουργία του.

Οι απαιτήσεις αυτές καλύπτουν τις βασικές ανάγκες της πλατφόρμας και αποτελούν τη βάση πάνω στην οποία σχεδιάστηκε και υλοποιήθηκε το σύστημα.

3.1.3 Μη Λειτουργικές Απαιτήσεις

Επίσης λήφθηκε υπόψη μια σειρά από μη λειτουργικές απαιτήσεις, οι οποίες είναι απαραίτητες για την ομαλή, ασφαλή και αξιόπιστη λειτουργία του συστήματος. Οι απαιτήσεις αυτές δεν αφορούν άμεσα συγκεκριμένες λειτουργίες, αλλά χαρακτηριστικά ποιότητας του συστήματος.

Απαιτήσεις Ασφάλειας

Η ασφάλεια αποτελεί βασική απαίτηση του συστήματος, καθώς η εφαρμογή διαχειρίζεται λογαριασμούς χρηστών, ακαδημαϊκά δεδομένα και εργασίες των φοιτητών. Για τον λόγο αυτό, η πρόσβαση στις προστατευμένες λειτουργίες γίνεται μέσω JWT token [13]. Κάθε αίτημα προς το backend ελέγχεται ώστε να διαπιστωθεί αν ο χρήστης είναι ταυτοποιημένος και αν έχει δικαίωμα πρόσβασης στη συγκεκριμένη λειτουργία.

Επιπλέον, τα passwords των χρηστών δεν αποθηκεύονται σε απλή μορφή, αλλά αποθηκεύονται κρυπτογραφημένα μέσω hashing [14]. Το σύστημα επίσης περιορίζει την πρόσβαση στα δεδομένα, έτσι ώστε κάθε καθηγητής να βλέπει μόνο τα δικά του μαθήματα, παραδοτέα και αρχεία.

Απαιτήσεις Ευχρηστίας

Το σύστημα σχεδιάστηκε με έμφαση στην απλότητα και στην ευκολία χρήσης. Ο καθηγητής μπορεί να οργανώσει ένα μάθημα, να δημιουργήσει παραδοτέο, να ανεβάσει αρχεία και να δει αποτελέσματα χωρίς να χρειάζεται τεχνικές γνώσεις.

Ιδιαίτερη έμφαση δόθηκε στην κατανοητή παρουσίαση των αποτελεσμάτων. Τα αποτελέσματα της ανάλυσης δεν παρουσιάζονται μόνο ως τελικές ετικέτες, αλλά

συνοδεύονται από ποσοστά, ενδείξεις και αναλυτικές πληροφορίες, ώστε ο καθηγητής να μπορεί να τα ερμηνεύσει πιο εύκολα.

Απαιτήσεις Αξιοπιστίας

Το σύστημα λειτουργεί αξιόπιστα ακόμη και όταν ο χρήστης υποβάλει μη υποστηριζόμενα αρχεία, κενά ZIP, λανθασμένες μορφές αρχείων ή αρχεία που δεν μπορούν να αναλυθούν. Σε τέτοιες περιπτώσεις, η εφαρμογή πρέπει να εμφανίζει κατάλληλο μήνυμα στον χρήστη και να συνεχίζει να λειτουργεί κανονικά.

Η αξιοπιστία είναι ιδιαίτερα σημαντική επειδή η διαδικασία ανάλυσης μπορεί να αποτύχει για λόγους που δεν σχετίζονται απαραίτητα με το ίδιο το σύστημα, όπως ένα προβληματικό αρχείο, μη αναγνωρίσιμος τύπος αρχείου ή ένα σφάλμα κατά την ανάλυση των εργασιών. Για τον λόγο αυτό, το σύστημα χειρίζεται τέτοιες περιπτώσεις χωρίς να διακόπτεται η συνολική λειτουργία του συστήματος.

Απαιτήσεις Επεκτασιμότητας

Το σύστημα σχεδιάστηκε ώστε να μπορεί να επεκταθεί στο μέλλον. Η αρχιτεκτονική του επιτρέπει την προσθήκη νέων τύπων αρχείων και νέων μηχανισμών ανάλυσης. Η διάκριση ανάμεσα στο frontend, στο backend, στη βάση δεδομένων και στους μηχανισμούς ανάλυσης βοηθά ώστε οι αλλαγές σε ένα μέρος του συστήματος να μην απαιτούν πλήρη επανασχεδιασμό ολόκληρης της εφαρμογής.

Η επεκτασιμότητα είναι σημαντική, καθώς ο τομέας της Τεχνητής Νοημοσύνης εξελίσσεται συνεχώς. Επομένως, το σύστημα πρέπει να μπορεί να προσαρμοστεί σε νέες τεχνικές και νέες ανάγκες που μπορεί να προκύψουν στο μέλλον.

Απαιτήσεις Διαθεσιμότητας

Το σύστημα φιλοξενείται σε πραγματικό server, ώστε να είναι προσβάσιμο μέσω browser χωρίς τοπική εγκατάσταση. Η πρόσβαση πραγματοποιείται μέσω HTTPS, ενισχύοντας την ασφάλεια της επικοινωνίας μεταξύ του χρήστη και της εφαρμογής.

3.2 Κατηγορίες Χρηστών

Το σύστημα απευθύνεται σε μέλη του ακαδημαϊκού προσωπικού, και συγκεκριμένα σε καθηγητές που χρειάζονται έναν οργανωμένο τρόπο να διαχειρίζονται τις εργασίες των φοιτητών τους και να λαμβάνουν βοηθητικές ενδείξεις σχετικά με πιθανή χρήση εργαλείων ΑΙ. Σε αντίθεση με συστήματα που υποστηρίζουν πολλούς ρόλους με διαφορετικά δικαιώματα, το παρόν σύστημα διατηρεί μόνο δύο κατηγορίες χρηστών, τον διαχειριστή και τους καθηγητές.

Ο διαχειριστής είναι υπεύθυνος για τη γενική εποπτεία του συστήματος. Η κύρια αρμοδιότητά του είναι η έγκριση ή απόρριψη αιτήσεων εγγραφής νέων χρηστών. Ο διαχειριστής δεν ασχολείται με τα ακαδημαϊκά δεδομένα, δεν βλέπει μαθήματα, παραδοτέα ή υποβολές των καθηγητών, και δεν παρεμβαίνει στη διαδικασία αξιολόγησης. Ο ρόλος του περιορίζεται στη διασφάλιση ότι μόνο εξουσιοδοτημένα άτομα έχουν πρόσβαση στο σύστημα.

Ο καθηγητής είναι ο κύριος χρήστης του συστήματος. Μετά τη σύνδεσή του, μπορεί να δημιουργεί μαθήματα, ακαδημαϊκά εξάμηνα, παραδοτέα και ομάδες. Ανεβάζει τις εργασίες των φοιτητών του, εκτελεί την ανάλυση και βλέπει τα αποτελέσματα μέσω αναφορών και στατιστικών. Κάθε καθηγητής λειτουργεί μόνο μέσα στο δικό του περιβάλλον εργασίας. Βλέπει μόνο τα δικά του μαθήματα, τα δικά του παραδοτέα και τις δικές του υποβολές, χωρίς να έχει πρόσβαση ή δυνατότητα παρέμβασης στα δεδομένα κάποιου άλλου καθηγητή.

Η πρόσβαση στο σύστημα γίνεται μέσω εγγραφής, επιβεβαίωσης email μέσω κωδικού μιας χρήσης (OTP) και έγκρισης από τον διαχειριστή. Η τριπλή αυτή διαδικασία εξασφαλίζει ότι μόνο εξουσιοδοτημένα άτομα μπορούν να χρησιμοποιήσουν την εφαρμογή, διατηρώντας ταυτόχρονα την ασφάλεια και την απλότητα της εμπειρίας χρήσης.

Η απόφαση να μην συμπεριληφθεί ρόλος του φοιτητή στην τρέχουσα έκδοση ήταν συνειδητή. Στο παρόν στάδιο, τα αρχεία ανεβαίνουν από τον ίδιο τον καθηγητή, ο οποίος συλλέγει τις εργασίες μέσω άλλων εργαλείων και τις υποβάλλει στο σύστημα. Ωστόσο, η αρχιτεκτονική του συστήματος επιτρέπει τη μελλοντική προσθήκη νέων ρόλων, όπως ο ρόλος του φοιτητή, ο οποίος θα μπορεί να υποβάλλει απευθείας τις εργασίες του.

3.3 Περιορισμοί Συστήματος

Το σύστημα καλύπτει τις βασικές ανάγκες για τη διαχείριση και ανάλυση των εργασιών των φοιτητών, ωστόσο υπάρχουν κάποιοι περιορισμοί που αξίζει να αναφερθούν.

- Ο πιο σημαντικός αφορά την ακρίβεια των αποτελεσμάτων. Τα αποτελέσματα της ανάλυσης δεν αποδεικνύουν ότι κάποιο αρχείο δημιουργήθηκε από εργαλείο AI [11, 12, 18]. Αποτελούν ενδείξεις και βοηθητικές πληροφορίες, όχι τελικές αποφάσεις. Καμία μέθοδος ανίχνευσης δεν είναι αλάνθαστη, και μπορεί να υπάρξουν τόσο λανθασμένες θετικές όσο και λανθασμένες αρνητικές εκτιμήσεις. Η τελική κρίση ανήκει πάντα στον καθηγητή.
- Όσον αφορά τους τύπους αρχείων, η ανάλυση κειμένου γίνεται μόνο σε αρχεία DOCX. Άλλες μορφές, όπως PDF, δεν υποστηρίζονται ακόμα. Για τον πηγαίο κώδικα, η ανάλυση καλύπτει τις γλώσσες για τις οποίες υπάρχει tree-sitter parser, όπως Python, Java, JavaScript, C++, Go, PHP και Ruby.
- Ένας ακόμα περιορισμός σχετίζεται με την απόδοση. Ο server του τμήματος δεν διαθέτει GPU, οπότε η ανάλυση κώδικα μέσω DetectCodeGPT εκτελείται σε CPU.

Αυτό σημαίνει ότι ο χρόνος ανάλυσης των εγγράφων είναι μεγαλύτερος απ' ό,τι θα ήταν σε περιβάλλον με GPU.

- Η μαζική υποβολή μέσω ZIP απαιτεί μια συγκεκριμένη δομή φακέλων. Κάθε φάκελος μέσα στο ZIP αντιστοιχεί σε μία ομάδα, και τα αρχεία μέσα του γίνονται υποβολές.
- Η πρόσβαση στο σύστημα γίνεται μόνο μέσα από το δίκτυο του Τμήματος Πληροφορικής ή απομακρυσμένα μέσω VPN. Χρήστες εκτός δικτύου χωρίς VPN δεν μπορούν να συνδεθούν.
- Τέλος, οι μέθοδοι ανίχνευσης που χρησιμοποιούνται βρίσκονται ακόμα σε ερευνητικό στάδιο. Το DetectCodeGPT δεν έχει δοκιμαστεί εκτενώς σε όλους τους τύπους κώδικα, και η ανάλυση κειμένου δεν έχει αξιολογηθεί ειδικά για ελληνικό κείμενο [12]. Αυτό σημαίνει ότι τα αποτελέσματα μπορεί να διαφέρουν ανάλογα με τη γλώσσα, το μήκος του κειμένου και τον τύπο του κώδικα.

Οι περιορισμοί αυτοί δεν μειώνουν τη χρησιμότητα του συστήματος, αλλά είναι σημαντικό να τους γνωρίζουν οι καθηγητές που θα το χρησιμοποιούν αλλά και όποιος θελήσει να το επεκτείνει στο μέλλον.

3.4 Σενάρια Χρήσης

Στην ενότητα αυτή περιγράφονται τα βασικά σενάρια χρήσης του συστήματος, δηλαδή οι ενέργειες που εκτελούν οι χρήστες κατά την αλληλεπίδρασή τους με το σύστημα. Τα σενάρια αυτά καλύπτουν τις κύριες λειτουργίες και για τους δύο ρόλους χρηστών, τον διαχειριστή και τον καθηγητή.

Σενάριο 1: Εγγραφή και σύνδεση στο σύστημα

Ο καθηγητής επισκέπτεται τη σελίδα register και συμπληρώνει τα στοιχεία του (όνομα, email, κωδικός). Το σύστημα αποστέλλει κωδικό μιας χρήσης (OTP) στο προσωπικό του

email. Ο χρήστης εισάγει τον κωδικό για να επιβεβαιώσει το email του. Μετά την επιβεβαίωση, ο λογαριασμός παραμένει ανενεργός μέχρι να τον εγκρίνει ο διαχειριστής. Μόλις εγκριθεί, ο καθηγητής μπορεί να συνδεθεί κανονικά.

Σενάριο 2: Έγκριση χρηστών από τον διαχειριστή

Ο διαχειριστής συνδέεται στο σύστημα και βλέπει τη λίστα με τους χρήστες που περιμένουν έγκριση. Για κάθε αίτηση μπορεί να εγκρίνει ή να απορρίψει τον λογαριασμό. Μετά την έγκριση, ο χρήστης λαμβάνει πρόσβαση στο σύστημα.

Σενάριο 3: Δημιουργία μαθήματος και παραδοτέου

Ο καθηγητής δημιουργεί ένα νέο μάθημα (π.χ. EPL484 - Software Evolution). Στη συνέχεια προσθέτει ακαδημαϊκό εξάμηνο (π.χ. Spring 2026) και μέσα σε αυτό δημιουργεί ένα ή περισσότερα παραδοτέα (π.χ. Project 1, Assignment 2). Η δομή αυτή ακολουθεί τον τρόπο που οργανώνονται οι εργασίες στο σύστημα.

Σενάριο 4: Μαζική υποβολή αρχείων μέσω ZIP

Ο καθηγητής ετοιμάζει ένα ZIP αρχείο με φακέλους ανά ομάδα φοιτητών. Κάθε φάκελος περιέχει τα αρχεία της αντίστοιχης ομάδας. Ανεβάζει το ZIP στο σύστημα επιλέγοντας το παραδοτέο στο οποίο αντιστοιχούν τα αρχεία. Το σύστημα εξάγει αυτόματα τα αρχεία, δημιουργεί τις ομάδες με βάση τα ονόματα των φακέλων και καταχωρεί κάθε αρχείο ως υποβολή. Ο καθηγητής μπορεί επίσης να επιλέξει αν θέλει να εκτελεστεί αυτόματα την ανάλυση των αρχείων κατά την υποβολή τους.

Σενάριο 5: Εκτέλεση ανάλυσης

Η ανάλυση μπορεί να εκτελεστεί με δύο τρόπους. Είτε αυτόματα κατά τη στιγμή της υποβολής, είτε χειροκίνητα σε μεταγενέστερο στάδιο. Το σύστημα αναγνωρίζει τον τύπο κάθε αρχείου και εφαρμόζει την κατάλληλη μέθοδο ανάλυσης. Για αρχεία DOCX, η ανάλυση γίνεται σε επίπεδο παραγράφου ενώ για αρχεία πηγαίου κώδικα, χρησιμοποιείται το DetectCodeGPT.

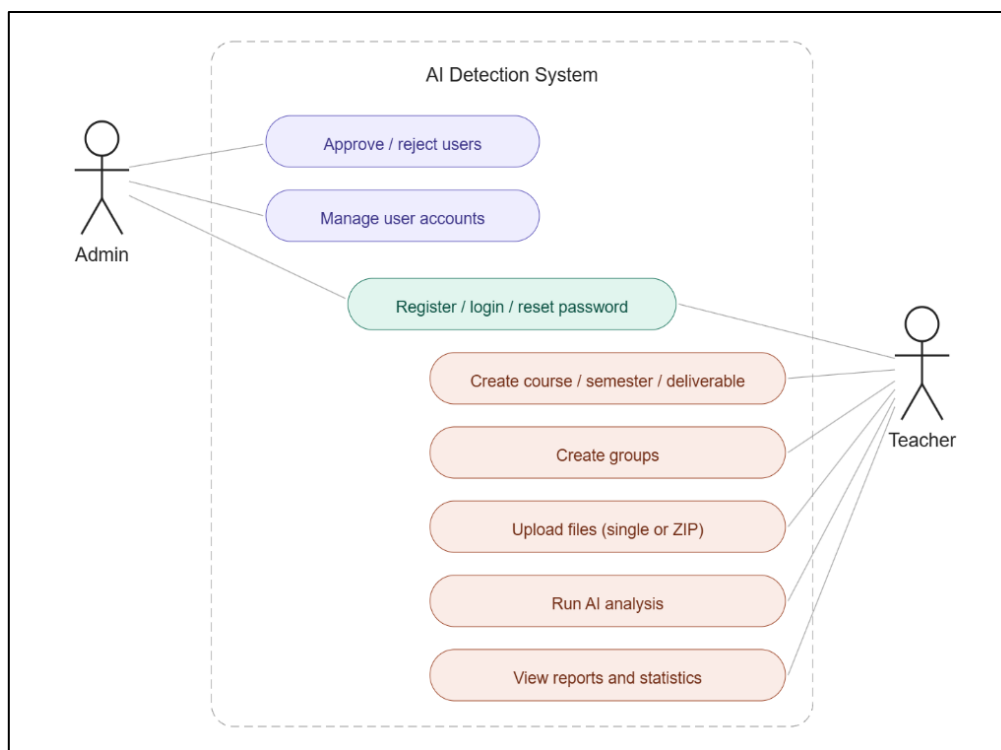
Σενάριο 6: Προβολή αποτελεσμάτων ανάλυσης

Μετά την ολοκλήρωση της ανάλυσης, ο καθηγητής μπορεί να δει την αναφορά κάθε αρχείου. Για τα DOCX, η αναφορά περιλαμβάνει ανάλυση ανά παράγραφο με ποσοστά, κατηγοριοποίηση και συνολική εκτίμηση. Για τον κώδικα, η αναφορά περιλαμβάνει ποσοστό πιθανότητας, ενδείξεις και αναλυτικές πληροφορίες. Ο καθηγητής μπορεί επίσης να δει τα στατιστικά ολόκληρου του παραδοτέου.

Σενάριο 7: Επαναφορά κωδικού πρόσβασης

Ο χρήστης που ξέχασε τον κωδικό του πατάει «Forgot Password» στη σελίδα login. Εισάγει το email του και λαμβάνει κωδικό OTP. Μετά την εισαγωγή του κωδικού, μπορεί να ορίσει νέο κωδικό πρόσβασης.

Στο Σχήμα 2 παρουσιάζεται το διάγραμμα σεναρίων χρήσης (Use Case Diagram) του συστήματος, στο οποίο φαίνονται οι βασικές ενέργειες κάθε ρόλου του συστήματος.



Σχήμα 2 - Διάγραμμα Σεναρίων Χρήσης (Use Case Diagram)

Κεφάλαιο 4

4. Σχεδίαση και Υλοποίηση

4.1 Αρχιτεκτονική Συστήματος	37
4.2 Εργαλεία Ανάπτυξης	39
4.3 Ασφάλεια Συστήματος	44
4.4 Βάση Δεδομένων	45
4.5 Δομή Αρχείων	52
4.6 Βιβλιοθήκες	52

4.1 Αρχιτεκτονική Συστήματος

Το σύστημα ακολουθεί αρχιτεκτονική client-server. Ο χρήστης αλληλοεπιδρά με την εφαρμογή μέσω του browser (frontend), ο οποίος επικοινωνεί με τον server (backend) μέσω REST API. Ο server αναλαμβάνει τη λογική της εφαρμογής, τη διαχείριση δεδομένων, την αποθήκευση αρχείων και την εκτέλεση της ανάλυσης των εργασιών.

Η αρχιτεκτονική αποτελείται από τα ακόλουθα τμήματα:

Το frontend είναι μια διαδικτυακή εφαρμογή βασισμένη σε Next.js (React), η οποία εκτελείται στον browser του χρήστη. Παρέχει τη διεπαφή χρήστη για σύνδεση, διαχείριση μαθημάτων, ανέβασμα αρχείων και την προβολή των αποτελεσμάτων. Το frontend δεν εκτελεί καμία λογική ανάλυσης ή επεξεργασίας δεδομένων. Αποστέλλει αιτήματα στο backend και εμφανίζει τα δεδομένα που λαμβάνει ως απάντηση.

Το backend είναι ένα REST API υλοποιημένο με FastAPI (Python). Αναλαμβάνει τη διαχείριση χρηστών, τα μαθήματα, τα αρχεία και τη βάση δεδομένων. Επίσης, εκτελεί τους μηχανισμούς ανάλυσης για τα αρχεία που υποβάλλονται. Η λογική της εφαρμογής βρίσκεται αποκλειστικά στο backend, ώστε αν το frontend χρειάζεται να αντικατασταθεί ή να τροποποιηθεί στο μέλλον, να μην επηρεαστεί η υπόλοιπη λειτουργία του συστήματος.

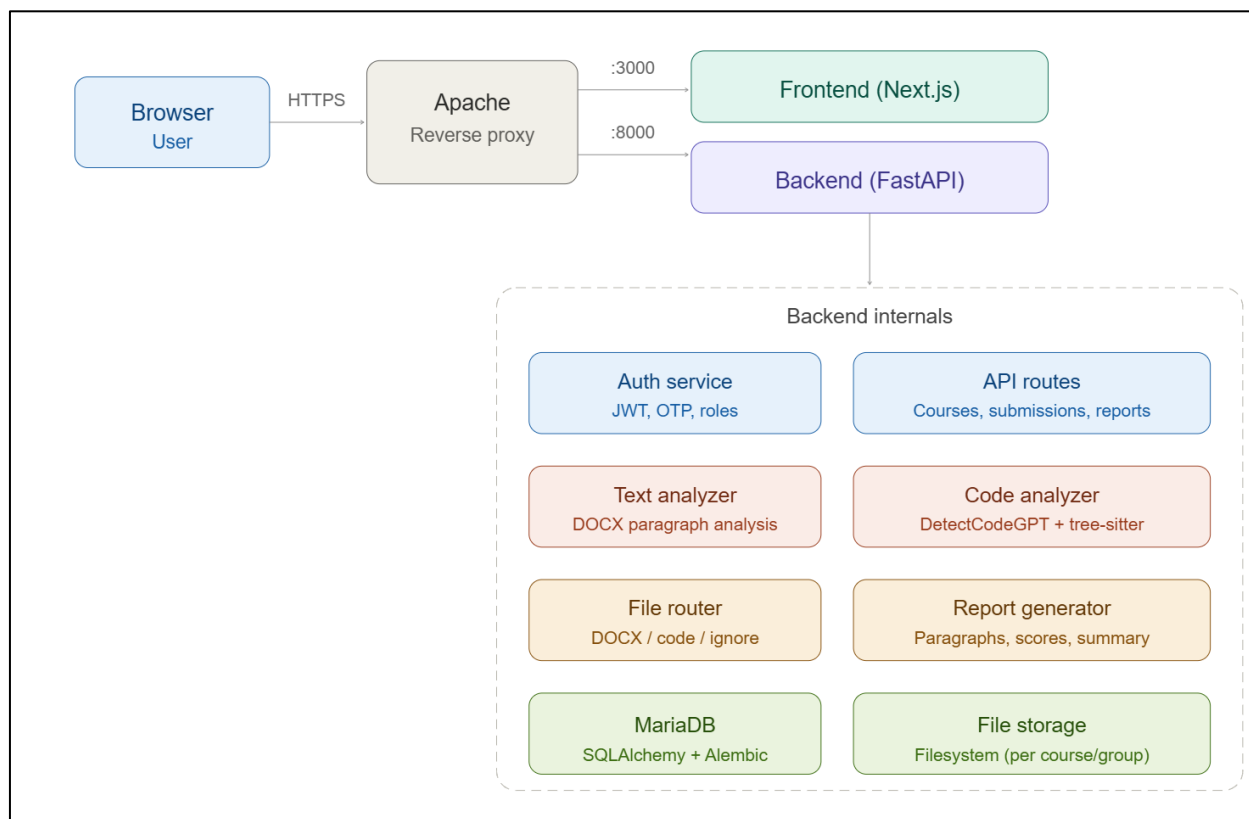
Η βάση δεδομένων (MariaDB) αποθηκεύει τα δεδομένα χρηστών, μαθημάτων, ακαδημαϊκών εξαμήνων, παραδοτέων, ομάδων, υποβολών και αναφορών ανάλυσης. Η επικοινωνία με τη βάση γίνεται μέσω SQLAlchemy, ενώ οι αλλαγές στο schema διαχειρίζονται μέσω Alembic.

Οι μηχανισμοί ανάλυσης αποτελούν ξεχωριστά modules μέσα στο backend. Το σύστημα αναγνωρίζει τον τύπο κάθε αρχείου και εφαρμόζει την κατάλληλη μέθοδο. Για αρχεία DOCX χρησιμοποιείται ανάλυση σε επίπεδο παραγράφου, ενώ για αρχεία πηγαίου κώδικα χρησιμοποιείται το DetectCodeGPT σε συνδυασμό με tree-sitter parsers.

Ο web server (Apache) λειτουργεί ως reverse proxy [15], δηλαδή είναι αυτός που δέχεται τα αιτήματα από τον browser και τα στέλνει στο σωστό μέρος. Frontend (port 3000) & Backend (port 8000). Εξυπηρετεί επίσης HTTPS μέσω του wildcard certificate του Πανεπιστημίου, ώστε η επικοινωνία μεταξύ χρήστη και server να είναι κρυπτογραφημένη.

Η αποθήκευση των αρχείων γίνεται στο filesystem του server. Κάθε αρχείο αποθηκεύεται σε φάκελο που αντιστοιχεί στο μάθημα, το ακαδημαϊκό εξάμηνο, το παραδοτέο και την ομάδα, ώστε η δομή του filesystem να αντικατοπτρίζει τη δομή των δεδομένων στη βάση.

Στο Σχήμα 3 παρουσιάζεται η συνολική αρχιτεκτονική του συστήματος.



Σχήμα 3 - Αρχιτεκτονική Συστήματος

4.2 Εργαλεία Ανάπτυξης

Πιο κάτω αναλύονται οι τεχνολογίες και τα εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη του συστήματος.

4.2.1 Visual Studio Code

Το Visual Studio Code είναι ένα εργαλείο επεξεργασίας κώδικα που αναπτύχθηκε από τη Microsoft. Χρησιμοποιήθηκε ως το κύριο περιβάλλον ανάπτυξης για ολόκληρο το project, τόσο για τον κώδικα Python του backend όσο και για τον κώδικα TypeScript [21] (React) του frontend. Επιλέχθηκε γιατί υποστηρίζει πολλαπλές γλώσσες προγραμματισμού, διαθέτει

ενσωματωμένο terminal και ενσωμάτωση με Git, και προσφέρει μεγάλο αριθμό extensions που διευκολύνουν τη διαδικασία ανάπτυξης. Κατά τη διάρκεια της εργασίας χρησιμοποιήθηκαν extensions για Python, TypeScript [21], React και για τη σύνδεση με τον απομακρυσμένο server μέσω SSH.

4.2.2 Γλώσσα Προγραμματισμού Python

Η Python είναι μια γλώσσα προγραμματισμού γενικής χρήσης που χρησιμοποιήθηκε για ολόκληρο το backend του συστήματος. Επιλέχθηκε γιατί διαθέτει πλούσιο οικοσύστημα βιβλιοθηκών, ιδιαίτερα στους τομείς της ανάλυσης δεδομένων και της μηχανικής μάθησης, που είναι κεντρικοί στη λειτουργία του συστήματος. Τα εργαλεία ανίχνευσης AI-generated περιεχομένου που ενσωματώθηκαν στο σύστημα, όπως το DetectCodeGPT, είναι γραμμένα σε Python, οπότε η επιλογή της ίδιας γλώσσας για το backend διευκόλυνε την ενσωμάτωσή τους. Η Python χρησιμοποιείται επίσης σε συνδυασμό με βιβλιοθήκες όπως PyTorch και Transformers για την εκτέλεση γλωσσικών μοντέλων, καθώς και με βιβλιοθήκες όπως python-docx για την ανάγνωση αρχείων DOCX.

4.2.3 FastAPI

Το FastAPI [5] είναι ένα framework της Python για τη δημιουργία REST APIs. Χρησιμοποιήθηκε για την υλοποίηση ολόκληρου του backend, δηλαδή για τη διαχείριση χρηστών, μαθημάτων, υποβολών, αναλύσεων και αναφορών. Επιλέχθηκε γιατί υποστηρίζει ασύγχρονη εκτέλεση, κάτι που είναι χρήσιμο όταν το σύστημα εκτελεί αναλύσεις που μπορεί να πάρουν χρόνο. Το FastAPI ελέγχει αυτόματα αν τα δεδομένα που στέλνει ο χρήστης είναι στη σωστή μορφή πριν τα επεξεργαστεί, μέσω του Pydantic [22].

4.2.4 Γλώσσα Περιγραφής HTML

Η HTML (HyperText Markup Language) είναι η βασική γλώσσα περιγραφής που χρησιμοποιείται για τη δόμηση του περιεχομένου των ιστοσελίδων. Καθορίζει τη δομή κάθε

σελίδας, δηλαδή ποια στοιχεία υπάρχουν και πώς οργανώνονται, όπως φόρμες, πίνακες, κουμπιά και περιοχές εμφάνισης αποτελεσμάτων. Στο σύστημα η HTML δεν γράφεται απευθείας, αλλά παράγεται μέσω JSX, μιας σύνταξης που χρησιμοποιεί η React και επιτρέπει τη δημιουργία HTML μέσα σε κώδικα JavaScript.

4.2.5 Γλώσσα Μορφοποίησης CSS

Η CSS (Cascading Style Sheets) είναι η γλώσσα που χρησιμοποιείται για τη μορφοποίηση και την εμφάνιση των στοιχείων HTML. Καθορίζει χρώματα, γραμματοσειρές, αποστάσεις, μεγέθη και τη γενική εικόνα της εφαρμογής. Στο σύστημα η CSS χρησιμοποιήθηκε μέσω inline styles μέσα στα React components.

4.2.6 Γλώσσα Προγραμματισμού JavaScript / TypeScript

Η JavaScript είναι η γλώσσα προγραμματισμού που εκτελείται στον browser του χρήστη και επιτρέπει τη δημιουργία δυναμικών ιστοσελίδων. Μέσω αυτής γίνεται η αποστολή αιτημάτων στο backend, η ενημέρωση της διεπαφής χωρίς ανανέωση σελίδας και η δυναμική εμφάνιση δεδομένων.

Στο frontend του συστήματος χρησιμοποιήθηκε η TypeScript [21] αντί για καθαρή JavaScript. Η TypeScript είναι ένα υπερσύνολο της JavaScript που προσθέτει static typing, δηλαδή τη δυνατότητα ορισμού τύπων στις μεταβλητές και τις παραμέτρους συναρτήσεων. Αυτό βοηθά στον εντοπισμό σφαλμάτων κατά τη διάρκεια της ανάπτυξης, πριν εκτελεστεί ο κώδικας, και κάνει τον κώδικα πιο ασφαλή και ευανάγνωστο.

4.2.7 React

Η React [20] είναι μια βιβλιοθήκη JavaScript που αναπτύχθηκε από τη Meta και χρησιμοποιείται για τη δημιουργία διαδραστικών διεπαφών χρήστη. Η βασική αρχή της είναι η χρήση επαναχρησιμοποιήσιμων components, δηλαδή ανεξάρτητων κομματιών κώδικα που

αντιπροσωπεύουν τμήματα της διεπαφής. Κάθε component διαχειρίζεται τη δική του κατάσταση και ενημερώνεται αυτόματα όταν αλλάζουν τα δεδομένα. Η προσέγγιση αυτή κάνει τον κώδικα πιο οργανωμένο και ευκολότερο στη συντήρηση, ιδιαίτερα σε εφαρμογές με πολλές οθόνες και λειτουργίες. Στο σύστημα, όλες οι σελίδες και τα στοιχεία της διεπαφής υλοποιήθηκαν ως React components. Αυτό περιλαμβάνει τα dashboards, τις φόρμες δημιουργίας μαθημάτων και παραδοτέων, τους πίνακες υποβολών, τις σελίδες αναφορών ανάλυσης και τα στατιστικά γραφήματα.

4.2.8 Next.js

Το Next.js [6] είναι ένα framework βασισμένο στη React που χρησιμοποιήθηκε για την ανάπτυξη του frontend. Προσθέτει δυνατότητες πάνω από τη React, όπως server-side rendering, δυναμικό routing μέσω φακέλων και αυτόματη βελτιστοποίηση εικόνων και κώδικα. Η έκδοση που χρησιμοποιήθηκε είναι η 16, η οποία ενσωματώνει το App Router, ένα νεότερο σύστημα routing που οργανώνει τις σελίδες σε φακέλους αντί για αρχεία ρυθμίσεων. Αυτό σημαίνει ότι η δομή των φακέλων στον κώδικα αντιστοιχεί απευθείας στα URLs της εφαρμογής, κάτι που κάνει την πλοήγηση πιο εύκολη.

4.2.9 SQLAlchemy και Alembic

Το SQLAlchemy [8] είναι ένα ORM (Object-Relational Mapper) της Python που χρησιμοποιήθηκε για την επικοινωνία του backend με τη βάση δεδομένων. Αντί να γράφονται SQL queries απευθείας, τα δεδομένα ορίζονται ως Python classes και η βιβλιοθήκη αναλαμβάνει τη μετατροπή τους σε εντολές SQL. Αυτό κάνει τον κώδικα πιο κατανοητό και λιγότερο εξαρτημένο από τη συγκεκριμένη βάση δεδομένων. Στην πράξη, αυτό σημαίνει ότι αν στο μέλλον χρειαστεί αλλαγή βάσης δεδομένων, ο κώδικας δεν θα χρειαστεί σημαντικές τροποποιήσεις.

Το Alembic χρησιμοποιήθηκε για τη διαχείριση αλλαγών στο schema της βάσης. Κάθε αλλαγή στη δομή των πινάκων καταγράφεται ως migration script, ώστε να μπορεί να

αναπαραχθεί σε οποιοδήποτε περιβάλλον. Αυτό ήταν ιδιαίτερα χρήσιμο κατά τη μετάβαση από το τοπικό περιβάλλον ανάπτυξης στον server του τμήματος, γιατί η βάση δεδομένων μπορούσε να δημιουργηθεί από την αρχή τρέχοντας τα migration scripts.

4.2.10 MariaDB

Η MariaDB [7] είναι ένα σύστημα διαχείρισης σχεσιακής βάσης δεδομένων ανοιχτού κώδικα, συμβατό με MySQL. Χρησιμοποιείται στο production περιβάλλον του συστήματος για την αποθήκευση όλων των δεδομένων, δηλαδή χρηστών, μαθημάτων, ακαδημαϊκών εξαμήνων, παραδοτέων, ομάδων, υποβολών και αναφορών ανάλυσης. Επιλέχθηκε γιατί ήταν ήδη διαθέσιμη στον server του Τμήματος σε έκδοση 10.5, είναι αξιόπιστη, υποστηρίζεται ευρέως και είναι πλήρως συμβατή με το SQLAlchemy.

4.2.11 Σύστημα Διαχείρισης Βάσεων Δεδομένων phpMyAdmin

Το phpMyAdmin είναι μια διαδικτυακή εφαρμογή για τη διαχείριση βάσεων δεδομένων MySQL και MariaDB μέσω γραφικού περιβάλλοντος. Χρησιμοποιήθηκε κατά τη διάρκεια της ανάπτυξης του συστήματος. Βοήθησε στην αναζήτηση και επεξεργασία των δεδομένων στη βάση, χωρίς να χρειάζεται σύνδεση μέσω command line.

4.2.12 DetectCodeGPT

Το DetectCodeGPT [3] είναι μια μέθοδος ανίχνευσης AI-generated πηγαίου κώδικα. Βασίζεται σε perturbation-based detection, δηλαδή εφαρμόζει μικρές αλλαγές στον κώδικα και εξετάζει πώς αντιδρά ένα γλωσσικό μοντέλο σε αυτές τις αλλαγές. Η ιδέα είναι ότι αν ο κώδικας έχει δημιουργηθεί από AI, το μοντέλο αντιδρά με συγκεκριμένο τρόπο στις αλλαγές, ενώ σε ανθρώπινο κώδικα η αντίδραση είναι διαφορετική. Η μέθοδος βρίσκεται ακόμα σε ερευνητικό στάδιο, αλλά αποτελεί μια από τις πιο ενδιαφέρουσες προσεγγίσεις στον τομέα αυτό.

4.2.13 Tree-sitter

Το tree-sitter [4] είναι ένα εργαλείο parsing πηγαίου κώδικα που χρησιμοποιείται στο σύστημα μαζί με το DetectCodeGPT. Αναλύει τη δομή του κώδικα και δημιουργεί ένα Abstract Syntax Tree (AST), δηλαδή μια αναπαράσταση της δομής του προγράμματος με δένδρο. Μέσω αυτού, το σύστημα μπορεί να εξετάζει χαρακτηριστικά όπως τα ονόματα μεταβλητών, τη δομή των συναρτήσεων και τον τρόπο οργάνωσης του κώδικα. Υποστηρίζει πολλές γλώσσες προγραμματισμού, όπως Python, Java, JavaScript, C++, Go, PHP και Ruby, κάτι που επιτρέπει στο σύστημα να αναλύει κώδικα σε διαφορετικές γλώσσες.

4.3 Ασφάλεια Συστήματος

Η ασφάλεια του συστήματος αποτελεί βασική απαίτηση, καθώς η εφαρμογή διαχειρίζεται λογαριασμούς χρηστών, ακαδημαϊκά δεδομένα και εργασίες φοιτητών. Για τον λόγο αυτό, υλοποιήθηκαν μηχανισμοί που καλύπτουν τόσο την αυθεντικοποίηση των χρηστών όσο και τον έλεγχο πρόσβασης στα δεδομένα.

Η αυθεντικοποίηση βασίζεται σε JWT (JSON Web Tokens). Κάθε φορά που ένας χρήστης συνδέεται, το backend δημιουργεί ένα token το οποίο αποθηκεύεται στον browser μέσω cookie. Σε κάθε επόμενο αίτημα, το token αποστέλλεται μαζί με το αίτημα και το backend ελέγχει αν είναι έγκυρο. Αν το token έχει λήξει ή δεν είναι σωστό, ο χρήστης ανακατευθύνεται στη σελίδα σύνδεσης. Με αυτόν τον τρόπο δεν χρειάζεται να αποθηκεύονται sessions στον server.

Κατά την εγγραφή, ο χρήστης λαμβάνει έναν κωδικό μιας χρήσης (OTP) στο email του μέσω της υπηρεσίας Brevo [17]. Ο κωδικός αυτός πρέπει να εισαχθεί στο σύστημα για να ολοκληρωθεί η επιβεβαίωση του email. Αν ο χρήστης δεν εισάγει τον σωστό κωδικό, η εγγραφή δεν προχωράει. Ο ίδιος μηχανισμός χρησιμοποιείται και κατά την επαναφορά του κωδικού πρόσβασης.

Ακόμα και μετά την επιβεβαίωση του email, ο λογαριασμός παραμένει ανενεργός μέχρι να τον εγκρίνει ο διαχειριστής. Αυτό αποτρέπει μη εξουσιοδοτημένα άτομα από το να αποκτήσουν πρόσβαση στο σύστημα.

Επιπλέον, κατά την εγγραφή ο χρήστης πρέπει να εισάγει ένα invite code, το οποίο λειτουργεί ως πρώτο φίλτρο πρόσβασης. Χωρίς τον σωστό κωδικό, δεν επιτρέπεται η δημιουργία ενός λογαριασμού. Αυτό εμποδίζει άλλους χρήστες, που δεν είναι καθηγητές, να υποβάλουν αίτηση εγγραφής, μειώνοντας έτσι τον φόρτο του διαχειριστή.

Οι κωδικοί πρόσβασης δεν αποθηκεύονται σε απλή μορφή στη βάση δεδομένων. Πριν αποθηκευτούν, περνάνε από hashing μέσω bcrypt. Αυτό σημαίνει ότι ακόμα και αν κάποιος αποκτήσει πρόσβαση στη βάση, δεν μπορεί να ανακτήσει τους πραγματικούς κωδικούς.

Ο έλεγχος πρόσβασης γίνεται με βάση τον ρόλο του χρήστη. Κάθε endpoint στο backend ελέγχει αν ο χρήστης έχει δικαίωμα να εκτελέσει τη συγκεκριμένη ενέργεια. Ο καθηγητής βλέπει μόνο τα δικά του μαθήματα, τα παραδοτέα των δικών του φοιτητών και τις δικές του υποβολές. Ο διαχειριστής έχει πρόσβαση μόνο στη διαχείριση χρηστών, χωρίς δυνατότητα προβολής των δεδομένων των χρηστών.

Τέλος, η επικοινωνία μεταξύ του browser και του server γίνεται αποκλειστικά μέσω HTTPS, χρησιμοποιώντας το wildcard certificate του Πανεπιστημίου. Αυτό εξασφαλίζει ότι τα δεδομένα που ανταλλάσσονται, είναι ασφαλείς κατά την μεταφορά τους.

4.4 Βάση Δεδομένων

Η βάση δεδομένων του συστήματος αποθηκεύει όλες τις πληροφορίες που χρειάζονται για τη λειτουργία της εφαρμογής, από τους χρήστες μέχρι τα αποτελέσματα ανάλυσης. Η δομή της σχεδιάστηκε ώστε να αντικατοπτρίζει τον τρόπο με τον οποίο οργανώνονται πρακτικά οι ακαδημαϊκές εργασίες σε ένα πανεπιστημιακό περιβάλλον, δηλαδή μαθήματα που περιέχουν εξάμηνα, εξάμηνα που περιέχουν παραδοτέα, παραδοτέα που περιέχουν ομάδες, και ομάδες που περιέχουν τις εργασίες τους.

4.4.1 Πίνακες Βάσης Δεδομένων

Πιο κάτω παρουσιάζεται αναλυτική επεξήγηση κάθε πίνακα της βάσης δεδομένων.

Πίνακας roles

Αποθηκεύει τους διαθέσιμους ρόλους χρηστών. Κάθε ρόλος έχει ένα αναγνωριστικό (role_id), ένα όνομα (name) και μια περιγραφή (description). Στην τρέχουσα έκδοση του συστήματος υπάρχουν δύο ρόλοι: ADMIN και TEACHER.

#	Name	Type
1	role_id 	int(11)
2	name 	varchar(255)
3	description	varchar(255)

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	role_id
name	BTREE	Yes	No	name

Πίνακας users

Αποθηκεύει τις πληροφορίες των χρηστών. Τα πεδία περιλαμβάνουν το email, τον κωδικό πρόσβασης σε hashed μορφή (password_hash), τον ρόλο (role_id), την κατάσταση του λογαριασμού (status), τις ημερομηνίες δημιουργίας (created_at), επιβεβαίωσης email (email_verified_at) και έγκρισης (approved_at), καθώς και τον χρήστη που ενέκρινε τον λογαριασμό (approved_by). Επίσης αποθηκεύεται το όριο αποθήκευσης αρχείων (quota_limit_bytes) και τρία προαιρετικά κατώφλια ανίχνευσης (threshold_hi, threshold_lo, threshold_ai_par) που μπορεί να ρυθμίσει ο χρήστης.

#	Name	Type
☐ 1	id 	int(11)
☐ 2	email 	varchar(255)
☐ 3	password_hash	varchar(255)
☐ 4	role_id 	int(11)
☐ 5	created_at	datetime
☐ 6	status	varchar(255)
☐ 7	email_verified_at	datetime
☐ 8	approved_at	datetime
☐ 9	approved_by	int(11)
☐ 10	quota_limit_bytes	bigint(20)
☐ 11	threshold_hi	float
☐ 12	threshold_lo	float
☐ 13	threshold_ai_par	float

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	id
ix_users_email	BTREE	Yes	No	email
ix_users_id	BTREE	No	No	id
ix_users_role_id	BTREE	No	No	role_id

Πίνακας otp_codes

Αποθηκεύει τους κωδικούς μιας χρήσης (OTP) που αποστέλλονται κατά την εγγραφή ή την επαναφορά κωδικού. Κάθε εγγραφή συνδέεται με ένα email και περιλαμβάνει τον κωδικό σε hashed μορφή (code_hash), τον σκοπό χρήσης (purpose), την ημερομηνία λήξης (expires_at), τον αριθμό αποτυχημένων προσπαθειών (attempts) και την ημερομηνία χρήσης (consumed_at). Η αποθήκευση του κωδικού σε hashed μορφή αντί σε απλό κείμενο γίνεται για λόγους ασφάλειας.

#	Name	Type
1	id 	int(11)
2	email 	varchar(255)
3	purpose	varchar(255)
4	code_hash	varchar(255)
5	expires_at	datetime
6	attempts	int(11)
7	consumed_at	datetime
8	created_at	datetime

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	id
ix_otp_codes_email	BTREE	No	No	email
ix_otp_codes_id	BTREE	No	No	id

Πίνακας courses

Αποθηκεύει τα μαθήματα. Κάθε μάθημα συνδέεται με τον καθηγητή που το δημιούργησε (teacher_id) και περιλαμβάνει τον κωδικό μαθήματος (code), τον τίτλο (title) και την ημερομηνία δημιουργίας (created_at).

#	Name	Type
1	id 	int(11)
2	teacher_id 	int(11)
3	code	varchar(255)
4	title	varchar(255)
5	created_at	datetime

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	id
ix_courses_id	BTREE	No	No	id
ix_courses_teacher_id	BTREE	No	No	teacher_id

Πίνακας course_years

Αποθηκεύει τα εξάμηνα του κάθε μαθήματος. Κάθε εγγραφή συνδέεται με ένα μάθημα (course_id) και περιλαμβάνει το ακαδημαϊκό εξάμηνο (academic_year, π.χ. Spring 2026) και την ημερομηνία δημιουργίας. Ο πίνακας αυτός επιτρέπει στον καθηγητή να διαχωρίζει τις εργασίες ανά εξάμηνο χωρίς να χρειάζεται να δημιουργεί νέο μάθημα κάθε φορά.

#	Name	Type
1	id 	int(11)
2	course_id 	int(11)
3	academic_year	varchar(255)
4	created_at	datetime

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	id
ix_course_years_course_id	BTREE	No	No	course_id
ix_course_years_id	BTREE	No	No	id

Πίνακας deliverables

Αποθηκεύει τα παραδοτέα. Κάθε παραδοτέο ανήκει σε ένα ακαδημαϊκό εξάμηνο (course_year_id) και σε έναν καθηγητή (teacher_id). Τα πεδία περιλαμβάνουν τον τίτλο (title), την προθεσμία υποβολής (due_date) και την ημερομηνία δημιουργίας (created_at).

#	Name	Type
1	id 	int(11)
2	course_year_id 	int(11)
3	teacher_id 	int(11)
4	title	varchar(255)
5	due_date	datetime
6	created_at	datetime

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	id
ix_deliverables_course_year_id	BTREE	No	No	course_year_id
ix_deliverables_id	BTREE	No	No	id
ix_deliverables_teacher_id	BTREE	No	No	teacher_id

Πίνακας groups

Αποθηκεύει τις ομάδες φοιτητών μέσα σε κάθε παραδοτέο. Κάθε ομάδα συνδέεται με ένα παραδοτέο (deliverable_id) και έναν καθηγητή (teacher_id), και έχει ένα όνομα (name) που προέρχεται από τον φάκελο μέσα στο ZIP αρχείο κατά τη μαζική υποβολή.

#	Name	Type
1	id 	int(11)
2	deliverable_id 	int(11)
3	teacher_id 	int(11)
4	name	varchar(255)
5	created_at	datetime

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	id
ix_groups_deliverable_id	BTREE	No	No	deliverable_id
ix_groups_id	BTREE	No	No	id
ix_groups_teacher_id	BTREE	No	No	teacher_id

Πίνακας group_members

Αποθηκεύει τα μέλη κάθε ομάδας. Κάθε εγγραφή συνδέεται με μια ομάδα (group_id) και περιλαμβάνει ένα αναγνωριστικό φοιτητή (student_identifier) και την ημερομηνία προσθήκης (added_at). Το πεδίο student_identifier είναι varchar και όχι foreign key προς τον πίνακα users, γιατί στην τρέχουσα έκδοση οι φοιτητές δεν έχουν λογαριασμό στο σύστημα. Το αναγνωριστικό εισάγεται από τον καθηγητή για σκοπούς οργάνωσης.

#	Name	Type
1	group_id 	int(11)
2	student_identifier 	varchar(255)
3	added_at	datetime

Πίνακας submissions

Αποθηκεύει τα αρχεία που ανεβαίνουν στο σύστημα. Κάθε υποβολή συνδέεται με μια ομάδα (group_id), ένα παραδοτέο (deliverable_id) και τον χρήστη που την ανέβασε (owner_id). Τα

Βασικά πεδία περιλαμβάνουν το αρχικό όνομα αρχείου (original_filename), τη διαδρομή αποθήκευσης (stored_path), τον τύπο αρχείου (file_type) και το μέγεθος (file_size).

Για σκοπούς οργάνωσης, αποθηκεύονται επίσης το μάθημα (course), η εργασία (assignment), το αναγνωριστικό φοιτητή (student_id), ετικέτες (tags) και σημειώσεις (notes).

Τα αποτελέσματα ανάλυσης αποθηκεύονται απευθείας στον ίδιο πίνακα. Τα σχετικά πεδία είναι η κατάσταση ανάλυσης (ai_check_status), οι ημερομηνίες αίτησης και ολοκλήρωσης (ai_requested_at, ai_checked_at), ο βαθμός αξιοπιστίας (trustworthiness), το ποσοστό πιθανής χρήσης AI (percentage_ai), η τελική κατηγοριοποίηση (final_label), μια συνοπτική αναφορά σε JSON μορφή (summary_json) και η αναλυτική ανάλυση ανά παράγραφο (paragraphs_json). Σε περίπτωση αποτυχίας της ανάλυσης, αποθηκεύεται το σφάλμα στο πεδίο ai_check_error.

#	Name	Type
1	id	int(11)
2	owner_id	int(11)
3	deliverable_id	int(11)
4	group_id	int(11)
5	original_filename	varchar(255)
6	stored_path	varchar(255)
7	file_type	varchar(255)
8	file_size	int(11)
9	course	varchar(255)
10	assignment	varchar(255)
11	student_id	varchar(255)
12	tags	varchar(255)
13	final_label	varchar(255)
14	trustworthiness	int(11)
15	percentage_ai	float
16	summary_json	longtext
17	paragraphs_json	longtext
18	ai_check_status	varchar(255)
19	ai_requested_at	datetime
20	ai_checked_at	datetime
21	ai_check_error	text
22	notes	text
23	created_at	datetime

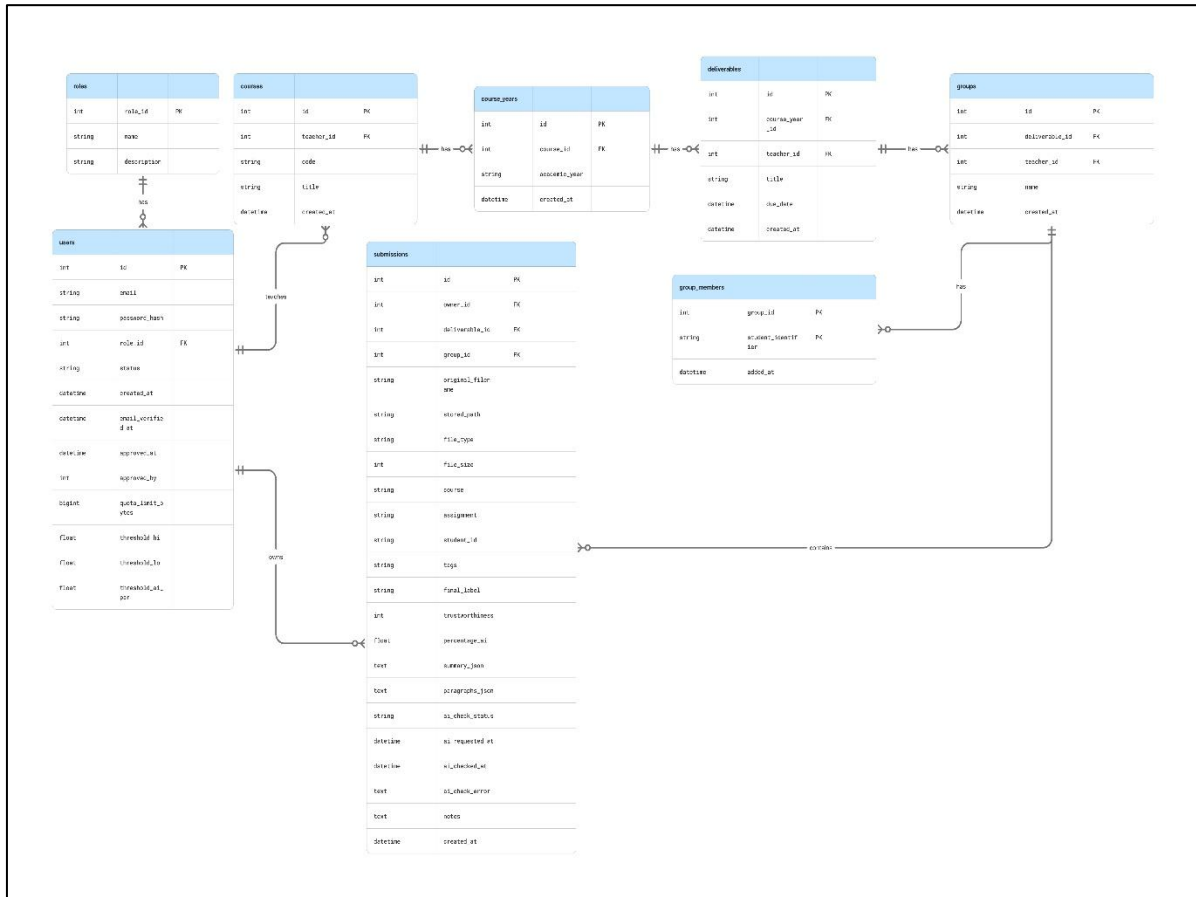
Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	id
ix_submissions_deliverable_id	BTREE	No	No	deliverable_id
ix_submissions_group_id	BTREE	No	No	group_id
ix_submissions_id	BTREE	No	No	id
ix_submissions_owner_id	BTREE	No	No	owner_id

4.4.2 Διάγραμμα Οντοτήτων-Σχέσεων

Οι σχέσεις μεταξύ των οντοτήτων είναι:

- Ένας χρήστης (teacher) μπορεί να έχει πολλά μαθήματα (1:N).
- Ένα μάθημα μπορεί να έχει πολλά ακαδημαϊκά εξάμηνα (1:N).
- Ένα ακαδημαϊκό εξάμηνο μπορεί να έχει πολλά παραδοτέα (1:N).
- Ένα παραδοτέο μπορεί να έχει πολλές ομάδες (1:N).
- Μία ομάδα μπορεί να έχει πολλές υποβολές (1:N).
- Μία ομάδα μπορεί να έχει πολλά μέλη (1:N μέσω group_members).
- Ένας χρήστης ανήκει σε έναν ρόλο (N:1).

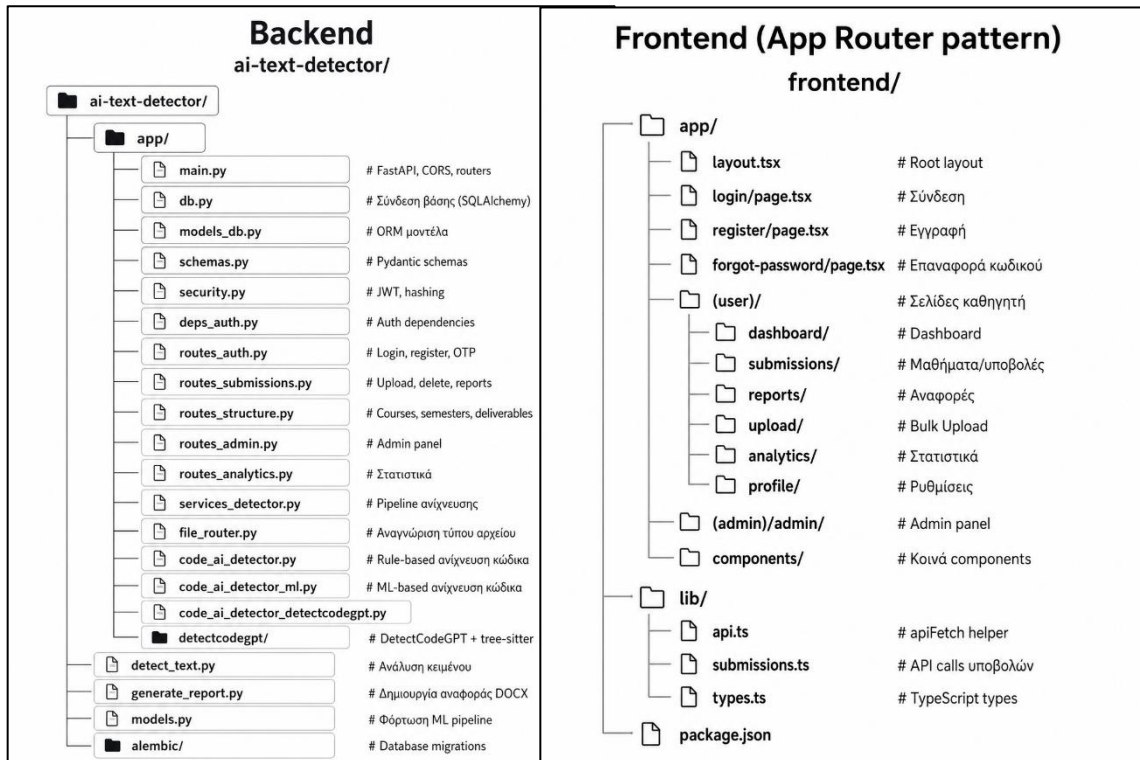
Τα διαγράμματα οντοτήτων σχέσεων (ERD) χρησιμοποιούνται για την εννοιολογική αναπαράσταση δεδομένων. Στο σχήμα 4 παρουσιάζεται το διάγραμμα οντοτήτων σχέσεων της βάσης δεδομένων.



Σχήμα 4 - Διάγραμμα Οντοτήτων Σχέσεων

4.5 Δομή Αρχείων

Ο κώδικας οργανώνεται σε δύο κύριους φακέλους: ai-text-detector (backend) και frontend.



4.6 Βιβλιοθήκες

4.6.1 Backend (Python)

- Framework: fastapi, uvicorn, starlette, pydantic [22].
- Βάση Δεδομένων: SQLAlchemy, alembic, PyMySQL.
- Ανάγνωση Αρχείων: python-docx, pdfminer.six, pdfplumber.
- ML/AI: transformers, tokenizers, huggingface_hub, safetensors.
- Κλασικό ML: scikit-learn, scipy, numpy, pandas, nltk.

4.6.2 Frontend

- next (v16.1.1),
- react (v19.2.3),
- react-dom.

Κεφάλαιο 5

5. Έλεγχος Λειτουργικότητας του Συστήματος

5.1 Μεθοδολογία Ελέγχου	53
5.2 Δοκιμές ανά Λειτουργία	54
5.3 Συνοπτικά Αποτελέσματα	58
5.4 Αξιολόγηση Ευχρηστίας	59

5.1 Μεθοδολογία Ελέγχου

Ο έλεγχος του συστήματος έγινε χειροκίνητα, κατά τη διάρκεια της ανάπτυξης αλλά και μετά την εγκατάσταση στον server του Τμήματος. Με κάθε νέα λειτουργία, το σύστημα δοκιμαζόταν αμέσως μέσω browser, όπως θα τη χρησιμοποιούσε ένας πραγματικός χρήστης. Ο έλεγχος βασίστηκε αποκλειστικά σε manual testing και για αυτό δεν χρησιμοποιήθηκαν αυτοματοποιημένα tests.

Ο έλεγχος έγινε με διάφορα σενάρια χρήσης του συστήματος. Για κάθε λειτουργία, δηλαδή, υπήρχε ένα σενάριο που αντιστοιχούσε σε κάτι που θα έκανε ένας καθηγητής στην πράξη.

- Δημιουργία μαθήματος,
- Προσθήκη εξαμήνου,
- Ανέβασμα αρχείων,
- Εκτέλεση ανάλυσης και
- Προβολή αποτελεσμάτων

Πέρα όμως από την κανονική χρήση, δοκιμάστηκαν και ακραίες περιπτώσεις, όπως κενά αρχεία και λάθος τύποι αρχείων. Αυτό βοήθησε στο να επιβεβαιωθεί ότι το σύστημα δεν καταρρέει αλλά εμφανίζει τα κατάλληλα μηνύματα σφάλματος.

Για τις δοκιμές ετοιμάστηκαν αρχεία ειδικά για αυτόν τον σκοπό. Στα αρχεία κειμένου υπήρχαν τρεις κατηγορίες:

- αρχεία γραμμένα εξ ολοκλήρου από άνθρωπο,
- αρχεία παραγμένα εξ ολοκλήρου από AI (μέσω ChatGPT) και
- αρχεία μικτού περιεχομένου, δηλαδή κείμενο που είχε μερικές ενδείξεις ότι κάποιες παράγραφοι αντικαταστάθηκαν με AI-generated.

Αντίστοιχα, για τον κώδικα ετοιμάστηκαν αρχεία σε Python, Java και JavaScript, τόσο γραμμένα χειροκίνητα όσο και παραγμένα μέσω εργαλείων AI (ChatGPT, GitHub Copilot).

5.2 Δοκιμές ανά Λειτουργία

5.2.1 Εγγραφή και Σύνδεση

Ο πρώτος έλεγχος έγινε στην βασική ροή του συστήματος, δηλαδή την εγγραφή νέου χρήστη, τη λήψη OTP στο email, την εισαγωγή κωδικού, την αναμονή για έγκριση από τον διαχειριστή, και τέλος την σύνδεση του χρήστη στο σύστημα. Ελέγχθηκε επίσης τι συμβαίνει αν εισαχθεί λανθασμένος κωδικός OTP, αν ο χρήστης προσπαθήσει να συνδεθεί πριν τον εγκρίνει ο διαχειριστής και αν καταχωρηθεί λάθος κωδικός.

Η ροή λειτούργησε κανονικά σε όλες τις περιπτώσεις. Το email με τον κωδικό OTP φτάνει μέσα σε λίγα δευτερόλεπτα, τα μηνύματα σφάλματος είναι κατανοητά, και μετά την έγκριση η σύνδεση γίνεται χωρίς πρόβλημα.

5.2.2 Μαθήματα, Εξάμηνα και Παραδοτέα

Δημιουργήθηκαν δοκιμαστικά μαθήματα με κωδικούς και τίτλους (π.χ. EPL425 - Internet Technologies), προστέθηκαν ακαδημαϊκά εξάμηνα (Spring 2025, Spring 2026) και μέσα σε αυτά δημιουργήθηκαν παραδοτέα. Δοκιμάστηκε και η διαγραφή μαθήματος, για να επιβεβαιωθεί ότι σβήνονται μαζί τα εξάμηνα, τα παραδοτέα, οι ομάδες και τα αρχεία που

ανήκουν σε αυτό. Η ιεραρχία δημιουργείται σωστά και τα dropdown φίλτρα ενημερώνονται αμέσως μετά τη δημιουργία νέων εγγραφών.

5.2.3 Μαζική Υποβολή μέσω ZIP

Δημιουργήθηκαν ZIP αρχεία με τη σωστή δομή (ένας φάκελος ανά ομάδα, μέσα τα αρχεία), και δοκιμάστηκε η μαζική υποβολή. Ελέγχθηκε αν οι ομάδες δημιουργούνται αυτόματα βάσει των ονομάτων φακέλων, αν τα αρχεία κατανέμονται σωστά, αν η ενότητα Result εμφανίζει τα σωστά νούμερα, και αν η αυτόματη ανάλυση κατά το upload λειτουργεί.

Δοκιμάστηκαν και ακραίες περιπτώσεις. Όπως, ZIP χωρίς φακέλους (flat), ZIP με κενούς φακέλους, ZIP με αρχεία που δεν υποστηρίζονται, και αρκετά μεγάλα ZIP.

Η κανονική ροή δούλεψε χωρίς πρόβλημα. Το σύστημα δημιουργεί τις ομάδες σωστά και στο Result φαίνεται ακριβώς πόσα αρχεία ανέβηκαν, πόσες ομάδες δημιουργήθηκαν και αν κάτι απέτυχε.

5.2.4 Ανάλυση Αρχείων Κειμένου

Η ανάλυση των αρχείων δοκιμάστηκε σε τρεις κατηγορίες αρχείων:

- Ανθρώπινα,
- AI-Generated
- Μικτά

Τα ανθρώπινα αρχεία αναγνωρίστηκαν σωστά στην πλειονότητα των περιπτώσεων. Τα αρχεία AI πήραν ετικέτα AI ή UNCERTAIN ανάλογα με τη δομή τους. Τα μικτά αρχεία παρουσίασαν ποσοστά που αντικατοπτρίζουν λογικά τη σύνθεσή τους. Δηλαδή αν το μισό κείμενο ήταν AI, το ποσοστό κυμαινόταν γύρω στο 50%.

Η ανάλυση σε επίπεδο παραγράφου λειτουργεί σωστά. Η δομή του εγγράφου (Document Structure) αντικατοπτρίζει τις πραγματικές επικεφαλίδες του αρχείου, και κάθε παράγραφος χρωματίζεται ανάλογα με το score. Φυσικά, αξίζει να σημειωθεί ότι μπορεί να υπάρξουν ψευδώς θετικά ή ψευδώς αρνητικά αποτελέσματα. Κανένα σύστημα ανίχνευσης δεν είναι αλάνθαστο.

5.2.5 Ανάλυση Αρχείων Κώδικα

Δοκιμάστηκαν αρχεία σε Python, Java και JavaScript , τόσο ανθρώπινα όσο και AI-generated.

Το σύστημα αναγνωρίζει σωστά τον τύπο και δοκιμάζει πρώτα το DetectCodeGPT. Αν αποτύχει (π.χ. μη υποστηριζόμενη γλώσσα ή πρόβλημα κατά την εκτέλεση), περνάει αυτόματα στον υβριδικό detector ως fallback. Η αναφορά κώδικα εμφανίζει σωστά τις class probabilities, τα signals και τα detected patterns. Η προεπισκόπηση κώδικα με syntax highlighting δουλεύει κανονικά.

Τα αποτελέσματα για κώδικα πρέπει να ερμηνεύονται με προσοχή. Η ανίχνευση AI-generated κώδικα βρίσκεται ακόμα σε πρώιμο ερευνητικό στάδιο, και τα αποτελέσματα δεν είναι πάντα τόσο ξεκάθαρα όσο στα αρχεία κειμένου.

5.2.6 Αναφορές Ανάλυσης

Ελέγχθηκε η εμφάνιση αναφορών και για τους δύο τύπους αρχείων. Στα αρχεία κειμένου δοκιμάστηκε το summary, η δένδρική δομή εγγράφου, η χρωματική κωδικοποίηση παραγράφων (κόκκινο για AI, πράσινο για human) και η προεπισκόπηση PDF. Στα αρχεία κώδικα δοκιμάστηκαν τα code signals, οι πιθανότητες ανά κατηγορία και η code preview.

Και οι δύο τύποι αναφορών εμφανίζονται σωστά. Η προεπισκόπηση PDF δημιουργείται αυτόματα στον server μέσω LibreOffice και φορτώνει κανονικά στο δεξί μέρος της οθόνης.

5.2.7 Στατιστικά και Analytics

Δοκιμάστηκε η σελίδα Analytics με δεδομένα από πολλαπλά μαθήματα και παραδοτέα. Ελέγχθηκε η εναλλαγή μεταξύ AI και HUMAN, τα αριθμητικά στοιχεία του πίνακα (μέσος όρος, διάμεσος, min, max), τα φίλτρα και τα boxplot γραφήματα.

Τα στατιστικά υπολογίζονται σωστά. Τα φίλτρα συνδυάζονται χωρίς πρόβλημα. Η χρωματική κωδικοποίηση των boxplot (πράσινο κάτω από 15%, πορτοκαλί 15-40%, κόκκινο πάνω από 40%) αντικατοπτρίζει σωστά τα επίπεδα.

5.2.8 Dashboard Διαχειριστή

Δοκιμάστηκε η έγκριση χρηστών, η εμφάνιση storage ανά χρήστη, και η λειτουργία φιλτραρίσματος (All / Pending / Approved). Ελέγχθηκε επίσης ότι ο admin δεν βλέπει μαθήματα και ο teacher δεν μπορεί να μπει στο admin panel.

Ο διαχωρισμός ρόλων λειτουργεί σωστά. Κάθε χρήστης βλέπει μόνο ό,τι αντιστοιχεί στον ρόλο του.

5.2.9 Ρυθμίσεις Κατωφλίων

Δοκιμάστηκε η αλλαγή των κατωφλίων ανίχνευσης μέσω της σελίδας Profile του χρήστη. Στη συνέχεια, πραγματοποιήθηκε νέα ανάλυση σε υπάρχοντα αρχεία, ώστε να ελεγχθεί αν εφαρμόζονται σωστά οι νέες τιμές. Οι αλλαγές αποθηκεύονται επιτυχώς και επηρεάζουν τις μελλοντικές αναλύσεις, ενώ η λειτουργία επαναφοράς στις προεπιλεγμένες τιμές εκτελείται κανονικά.

5.3 Συνοπτικά Αποτελέσματα

Ο Πίνακας 1 συγκεντρώνει όλα τα αποτελέσματα των ελέγχων που έγιναν στο σύστημα.

Λειτουργία	Κατάσταση
Εγγραφή και σύνδεση	Επιτυχής
Μηχανισμός OTP	Επιτυχής
Έγκριση χρηστών (admin)	Επιτυχής
Δημιουργία μαθημάτων	Επιτυχής
Δημιουργία εξαμήνων	Επιτυχής
Δημιουργία παραδοτέων	Επιτυχής
Μαζική υποβολή (ZIP)	Επιτυχής
Ανάλυση DOCX	Επιτυχής
Ανάλυση κώδικα	Επιτυχής
Fallback detector	Επιτυχής
Αναφορά DOCX	Επιτυχής
Αναφορά κώδικα	Επιτυχής
PDF preview	Επιτυχής
Στατιστικά / Analytics	Επιτυχής
Boxplot γραφήματα	Επιτυχής
Dashboard admin	Επιτυχής
Διαχωρισμός ρόλων	Επιτυχής
Κατώφλια ανίχνευσης	Επιτυχής
Αλλαγή κωδικού	Επιτυχής
Μη υποστηριζόμενα αρχεία	Επιτυχής
Χρήση αποθηκευτικού χώρου	Επιτυχής

Όλες οι λειτουργίες ανταποκρίθηκαν στις απαιτήσεις. Τα σφάλματα που εντοπίστηκαν κατά την ανάπτυξη διορθώθηκαν πριν την τελική εγκατάσταση στον server.

5.4 Αξιολόγηση Ευχρηστίας

Για την αξιολόγηση της ευχρηστίας ετοιμάστηκε ερωτηματολόγιο που περιλαμβάνεται στο Παράρτημα Α. Το ερωτηματολόγιο απευθύνεται σε καθηγητές που θα χρησιμοποιήσουν το σύστημα στην πράξη, και καλύπτει θέματα όπως η ευκολία πλοήγησης, η δημιουργία μαθημάτων και παραδοτέων, η σαφήνεια παρουσίασης των αποτελεσμάτων, η χρησιμότητα ανίχνευσης κειμένου και κώδικα, η οργάνωσης των εργασιών σε μαθήματα, εξάμηνα, παραδοτέα και ομάδες, και η εμπιστοσύνη στα αποτελέσματα ως βοηθητικές ενδείξεις. Οι απαντήσεις δίνονται σε κλίμακα 1-5, όπου 1 αντιστοιχεί σε αρνητική αξιολόγηση και 5 σε θετική.

Βάσει feedback από ένα μέλος του ακαδημαϊκού προσωπικού που δοκίμασε το σύστημα σε πραγματικές συνθήκες, καταγράφηκαν οι παρακάτω βαθμολογίες. Δεδομένου ότι πρόκειται για αξιολόγηση από έναν μόνο χρήστη, τα αποτελέσματα δεν επιτρέπουν στατιστική γενίκευση, αλλά αποτελούν μια πρώτη ποιοτική ένδειξη για τη χρηστικότητα της πλατφόρμας.

Πίνακας 2 - Αποτελέσματα ερωτηματολογίου

Ερώτηση	Βαθμολογία
Ευκολία πλοήγησης	5/5
Ευκολία δημιουργίας μαθήματος και παραδοτέου	5/5
Χρησιμότητα βίντεο επίδειξης και εγχειριδίου χρήστη	4/5
Χρησιμότητα ανίχνευσης κειμένου (DOCX)	4/5
Χρησιμότητα ανίχνευσης πηγαίου κώδικα	5/5
Σαφήνεια παρουσίασης αποτελεσμάτων ανάλυσης	4/5

Ερώτηση	Βαθμολογία
Πρακτικότητα οργάνωσης (μαθήματα→εξάμηνα→παραδοτέα→ομάδες)	4/5
Καταλληλότητα προεπιλεγμένων κατωφλίων ανίχνευσης	4/5
Σημασία δυνατότητας ρύθμισης κατωφλίων	4/5
Εμπιστοσύνη στα αποτελέσματα ως ένδειξη (όχι απόδειξη)	4/5
Πιθανότητα χρήσης σε μάθημα που διδάσκει ο καθηγητής	4/5

Πέρα από τις βαθμολογίες, καταγράφηκαν και ορισμένα ποιοτικά σχόλια που αξίζει να αναφερθούν. Συγκεκριμένα, καταγράφηκε ότι τα αποτελέσματα θα πρέπει να αντιμετωπίζονται ως αφορμή για χειροκίνητο έλεγχο, αντί να θεωρούνται οριστική κατηγοριοποίηση. Η παρατήρηση αυτή ευθυγραμμίζεται με τη φιλοσοφία σχεδίασης του συστήματος, το οποίο παρουσιάζει τα αποτελέσματα ως ενδείξεις και όχι ως τελικές αποφάσεις. Επιπλέον, αναφέρθηκε ότι τα αποτελέσματα μπορούν να αξιοποιηθούν ως αφετηρία για περαιτέρω διερεύνηση, όπως για παράδειγμα την συζήτηση του καθηγητή με τον φοιτητή.

Ως προτεινόμενη βελτίωση, καταγράφηκε η δυνατότητα σύγκρισης μεταξύ ομάδων φοιτητών ή ακαδημαϊκών ετών, λειτουργία που θα μπορούσε να προστεθεί στη σελίδα Analytics σε μελλοντική έκδοση. Επιπλέον, σημειώθηκε ένα τεχνικό ζήτημα κατά τη μαζική υποβολή αρχείων κώδικα, και συγκεκριμένα σφάλμα proxy 502, το οποίο οφειλόταν στο ότι ο Apache reverse proxy έκλεινε τη σύνδεση πριν ολοκληρωθεί η ανάλυση στο backend. Η ανάλυση κώδικα μέσω DetectCodeGPT εκτελείται σε CPU και μπορεί να διαρκέσει αρκετά λεπτά, ειδικά σε μαζικές υποβολές με πολλά αρχεία. Η αρχική τιμή του ProxyTimeout ήταν 300 δευτερόλεπτα (5 λεπτά), η οποία δεν επαρκούσε. Το πρόβλημα αντιμετωπίστηκε με αύξηση του ProxyTimeout στα 600 δευτερόλεπτα (10 λεπτά) στο αρχείο ρύθμισης του Apache (/etc/httpd/conf.d/ai-detection.conf), ακολουθούμενη από επανεκκίνηση της υπηρεσίας. Επίσης επισημάνθηκε η ανάγκη για σαφέστερες οδηγίες σχετικά με τη δομή του αρχείου ZIP κατά τη μαζική υποβολή.

Συνολικά, οι βαθμολογίες κυμαίνονται μεταξύ 4 και 5, ενώ τα σχόλια δείχνουν ότι η πλατφόρμα γίνεται αντιληπτή ως χρήσιμο και υποστηρικτικό εργαλείο. Βέβαια, τα αποτελέσματα αυτά προέρχονται από έναν μόνο χρήστη και δεν αρκούν για γενικεύσεις. Μια πιο ολοκληρωμένη εικόνα θα μπορούσε να προκύψει από αξιολόγηση με περισσότερους συμμετέχοντες, σε πραγματικές συνθήκες χρήσης κατά τη διάρκεια ενός ακαδημαϊκού εξαμήνου.

Κεφάλαιο 6

6. Συζήτηση και Μελλοντικές Προεκτάσεις

6.1 Συζήτηση	62
6.2 Μελλοντικές Προεκτάσεις	64

6.1 Συζήτηση

Η παρούσα εργασία είχε ως στόχο τη δημιουργία ενός συστήματος που συνδυάζει τη διαχείριση ακαδημαϊκών εργασιών με την ανάλυση του περιεχομένου τους ως προς πιθανή χρήση εργαλείων Τεχνητής Νοημοσύνης. Το σύστημα αναπτύχθηκε, εγκαταστάθηκε σε πραγματικό server του Τμήματος και δοκιμάστηκε με ποικιλία αρχείων. Στην ενότητα αυτή γίνεται μια αποτίμηση του τι επιτεύχθηκε, τι λειτούργησε σωστά, τι θα μπορούσε να γίνει καλύτερα και ποιες ερωτήσεις παραμένουν ανοιχτές.

Πριν από την αποτίμηση των αποτελεσμάτων, είναι σημαντικό να σημειωθεί ότι η παρούσα εργασία δεν αντιμετωπίζει τα εργαλεία Τεχνητής Νοημοσύνης ως κάτι απαραίτητα αρνητικό στην εκπαίδευση. Όταν χρησιμοποιούνται με μέτρο, διαφάνεια και σύμφωνα με τους κανόνες του μαθήματος, μπορούν να λειτουργήσουν υποστηρικτικά για τους φοιτητές, βοηθώντας τους στην κατανόηση δύσκολων εννοιών, στην οργάνωση των ιδεών τους και στη διερεύνηση πιθανών λύσεων σε τεχνικά ή προγραμματιστικά προβλήματα. Το πρόβλημα προκύπτει όταν η χρήση τους αντικαθιστά την προσωπική προσπάθεια του φοιτητή. Για τον λόγο αυτό, το σύστημα που αναπτύχθηκε δεν έχει ως στόχο την τιμωρία της χρήσης AI, αλλά την παροχή βοηθητικών ενδείξεων που μπορούν να στηρίξουν τον καθηγητή σε μια πιο τεκμηριωμένη αξιολόγηση.

Ξεκινώντας από τα θετικά, το σύστημα καλύπτει ολόκληρη τη ροή εργασίας ενός καθηγητή. Από τη δημιουργία μαθήματος μέχρι τη μαζική υποβολή αρχείων και την προβολή αναλυτικών αποτελεσμάτων. Η αρχιτεκτονική του συστήματος είναι καθαρή. Frontend και Backend είναι πλήρως διαχωρισμένα, η βάση δεδομένων αντικατοπτρίζει τη φυσική ιεραρχία μαθημάτων, εξαμήνων και παραδοτέων, και οι μηχανισμοί ανάλυσης λειτουργούν ως ανεξάρτητα modules που μπορούν να αντικατασταθούν ή να επεκταθούν στο μέλλον. Η δυνατότητα μαζικής υποβολής μέσω ZIP και η αυτόματη δημιουργία ομάδων εξοικονομούν σημαντικό χρόνο σε σχέση με την χειροκίνητη καταχώρηση.

Η ανάλυση κειμένου σε επίπεδο παραγράφου αποτελεί ένα από τα πιο χρήσιμα χαρακτηριστικά. Αντί να δίνει απλώς μια ετικέτα για ολόκληρο το έγγραφο, το σύστημα δείχνει ακριβώς ποιες παράγραφοι θεωρούνται ύποπτες, σε ποια ενότητα του εγγράφου βρίσκονται και με ποιο score. Αυτό δίνει στον καθηγητή τη δυνατότητα να εστιάσει σε συγκεκριμένα σημεία αντί να αντιμετωπίζει ολόκληρο το κείμενο ως ύποπτο ή μη.

Ωστόσο, υπάρχουν σημαντικά σημεία που πρέπει να αναφερθούν.

Το σημαντικότερο σημείο αφορά τον τρόπο με τον οποίο πρέπει να ερμηνεύονται τα αποτελέσματα της ανάλυσης. Τα αποτελέσματα του συστήματος δεν μπορούν να θεωρηθούν ως απόλυτη απόδειξη, αλλά ως ενδείξεις που βοηθούν τον καθηγητή στην αξιολόγηση των εργασιών. Υπάρχει πάντοτε η πιθανότητα ένα αρχείο να χαρακτηριστεί λανθασμένα ως AI-generated, ενώ στην πραγματικότητα έχει γραφτεί από άνθρωπο, ή αντίστροφα. Ο περιορισμός αυτός δεν αφορά μόνο το συγκεκριμένο σύστημα, αλλά γενικότερα τις μεθόδους ανίχνευσης AI-generated περιεχομένου, όπως αυτές υπάρχουν σήμερα [11, 12, 18]. Για τον λόγο αυτό, η πλατφόρμα δεν σχεδιάστηκε με σκοπό να αντικαταστήσει την κρίση του καθηγητή, αλλά να λειτουργεί ως ένα υποστηρικτικό εργαλείο. Η τελική απόφαση παραμένει πάντοτε στην ευθύνη του ίδιου του καθηγητή.

Ένα δεύτερο σημείο αφορά την ανάλυση κώδικα. Ενώ η ανάλυση DOCX λειτουργεί αρκετά αξιόπιστα για αγγλικό κείμενο, η ανίχνευση AI-generated κώδικα βρίσκεται σε πολύ πιο πρώιμο στάδιο. Το DetectCodeGPT είναι ερευνητικό εργαλείο που δεν έχει δοκιμαστεί

εκτενώς σε κάθε γλώσσα και κάθε τύπο κώδικα. Ο υβριδικός detector που χρησιμοποιείται ως fallback (40% rule-based, 60% ML) δίνει μια εκτίμηση, αλλά τα αποτελέσματα πρέπει να ερμηνεύονται με ακόμα μεγαλύτερη επιφύλαξη σε σχέση με τα αρχεία κειμένου.

Ένα τρίτο σημείο αφορά τη γλώσσα του κειμένου. Το μοντέλο ανίχνευσης έχει εκπαιδευτεί σε αγγλικά κείμενα, επομένως η απόδοσή του σε ελληνικά κείμενα μπορεί να μην είναι το ίδιο αξιόπιστη. Στην παρούσα φάση δεν έχει γίνει συστηματική αξιολόγηση που να συγκρίνει την απόδοση του μοντέλου ανάμεσα σε αγγλικά και ελληνικά κείμενα. Επομένως, το θέμα αυτό παραμένει ανοιχτό και θα μπορούσε να εξεταστεί πιο αναλυτικά στο μέλλον.

Τέλος, αξίζει να σημειωθεί ότι τα εργαλεία Τεχνητής Νοημοσύνης εξελίσσονται συνεχώς και με γρήγορους ρυθμούς. Έτσι, ένα μοντέλο ανίχνευσης που σήμερα μπορεί να λειτουργεί ικανοποιητικά, δεν σημαίνει απαραίτητα ότι θα παραμείνει το ίδιο αποτελεσματικό και στο μέλλον [19]. Καθώς αυτά τα εργαλεία βελτιώνονται, γίνεται πιο δύσκολο να εντοπιστεί με βεβαιότητα αν ένα κείμενο έχει παραχθεί από άνθρωπο ή από ΑΙ. Για τον λόγο αυτό, η αξιοπιστία ενός τέτοιου συστήματος χρειάζεται να επανεξετάζεται με την πάροδο του χρόνου και να μην θεωρείται δεδομένη.

6.2 Μελλοντικές Προεκτάσεις

Η διαδικασία ανάπτυξης και αξιολόγησης του συστήματος ανέδειξε ορισμένες κατευθύνσεις που θα μπορούσαν να συμβάλουν στην περαιτέρω βελτίωση και εξέλιξή του.

Ρόλος Φοιτητή

Στην παρούσα έκδοση του συστήματος, τα αρχεία ανεβαίνουν αποκλειστικά από τον καθηγητή. Μια φυσική επέκταση θα ήταν η προσθήκη ρόλου φοιτητή, ώστε οι φοιτητές να μπορούν να υποβάλλουν απευθείας τις εργασίες τους μέσω της πλατφόρμας. Δεδομένου ότι ο σχεδιασμός του συστήματος βασίζεται ήδη στη διάκριση ρόλων, η προσθήκη ενός επιπλέον τύπου χρήστη θα μπορούσε να ενσωματωθεί χωρίς ουσιαστική αλλαγή στη

συνολική δομή του συστήματος. Κυρίως, θα χρειαζόταν νέα σελίδα στο frontend, νέο ξεχωριστό dashboard για τον φοιτητή και νέα προσαρμογή στα δικαιώματα πρόσβασης.

Υποστήριξη Περισσότερων Τύπων Αρχείων

Η ανάλυση κειμένου περιορίζεται προς το παρόν σε DOCX. Η προσθήκη υποστήριξης για PDF θα κάλυπτε μια σημαντική ανάγκη, καθώς πολλοί φοιτητές υποβάλλουν τις εργασίες τους σε αυτή τη μορφή.

Αναβάθμιση Μοντέλων Ανίχνευσης

Τα μοντέλα που χρησιμοποιούνται σήμερα αποτέλεσαν τη βάση πάνω στην οποία αναπτύχθηκε το σύστημα, χωρίς όμως να θεωρούνται μόνιμη λύση. Καθώς οι τεχνικές ανίχνευσης AI-generated περιεχομένου εξελίσσονται, είναι σημαντικό το σύστημα να μπορεί να προσαρμόζεται και να αξιοποιεί νεότερα ή πιο αποδοτικά μοντέλα. Για τον λόγο αυτό, η διαδικασία ανάλυσης έχει οργανωθεί σε ξεχωριστά modules, ώστε κάθε μέθοδος να μπορεί να αντικατασταθεί ή να επεκταθεί πιο εύκολα, χωρίς να επηρεάζεται η συνολική λειτουργία του συστήματος.

Αυτοματοποιημένα Tests

Η προσθήκη unit tests και integration tests θα βελτιώνει σημαντικά την αξιοπιστία κατά τη συντήρηση. Σε ένα σύστημα με τόσα modules, μια αλλαγή σε ένα σημείο μπορεί να επηρεάσει κάτι αλλού, και τα αυτοματοποιημένα tests θα εντόπιζαν τέτοια προβλήματα πριν φτάσουν στον χρήστη.

Αξιολόγηση Ακρίβειας με Benchmark Dataset

Ένα σημαντικό επόμενο βήμα θα ήταν η δημιουργία ενός συνόλου δεδομένων, δηλαδή αρχεία που είναι διαχωρισμένα σε ανθρώπινα και AI-generated. Με ένα τέτοιο σύνολο

δεδομένων, το σύστημα θα μπορούσε να αξιολογηθεί με πιο αντικειμενικό τρόπο, αφού θα υπήρχε βάση σύγκρισης με τα πραγματικά αποτελέσματα. Αυτό θα επέτρεπε τον υπολογισμό μετρικών όπως το precision και το recall, οι οποίες χρησιμοποιούνται συχνά για την αξιολόγηση μοντέλων ανίχνευσης. Επιπλέον, θα ήταν δυνατή η σύγκριση διαφορετικών μοντέλων κάτω από τις ίδιες συνθήκες, ώστε να φανεί ποια προσέγγιση αποδίδει καλύτερα.

Υποστήριξη Ελληνικής Γλώσσας

Τα μοντέλα ανίχνευσης κειμένου έχουν αναπτυχθεί κυρίως Αγγλικά. Επομένως, όταν το σύστημα χρησιμοποιείται για αρχεία γραμμένα στα Ελληνικά, τα αποτελέσματα μπορεί να επηρεάζονται και να χρειάζονται πιο προσεκτική ερμηνεία. Για τον λόγο αυτό, θα ήταν χρήσιμο να γίνει ξεχωριστός έλεγχος της απόδοσης στα Ελληνικά και, αν χρειαστεί, να προσαρμοστεί το σύστημα ανάλογα.

Ειδοποιήσεις και Ημερολόγιο

Θα μπορούσε να προστεθεί σύστημα ειδοποιήσεων που ενημερώνει τον καθηγητή όταν ολοκληρώνεται μια ανάλυση ή πλησιάζει η προθεσμία ενός παραδοτέου. Επίσης, η ενσωμάτωση ημερολογίου για τις ημερομηνίες παράδοσης θα βελτιώνει την οργάνωση.

Κλιμάκωση και Απόδοση

Ο server του Τμήματος δεν διαθέτει GPU, κάτι που επιβραδύνει σημαντικά την ανάλυση κώδικα μέσω DetectCodeGPT. Σε περιβάλλον με GPU η ανάλυση θα ήταν πολύ πιο γρήγορη. Επιπλέον, σε περίπτωση που η πλατφόρμα χρησιμοποιηθεί από πολλούς καθηγητές ταυτόχρονα, θα χρειαστεί σύστημα ουράς για τη διαχείριση πολλαπλών αναλύσεων, αντί να τρέχουν ως background tasks στο ίδιο process.

Κεφάλαιο 7

7. Συμπεράσματα

7.1 Συνοπτική Εκτίμηση

67

7.1 Συνοπτική Εκτίμηση

Η διπλωματική εργασία παρουσίασε τον σχεδιασμό και την υλοποίηση μιας διαδικτυακής πλατφόρμας για τη διαχείριση ακαδημαϊκών εργασιών και την ανίχνευση πιθανής χρήσης εργαλείων Τεχνητής Νοημοσύνης σε κείμενα και πηγαίο κώδικα. Η προτεινόμενη λύση ανταποκρίνεται στην ολοένα και αυξανόμενη ανάγκη των καθηγητών να διαθέτουν ένα οργανωμένο εργαλείο που τους βοηθά να διαχειρίζονται τις εργασίες των φοιτητών τους και παράλληλα να λαμβάνουν βοηθητικές ενδείξεις σχετικά με τη γνησιότητα του περιεχομένου. Η πλατφόρμα αυτή ανταποκρίνεται στην ανάγκη για ένα απλό, εύκολα προσβάσιμο σύστημα που συνδυάζει διαχείριση και ανάλυση σε ένα ενιαίο περιβάλλον.

Το βασικό πλεονέκτημα του συστήματος σε σχέση με τα υπάρχοντα εργαλεία ανίχνευσης ΑΙ είναι ότι δεν περιορίζεται στην ανάλυση αρχείων. Ο καθηγητής μπορεί μέσα από την ίδια εφαρμογή να δημιουργεί μαθήματα, να οργανώνει ακαδημαϊκά εξάμηνα και παραδοτέα, να ανεβάζει αρχεία μαζικά μέσω ZIP, να εκτελεί ανάλυση και να βλέπει τα αποτελέσματα. Η ανάλυση κειμένου γίνεται σε επίπεδο παραγράφου, δείχνοντας ακριβώς ποια σημεία ενός εγγράφου φαίνονται ύποπτα, ενώ για τον κώδικα χρησιμοποιείται συνδυασμός μεθόδων (DetectCodeGPT, rule-based και ML-based) ώστε να δίνεται πάντα κάποιο αποτέλεσμα. Τα αποτελέσματα παρουσιάζονται μέσα από αναλυτικές αναφορές ανά αρχείο και συγκεντρωτικά στατιστικά ανά παραδοτέο, δίνοντας στον καθηγητή μια ολοκληρωμένη εικόνα.

Η υλοποίηση εστιάζει στην απλότητα, την ευχρηστία και τη σαφήνεια παρουσίασης, έτσι ώστε η πλατφόρμα να μπορεί να χρησιμοποιηθεί από οποιονδήποτε καθηγητή χωρίς ιδιαίτερη τεχνική εξοικείωση. Η εφαρμογή λειτουργεί αποκλειστικά μέσω browser και δεν απαιτεί εγκατάσταση λογισμικού. Επιπλέον, η αρχιτεκτονική του συστήματος είναι σχεδιασμένη με τρόπο που επιτρέπει τη μελλοντική προσθήκη νέων ρόλων χρηστών, νέων τύπων αρχείων και νεότερων μοντέλων ανίχνευσης, χωρίς να χρειάζεται αναδιάρθρωση του υπόλοιπου συστήματος.

Η εγκατάσταση και λειτουργία της πλατφόρμας σε πραγματικό server του Τμήματος Πληροφορικής επιβεβαίωσε ότι το σύστημα δεν αποτελεί μόνο θεωρητική πρόταση αλλά μια λειτουργική εφαρμογή έτοιμη για χρήση σε πραγματικό ακαδημαϊκό περιβάλλον.

Πρέπει ωστόσο να τονιστεί ότι τα αποτελέσματα ανίχνευσης λειτουργούν αποκλειστικά ως βοηθητικές ενδείξεις και όχι ως αποδεικτικά στοιχεία. Κανένα σύστημα ανίχνευσης δεν είναι αλάνθαστο, και η τελική κρίση ανήκει πάντα στον ίδιο τον καθηγητή. Η πλατφόρμα ακολουθεί αυτή τη φιλοσοφία, καθώς δεν έχει ως στόχο να αποκαταστήσει την ανθρώπινη κρίση, αλλά να προσφέρει επιπλέον δεδομένα που μπορούν να βοηθήσουν στη διαδικασία αξιολόγησης των εργασιών.

Κατά τη διάρκεια της ανάπτυξης αποκτήθηκε σημαντική εμπειρία σε ένα ευρύ φάσμα τεχνολογιών, τόσο στο frontend (React, Next.js, TypeScript) όσο και στο backend (Python, FastAPI, SQLAlchemy). Ιδιαίτερη πρόκληση αποτέλεσε η ενσωμάτωση ερευνητικών εργαλείων ανίχνευσης σε παραγωγικό περιβάλλον, καθώς και η εγκατάσταση σε πραγματικό server με ρύθμιση reverse proxy, HTTPS και βάσης δεδομένων MariaDB. Η εμπειρία αυτή ανέδειξε τη σημαντική διαφορά ανάμεσα στην ανάπτυξη σε τοπικό περιβάλλον και τη λειτουργία σε πραγματικό.

Συμπερασματικά, ένα σύστημα σαν και αυτό καλύπτει ένα κενό στον ακαδημαϊκό χώρο προσφέροντας ένα ολοκληρωμένο εργαλείο για τη διαχείριση και ανάλυση εργασιών των φοιτητών. Η τεχνολογία μπορεί να αποτελέσει πολύτιμο βοήθημα στη διαδικασία αξιολόγησης, αλλά δεν μπορεί και δεν πρέπει να αντικαταστήσει την κρίση του καθηγητή.

Βιβλιογραφία

- [1] I. Sommerville, Software Engineering, 10th ed. Harlow, UK: Pearson, 2016
- [2] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn, "DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature," in Proc. ICML, 2023.
- [3] J. Shi, Z. Yang, and M. R. Lyu, "Between Lines of Code: Unraveling the Distinct Patterns of Machine and Human Programmers," in Proc. ICSE, 2025. [DetectCodeGPT]
- [4] Tree-sitter, "Tree-sitter - An incremental parsing system for programming tools." [Ηλεκτρονικό]. Διαθέσιμο από: <https://tree-sitter.github.io/>
- [5] S. Ramírez, "FastAPI." [Ηλεκτρονικό]. Διαθέσιμο από: <https://fastapi.tiangolo.com/>
- [6] Vercel, "Next.js - The React Framework." [Ηλεκτρονικό]. Διαθέσιμο από: <https://nextjs.org/>
- [7] MariaDB Foundation, "MariaDB Server." [Ηλεκτρονικό]. Διαθέσιμο από: <https://mariadb.org/>
- [8] M. Bayer, "SQLAlchemy - The Database Toolkit for Python." [Ηλεκτρονικό]. Διαθέσιμο από: <https://www.sqlalchemy.org/>
- [9] F. Miao and W. Holmes, Guidance for Generative AI in Education and Research. Paris: UNESCO, 2023.[Ηλεκτρονικό]. Διαθέσιμο από: <https://unesdoc.unesco.org/ark:/48223/pf00000390613>

- [10] J. Wu, S. Yang, R. Zhan, Y. Yuan, L. S. Chao, and D. F. Wong, "A Survey on LLM-Generated Text Detection: Necessity, Methods, and Future Directions," *Computational Linguistics*, vol. 51, no. 1, pp. 275-338, 2025.
- [11] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, and S. Feizi, "Can AI-Generated Text be Reliably Detected?," *arXiv:2303.11156*, 2023.
- [12] W. Liang, M. Yuksekgonul, Y. Mao, E. Wu, and J. Zou, "GPT detectors are biased against non-native English writers," *Patterns*, vol. 4, no. 7, 2023.
- [13] M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," RFC 7519, Internet Engineering Task Force, 2015. [Ηλεκτρονικό]. Διαθέσιμο από: <https://datatracker.ietf.org/doc/html/rfc7519>
- [14] OWASP Foundation, "Password Storage Cheat Sheet." [Ηλεκτρονικό]. Διαθέσιμο από: https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html
- [15] Apache Software Foundation, "Reverse Proxy Guide - Apache HTTP Server Version 2.4." [Ηλεκτρονικό]. Διαθέσιμο από: https://httpd.apache.org/docs/2.4/howto/reverse_proxy.html
- [16] Hugging Face, "Transformers Documentation." [Ηλεκτρονικό]. Διαθέσιμο από: <https://huggingface.co/docs/transformers/>
- [17] Brevo. [Ηλεκτρονικό]. Διαθέσιμο από: <https://www.brevo.com/>
- [18] D. Weber-Wulff, A. Anohina-Naumeca, S. Bjelobaba, T. Foltýnek, J. Guerrero-Dib, O. Popoola, P. Šigut, and L. Waddington, "Testing of Detection Tools for AI-Generated Text," *International Journal for Educational Integrity*, vol. 19, no. 1, Article 26, 2023. [Ηλεκτρονικό]. Διαθέσιμο από: <https://link.springer.com/article/10.1007/s40979-023-00146-z>

- [19] C. Ardito, "Generative AI detection in higher education assessments," *New Directions for Teaching and Learning*, 2024. [Ηλεκτρονικό]. Διαθέσιμο από: <https://onlinelibrary.wiley.com/doi/10.1002/tl.20624>

- [20] Meta Platforms, Inc., "React - A JavaScript library for building user interfaces." [Ηλεκτρονικό]. Διαθέσιμο από: <https://react.dev/>

- [21] Microsoft, "TypeScript - JavaScript with syntax for types." [Ηλεκτρονικό]. Διαθέσιμο από: <https://www.typescriptlang.org/>

- [22] A. Ronacher, "Pydantic - Data validation using Python type annotations." [Ηλεκτρονικό]. Διαθέσιμο από: <https://pydantic.dev/docs/>

- [23] Wang et al. (2024) - Generative AI in higher education: Seeing ChatGPT through universities' policies, resources, and guidelines. [Ηλεκτρονικό]. Διαθέσιμο από: https://www.sciencedirect.com/science/article/pii/S2666920X24001292?utm_source=chatgpt.com

Παράρτημα Α

Ερωτηματολόγιο Αξιολόγησης Συστήματος

Οι παρακάτω ερωτήσεις αποτελούν το ερωτηματολόγιο που σχεδιάστηκε για την αξιολόγηση του συστήματος από μέλη του ακαδημαϊκού προσωπικού. Κάθε ερώτηση βαθμολογείται σε κλίμακα Likert 5 βαθμών, όπου το 1 αντιστοιχεί σε αρνητική αξιολόγηση και το 5 σε θετική. Σε μερικές ερωτήσεις ζητείται απάντηση πολλαπλής επιλογής ή σύντομη απάντηση με κείμενο.

Μέρος Α - Γενικά Στοιχεία

S1. Όνομα (προαιρετικό)

S2. Πώς αλληλοεπιδράσατε με το σύστημα;

- Δοκίμασα το σύστημα ανεξάρτητα
- Παρακολούθησα το βίντεο επίδειξη του συστήματος (demo)
- Και τα δύο

Μέρος Β - Ευχρηστία και Πλοήγηση

S3. Πόσο εύκολη ήταν η πλοήγηση στο σύστημα;

S4. Πόσο εύκολο ήταν να δημιουργήσετε ένα μάθημα και να ορίσετε ένα παραδοτέο;

S5. Πόσο χρήσιμα ήταν το βίντεο επίδειξης και το εγχειρίδιο χρήστη για την κατανόηση της λειτουργίας του συστήματος;

S6. Υπήρξαν σημεία που σας φάνηκαν μπερδεμένα ή ασαφή; Αν ναι, περιγράψτε τα.

S7. Υπάρχουν λειτουργίες που θα θέλατε να προστεθούν ή να βελτιωθούν;

Μέρος Γ - Ανίχνευση ΑΙ και Αποτελέσματα

S8. Πόσο χρήσιμη είναι η δυνατότητα ελέγχου εγγράφων κειμένου για AI-generated περιεχόμενο;

S9. Πόσο χρήσιμη είναι η δυνατότητα ελέγχου πηγαίου κώδικα για AI-generated περιεχόμενο;

S10. Τα αποτελέσματα ανίχνευσης παρουσιάζονταν με σαφή και κατανοητό τρόπο;

Μέρος Δ - Οργάνωση και Ρυθμίσεις

S11. Η οργανωτική δομή (μαθήματα → ακαδημαϊκά εξάμηνα → παραδοτέα → ομάδες) είναι πρακτική για ακαδημαϊκή χρήση;

S12. Τα προεπιλεγμένα κατώφλια ανίχνευσης ήταν κατάλληλα;

S13. Πόσο σημαντική είναι η δυνατότητα προσαρμογής των κατωφλίων ανίχνευσης ανά παραδοτέο;

Μέρος Ε - Εμπιστοσύνη και Χρήση

S14. Θα προτιμούσατε το σύστημα να σημαίνει τα αποτελέσματα για χειροκίνητο έλεγχο αντί να δίνει οριστική κατηγοριοποίηση;

- Ναι, η σήμανση για έλεγχο είναι προτιμότερη
- Όχι, προτιμώ οριστική κατηγοριοποίηση
- Δεν έχω προτίμηση

S15. Σε ποιον βαθμό θα εμπιστευόσασταν τα αποτελέσματα του συστήματος ως ένδειξη (και όχι ως απόδειξη) χρήσης ΑΙ;

S16. Θα χρησιμοποιούσατε τα αποτελέσματα ως αφετηρία για περαιτέρω διερεύνηση, π.χ. συζήτηση με τον φοιτητή;

- Ναι
- Όχι
- Ίσως

S17. Πόσο πιθανό είναι να χρησιμοποιούσατε αυτό το σύστημα σε κάποιο μάθημά σας;

S18. Παρακαλούμε μοιραστείτε τυχόν πρόσθετα σχόλια ή προτάσεις που θα μπορούσαν να βοηθήσουν στη βελτίωση του συστήματος.

Παράρτημα Β

Οδηγίες Χρήσης Συστήματος Ανίχνευσης AI-Generated Περιεχομένου

1. Εισαγωγή

Το παράρτημα αυτό αποτελεί συνοπτικό εγχειρίδιο χρήσης του συστήματος. Στόχος του είναι να παρουσιάσει τη βασική ροή χρήσης της εφαρμογής από την πλευρά του τελικού χρήστη, δηλαδή κυρίως του καθηγητή που επιθυμεί να οργανώσει μαθήματα, να υποβάλει φοιτητικές εργασίες και να δει τα αποτελέσματα της ανάλυσης τους. Το σύστημα επιτρέπει στους καθηγητές να δημιουργούν μαθήματα, ακαδημαϊκά εξάμηνα και παραδοτέα, να υποβάλλουν αρχεία κειμένου και πηγαίου κώδικα, καθώς και να λαμβάνουν αναλυτικές αναφορές με ενδείξεις πιθανής χρήσης Τεχνητής Νοημοσύνης. Επιπλέον, παρέχει δυνατότητα ρύθμισης κατωφλίων ανίχνευσης και προβολής συγκεντρωτικών στατιστικών μέσω της σελίδας Analytics. Τα αποτελέσματα του συστήματος πρέπει να χρησιμοποιούνται ως υποστηρικτική ένδειξη και όχι ως απόλυτη απόδειξη χρήσης AI. Η τελική αξιολόγηση παραμένει στην κρίση του καθηγητή.

2. Προαπαιτούμενα

Για τη χρήση του συστήματος απαιτούνται τα ακόλουθα:

- Πρόσβαση σε υπολογιστή με σύνδεση στο διαδίκτυο.
- Χρήση σύγχρονου φυλλομετρητή.
- Πρόσβαση στη διεύθυνση του συστήματος.
- Σύνδεση στο δίκτυο του Πανεπιστημίου ή μέσω VPN.
- Κωδικός πρόσκλησης, σε περίπτωση δημιουργίας νέου λογαριασμού.

3. Εγγραφή και Σύνδεση

Κατά την πρώτη επίσκεψη στο σύστημα, ο χρήστης μεταφέρεται στη σελίδα σύνδεσης. Εκεί μπορεί να εισάγει το email και τον κωδικό πρόσβασής του για να αποκτήσει πρόσβαση στην εφαρμογή. Αν ο χρήστης δεν διαθέτει λογαριασμό, επιλέγει τη δημιουργία νέου λογαριασμού. Στη φόρμα εγγραφής εισάγει το email του, τον κωδικό πρόσβασης και το invite code που του έχει δοθεί από τον διαχειριστή. Το invite code λειτουργεί ως μηχανισμός περιορισμού πρόσβασης, ώστε να μπορούν να εγγραφούν μόνο εξουσιοδοτημένοι χρήστες.

Μετά την εγγραφή, ο χρήστης λαμβάνει κωδικό μιας χρήσης στο email του για επιβεβαίωση της διεύθυνσής του. Αφού ολοκληρωθεί η επιβεβαίωση, ο λογαριασμός παραμένει σε κατάσταση αναμονής μέχρι να εγκριθεί από τον διαχειριστή.

Σε περίπτωση που ο χρήστης ξεχάσει τον κωδικό πρόσβασής του, μπορεί να χρησιμοποιήσει την επιλογή επαναφοράς κωδικού. Μέσω αυτής της διαδικασίας λαμβάνει νέο κωδικό OTP στο email του και μπορεί να ορίσει νέο κωδικό πρόσβασης.

The image displays four distinct user interface screens for a system's authentication and registration process:

- Login Screen:** Features a 'Secure access' indicator. It includes a title 'Login', a subtitle 'Sign in to see only your own submissions and reports.', and input fields for 'Email' (pre-filled with 'name@ucy.ac.cy') and 'Password' (masked with '*****'). A 'Forgot password?' link is present. A dark blue 'Login' button is at the bottom. Links for 'Don't have an account?' and 'Create account ->' are at the very bottom.
- Register Screen:** Features a 'Create account' indicator. It includes a title 'Register', a subtitle 'Create an account to upload files and view only your own reports.', and input fields for 'Email' (pre-filled with 'name@ucy.ac.cy'), 'Invite code' (pre-filled with 'ADE2025-TEACHER'), and 'Password' (masked with '*****'). A note 'Ask the admin for the invite code.' is above the password field. A tip 'Tip: enter at least 8 characters.' is below the password field. A dark blue 'Register' button is at the bottom. A 'Go to login ->' button is at the bottom right.
- Forgot Password Screen:** Features a 'Password reset' indicator. It includes a title 'Forgot Password', a subtitle 'Enter your email to receive a verification code.', and an 'Email' input field (pre-filled with 'name@ucy.ac.cy'). A dark blue 'Send Reset Code' button is at the bottom. A 'Back to login' button is at the bottom right.
- Unlabeled Screen (Bottom Left):** Features a 'Password reset' indicator. It includes a title 'Forgot Password', a subtitle 'Enter your email to receive a verification code.', and an 'Email' input field (pre-filled with 'name@ucy.ac.cy'). A dark blue 'Send Reset Code' button is at the bottom. A 'Back to login' button is at the bottom right.

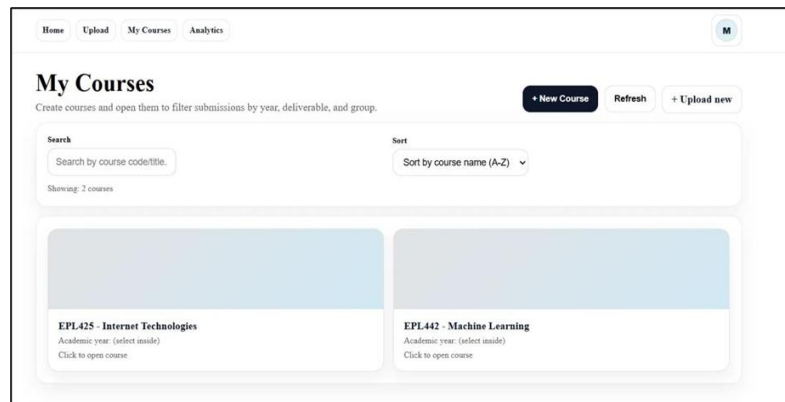
4. Δημιουργία Μαθήματος, Εξαμήνου και Παραδοτέου

Μετά τη σύνδεση, ο καθηγητής μεταφέρεται στο περιβάλλον χρήσης του συστήματος. Η βασική οργάνωση των δεδομένων βασίζεται σε τρία επίπεδα: μάθημα, ακαδημαϊκό εξάμηνο και παραδοτέο.

Αρχικά, ο καθηγητής δημιουργεί ένα μάθημα εισάγοντας τον κωδικό και, προαιρετικά, τον τίτλο του μαθήματος. Κάθε μάθημα λειτουργεί ως ξεχωριστή οντότητα μέσα στο σύστημα και περιέχει τα αντίστοιχα εξάμηνα, παραδοτέα και αρχεία. Στη συνέχεια, μέσα στο μάθημα, ο καθηγητής δημιουργεί ένα ακαδημαϊκό εξάμηνο. Το εξάμηνο μπορεί να αντιστοιχεί, για παράδειγμα, σε «Fall 2025», «Spring 2026» ή άλλο ακαδημαϊκό εξάμηνο. Με αυτόν τον τρόπο, το ίδιο μάθημα μπορεί να οργανώνεται ξεχωριστά για διαφορετικά ακαδημαϊκά έτη ή εξάμηνα. Αφού δημιουργηθεί το εξάμηνο, ο καθηγητής προσθέτει το παραδοτέο. Το παραδοτέο αντιστοιχεί στη συγκεκριμένη εργασία ή φάση έργου που θα υποβάλουν οι φοιτητές, όπως για παράδειγμα «Assignment 1», «Project 1» ή «Phase A». Προαιρετικά μπορεί να οριστεί και ημερομηνία παράδοσης.

Η βασική ροή είναι η εξής:

- Δημιουργία μαθήματος.
- Δημιουργία ακαδημαϊκού εξαμήνου μέσα στο μάθημα.
- Δημιουργία παραδοτέου μέσα στο συγκεκριμένο εξάμηνο.



My Courses

Create courses and open them to filter submissions by year, deliverable, and group.

[+ New Course](#) [Refresh](#) [+ Upload new](#)

Search

Sort

Showing: 2 courses

Course code

Course title (optional)

[Cancel](#) [Create](#)

Academic Semester [+ Add](#)

New term

Term **Year**

Will be saved as: Fall 2026

[Cancel](#) [Create](#)

Deliverable [+ Add](#)

Group-Student-Team

[Clear](#) [Apply Filters](#)

[+ Upload](#)

Filtered files: 8 submissions

Deliverable [+ Add](#)

New deliverable

Due date is optional.

[Cancel](#) [Create](#)

Academic Semester [+ Add](#)

Group-Student-Team

[Clear](#) [Apply Filters](#)

[+ Upload](#)

All files: 8 submissions

5. Υποβολή Αρχείων για Ανάλυση

Αφού δημιουργηθεί το παραδοτέο, ο καθηγητής μπορεί να υποβάλει αρχεία για ανάλυση. Η υποβολή γίνεται μέσω της σελίδας Upload, όπου ο χρήστης επιλέγει το μάθημα, το ακαδημαϊκό εξάμηνο και το παραδοτέο στο οποίο ανήκουν τα αρχεία.

Το σύστημα υποστηρίζει μαζική υποβολή αρχείων μέσω ZIP. Το ZIP αρχείο πρέπει να είναι οργανωμένο σε φακέλους, όπου κάθε φάκελος αντιστοιχεί σε μία ομάδα ή κάποιο φοιτητή. Για παράδειγμα, ένα ZIP μπορεί να περιέχει τους φακέλους:

- Group1/
- Group2/
- Group3/

Μέσα σε κάθε φάκελο τοποθετούνται τα αντίστοιχα αρχεία της ομάδας. Το σύστημα δημιουργεί αυτόματα τις ομάδες με βάση τα ονόματα των φακέλων και αποθηκεύει τα αρχεία στο αντίστοιχο παραδοτέο.

Κατά την υποβολή, ο καθηγητής μπορεί επίσης να επιλέξει αν θα εκτελεστεί άμεσα ο έλεγχος ΑΙ. Αν ενεργοποιηθεί αυτή η επιλογή, η ανάλυση ξεκινά κατά το ανέβασμα των αρχείων. Η διαδικασία μπορεί να διαρκέσει περισσότερο, ανάλογα με το πλήθος και το μέγεθος των αρχείων.

Το σύστημα υποστηρίζει αρχεία κειμένου, όπως DOCX, καθώς και αρχεία πηγαίου κώδικα, όπως Python, Java, JavaScript, C, C++ και άλλες γλώσσες προγραμματισμού.

Bulk Upload

Upload one ZIP that contains all groups for the selected deliverable. Supported files may include DOCX or source code files, depending on the submission type.

Course

EPL425 - Internet Technologies

Academic Semester

Spring 2026

Deliverable

Phase A

Status

Choose a ZIP that contains all groups

Tags

comma-separated (optional)

☒

Run AI check during bulk upload
(slower)

ZIP file (folders per group)

Expected structure: Group01/file.example (e.g. report.docx, main.py, Program.java)

Choose File

EPL425 - Phase A.zip

DOCX files will use document AI detection. Source code files will use code AI detection.

No zip selected yet.

Bulk Upload ZIP

Result

Bulk upload ok

Uploaded 6 files • Created 3 groups • Failed 0 files

All files were uploaded successfully.

← Back to courses

View submissions for EPL425 - Internet Technologies →

[Home](#) [Upload](#) [My Courses](#) [Analytics](#) P

Bulk Upload

Upload one ZIP that contains all groups for the selected deliverable. Supported files may include DOCX or source code files, depending on the submission type.

Course

Select course...

Academic Semester

Select semester...

Deliverable

Select deliverable...

Status

Select a course

Tags

comma-separated (optional)

☐

Run AI check during bulk upload (slower)

ZIP file (folders per group)

Expected structure: Group01/file.example (e.g. report.docx, main.py, Program.java)

Choose File

No file chosen

DOCX files will use document AI detection. Source code files will use code AI detection.

No zip selected yet.

Bulk Upload ZIP

Result

After bulk upload, files will appear under the selected deliverable's groups.

← Back to courses

6. Προβολή Αποτελεσμάτων Ανίχνευσης

Μετά την ολοκλήρωση της ανάλυσης, ο καθηγητής μπορεί να δει την αναφορά κάθε αρχείου πατώντας την επιλογή Report δίπλα από την αντίστοιχη υποβολή. Η μορφή της αναφοράς διαφέρει ανάλογα με τον τύπο του αρχείου.

Για αρχεία κειμένου, το σύστημα παρουσιάζει αναφορά σε επίπεδο παραγράφου. Στην κορυφή εμφανίζονται βασικά στοιχεία, όπως το όνομα του αρχείου, ο τύπος του αρχείου, η τελική ετικέτα, το ποσοστό πιθανής χρήσης AI, ο δείκτης αξιοπιστίας και η κατάσταση της ανάλυσης. Στη συνέχεια εμφανίζονται συνοπτικά στατιστικά για το έγγραφο, όπως ο αριθμός των παραγράφων, πόσες αναλύθηκαν και πόσες χαρακτηρίστηκαν ως πιθανώς AI-generated.

Ένα βασικό χαρακτηριστικό της αναφοράς κειμένου είναι η προβολή της δομής του εγγράφου. Το σύστημα οργανώνει τις παραγράφους με βάση τις επικεφαλίδες και επιτρέπει στον καθηγητή να δει ποιες ενότητες παρουσιάζουν μεγαλύτερη πιθανότητα χρήσης AI.

Παράλληλα, εμφανίζεται προεπισκόπηση του αρχικού αρχείου, ώστε να μπορεί να γίνει σύγκριση μεταξύ του πρωτότυπου και των αποτελεσμάτων της ανάλυσης.

Για αρχεία πηγαίου κώδικα, η αναφορά περιλαμβάνει διαφορετικά στοιχεία. Παρουσιάζονται η τελική ετικέτα, το ποσοστό AI, ο δείκτης αξιοπιστίας, η μέθοδος ανίχνευσης και τα κομμάτια κώδικα που αναλύθηκαν. Επιπλέον, εμφανίζονται πιθανότητες ανά κατηγορία, καθώς και σήματα που επηρέασαν το αποτέλεσμα, όπως μοτίβα στη δομή του κώδικα, χρήση βοηθητικών συναρτήσεων, σχόλια ή άλλα χαρακτηριστικά που εντόπισε ο αναλυτής.

Τα αποτελέσματα πρέπει να ερμηνεύονται με προσοχή. Ένα υψηλό ποσοστό δεν αποτελεί από μόνο του απόδειξη χρήσης AI, αλλά ένδειξη ότι το περιεχόμενο παρουσιάζει χαρακτηριστικά που συνδέονται με AI-generated κείμενο ή κώδικα.

Report - Submission #35

Full report (summary + paragraphs).

File
ep1343.del3.DD_template_2023.docx

File kind
DOXC

Final label
UNCERTAIN

Trust
49/100

% AI
11.11%

AI check
Done

Created: Thu, 30 Apr 2026 06:25:08 GMT

Run AI Check

Hide Preview

PDF Preview

Report / Summary

- Total paragraphs: 65
- Scored paragraphs: 9
- AI paragraphs: 1 (11.11%)
- Edited paragraphs: 0 (0.00%)
- Hybrid paragraphs: 3 (33.33%)

[Show raw summary JSON](#)

Document structure

Revision Chart 65 pars 11% AI

Introduction 22 pars 50% AI

Purpose 1 par 100% AI

#0 AI score: 0.697

The purpose of this document is to provide a detailed and clear design plan for the application. It sets clear guidelines about the application's structure, so everyone during the development can have a unified understanding of the architecture and functionality. This document is a reference tool, and it is intended for developers, designers, project managers, unit testers and stakeholders.

Scope 18 pars Clean

References 3 pars Clean

Architecture 5 pars Clean

Sample Scenarios and diagrams 37 pars Clean

Appendices 1 par Clean

PDF Preview

Reload Close

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed and clear design plan for the application. It sets clear guidelines about the application's structure, so everyone during the development can have a unified understanding of the architecture and functionality. This document is a reference tool, and it is intended for developers, designers, project managers, unit testers and stakeholders.

1.2 Scope

We are making a website for a private institute. This website, called 'Tutoring', will be a place in which it will provide general information to potentially interested "customers". Additionally current students at the institute will find features that aim to help them by providing a visual schedule of their weekly lessons, but also inform them about any extra lessons they may have to attend. The benefits of this product will reside in the ability to provide easy access to information for teaching staff and students associated with the teaching institute. The goal is to create an environment for both teachers and students where both can communicate without any misinterpretation of information regarding any extra lessons. Moreover, it will provide an easier way for individuals interested in tutoring to know when and where our client provides these lessons before even contacting the institute's owner.

What the Product Will Do

- General information about Institute
- Schedule for upcoming year

What the Product Will Not Do

- The teachers will not provide the lesson's content online
- Students will not be able to have feedback

How the Software Will Be Used

- Student should be able to "view":
 - Weekly (standard) schedule and extra lessons
- Teacher should be able to:
 - Insert extra lessons, "view" schedule of other teachers and move students from their classroom

Relevant Benefits, Goals

- Benefits:
 - Convenience: "Quality of life" features for students and teachers.
- Goals:
 - Develop a complete software solution to support the private institute.

4

Report - Submission #50

Full report for code AI detection.

File
group1.py
File kind
CODE
Final label
AI
Trust
37/100
% AI
63.25%
AI check
Done

Created: Tue, 12 May 2026 05:51:59 GMT

Run AI Check

Hide Preview
Open file

Code Analysis Summary

Final label AI	AI percentage 63.25%	Trustworthiness 37/100
Model hybrid-code-detector-v1	Raw detector label AI	Chunks analyzed 1

Class probabilities

Human generated 36.75%	Machine generated 63.25%	Machine refined 25.30%
Adversarial 12.65%		

Signals

- Contains generic boilerplate phrasing
- Over-structured helper-function decomposition
- Many functions for a relatively small file
- Low comment density
- Analyzed 1 code chunks
- Rule-based code signal analysis
- Contains boilerplate instructional phrasing
- Strong helper-function decomposition pattern
- Many functions relative to file size
- Very low comment density
- Interactive development style more typical of human coursework
- Structured defensive coding pattern
- Hybrid code analysis fallback
- Combined rule-based and ML-based code detector
- Fusion weights: 40% rules, 60% ML
- DetectCodeGPT unavailable or unsupported for this file
- Used hybrid fallback detector instead
- DetectCodeGPT adapter failed during execution
- Fallback detector should be used by the caller

Detected AI-like patterns

Code Preview

```

def display_menu():
    print("\n--- Task Manager ---")
    print("1. Add Task")
    print("2. View Tasks")
    print("3. Remove Task")
    print("4. Exit")

def add_task(tasks):
    task = input("Enter a task: ").strip()
    if task:
        tasks.append(task)
        print("Task added successfully.")
    else:
        print("Task cannot be empty.")

def view_tasks(tasks):
    if not tasks:
        print("No tasks available.")
        return

    print("\nYour tasks:")
    for index, task in enumerate(tasks, start=1):
        print(f"{index}. {task}")

def remove_task(tasks):
    view_tasks(tasks)
    if not tasks:
        return

    try:
        task_number = int(input("Enter the task number to remove: "))
        if 1 <= task_number <= len(tasks):
            removed_task = tasks.pop(task_number - 1)
            print(f"Removed task: {removed_task}")
        else:
            print("Invalid task number.")
    except ValueError:
        print("Please enter a valid number.")

def main():
    tasks = []

    while True:
        display_menu()
        choice = input("Choose an option: ").strip()

        if choice == "1":
            add_task(tasks)

```

7. Ρύθμιση Κατωφλίων Ανίχνευσης

Το σύστημα επιτρέπει στον καθηγητή να προσαρμόζει τα κατώφλια ανίχνευσης σύμφωνα με τις ανάγκες του μαθήματος ή της αξιολόγησης. Οι ρυθμίσεις αυτές βρίσκονται στη σελίδα Profile του χρήστη.

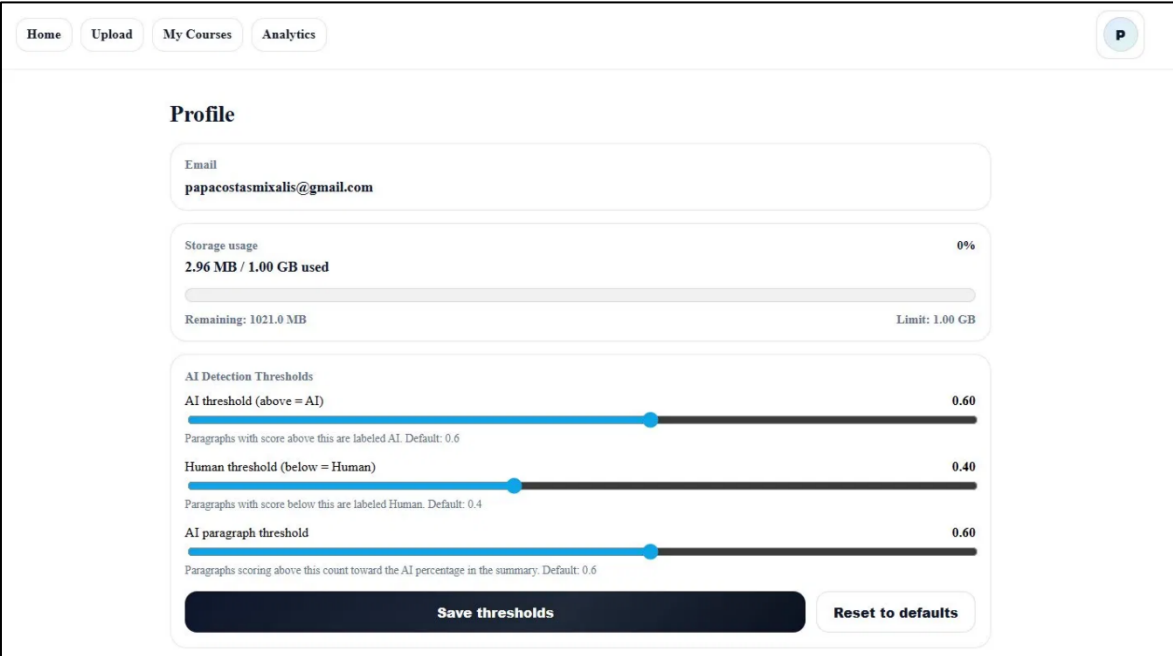
Τα βασικά κατώφλια είναι:

- AI threshold: καθορίζει από ποιο score και πάνω ένα αρχείο ή τμήμα περιεχομένου θεωρείται πιθανώς AI-generated.

- Human threshold: καθορίζει κάτω από ποιο score το περιεχόμενο θεωρείται περισσότερο ανθρώπινο.
- AI paragraph threshold: καθορίζει από ποιο score και πάνω μια παράγραφος προσμετράται στο συνολικό ποσοστό AI του εγγράφου.

Ο καθηγητής μπορεί να αλλάξει τις τιμές μέσω των αντίστοιχων sliders και να αποθηκεύσει τις αλλαγές. Αν χρειαστεί, μπορεί επίσης να επαναφέρει τις προεπιλεγμένες τιμές του συστήματος.

Η δυνατότητα αυτή είναι χρήσιμη επειδή διαφορετικά μαθήματα ή διαφορετικά είδη εργασιών μπορεί να απαιτούν διαφορετικό επίπεδο αυστηρότητας στην ανάλυση.



The screenshot shows the 'Profile' page within the 'Analytics' section. At the top, there are navigation tabs: 'Home', 'Upload', 'My Courses', and 'Analytics'. The 'Profile' section includes:

- Email:** papacostasmixalis@gmail.com
- Storage usage:** 2.96 MB / 1.00 GB used (0%). A progress bar shows the usage, with 'Remaining: 1021.0 MB' and 'Limit: 1.00 GB'.
- AI Detection Thresholds:**
 - AI threshold (above = AI):** A slider set to 0.60. Below it, text says 'Paragraphs with score above this are labeled AI. Default: 0.6'.
 - Human threshold (below = Human):** A slider set to 0.40. Below it, text says 'Paragraphs with score below this are labeled Human. Default: 0.4'.
 - AI paragraph threshold:** A slider set to 0.60. Below it, text says 'Paragraphs scoring above this count toward the AI percentage in the summary. Default: 0.6'.
- Buttons:** 'Save thresholds' and 'Reset to defaults'.

8. Σελίδα Analytics

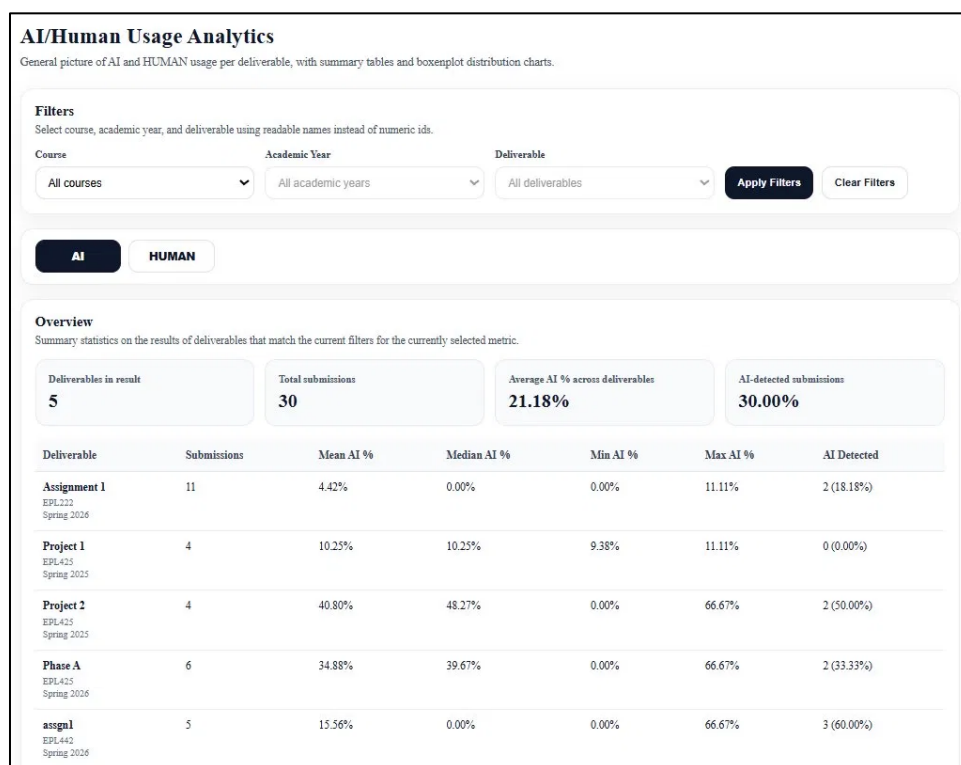
Η σελίδα Analytics παρέχει συγκεντρωτική εικόνα για τις υποβολές και τα αποτελέσματα ανίχνευσης. Μέσω αυτής της σελίδας, ο καθηγητής μπορεί να παρακολουθεί τα αποτελέσματα ανά μάθημα, ακαδημαϊκό εξάμηνο και παραδοτέο.

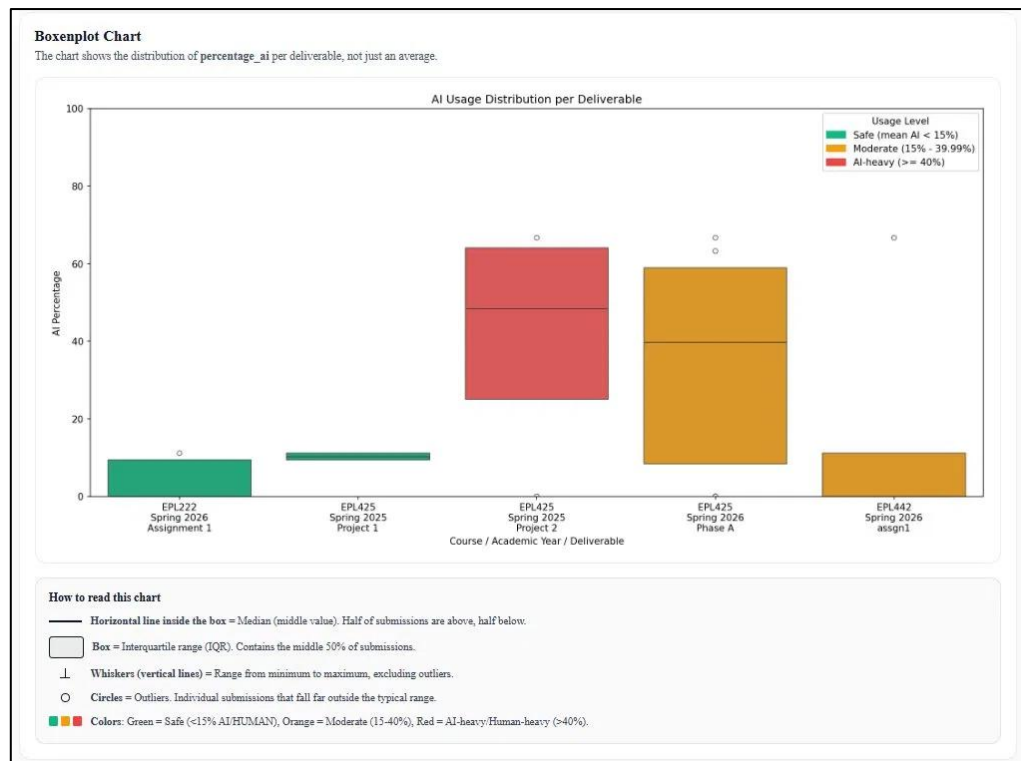
Στο πάνω μέρος της σελίδας υπάρχουν φίλτρα, με τα οποία ο χρήστης μπορεί να επιλέξει συγκεκριμένο μάθημα, εξάμηνο ή παραδοτέο. Επίσης, μπορεί να εναλλάσσει την προβολή μεταξύ AI και HUMAN, ανάλογα με το είδος της πληροφορίας που θέλει να εξετάσει.

Η σελίδα εμφανίζει συνοπτικά στατιστικά, όπως:

- συνολικό αριθμό υποβολών,
- μέσο ποσοστό AI ή Human,
- ελάχιστη και μέγιστη τιμή,
- διάμεσο,
- αριθμό υποβολών που χαρακτηρίστηκαν ως AI ή Human.

Επιπλέον, παρουσιάζονται γραφήματα τύπου boxplot, τα οποία δείχνουν την κατανομή των ποσοστών στις υποβολές κάθε παραδοτέου. Με αυτόν τον τρόπο, ο καθηγητής μπορεί να εντοπίσει αν ένα συγκεκριμένο παραδοτέο παρουσιάζει αυξημένα ποσοστά πιθανής χρήσης AI σε σχέση με άλλα. Η σελίδα Analytics βοηθά τον καθηγητή να αποκτήσει γρήγορη εποπτεία των αποτελεσμάτων, χωρίς να χρειάζεται να ανοίξει κάθε αναφορά ξεχωριστά.





9. Λειτουργίες Διαχειριστή

Ο διαχειριστής έχει πρόσβαση σε ξεχωριστό περιβάλλον διαχείρισης. Από εκεί μπορεί να παρακολουθεί τους χρήστες του συστήματος και να ελέγχει την πρόσβασή τους.

Οι βασικές λειτουργίες του διαχειριστή είναι:

- προβολή όλων των εγγεγραμμένων χρηστών,
- έγκριση χρηστών που βρίσκονται σε κατάσταση αναμονής,
- διαγραφή χρηστών, όταν χρειάζεται,
- παρακολούθηση της χρήσης αποθηκευτικού χώρου ανά χρήστη.

Με αυτόν τον τρόπο διασφαλίζεται ότι η πρόσβαση στο σύστημα γίνεται μόνο από εξουσιοδοτημένους χρήστες.

Admin Panel

M

Admin dashboard

Users

Manage registrations, approvals, deletions, and track uploads per user.

Pending approvals

0

Users waiting for admin approval

Total users

5

Based on current filter

Filter: All Refresh

ID	Email	Role	Status	Uploads	Created	Actions
5	miketesting945@gmail.com	TEACHER	APPROVED	8	7/5/2026	Delete
4	econst07@ucy.ac.cy	TEACHER	APPROVED	3	21/4/2026	Delete
3	papakostas.michalis@ucy.ac.cy	TEACHER	APPROVED	0	21/4/2026	Delete
2	papacostasmixalis@gmail.com	TEACHER	APPROVED	30	20/4/2026	Delete

Admin Panel

M

Admin Profile

Email

mpapak02@ucy.ac.cy

Users storage

miketesting945@gmail.com

83.7 KB / 1.00 GB used · Remaining 1023.9 MB

0%

econst07@ucy.ac.cy

1.09 MB / 1.00 GB used · Remaining 1022.9 MB

0%

papakostas.michalis@ucy.ac.cy

0 B / 1.00 GB used · Remaining 1.00 GB

0%

papacostasmixalis@gmail.com

2.96 MB / 1.00 GB used · Remaining 1021.0 MB

0%

B-12

Παράρτημα Γ

Τεχνικό Εγχειρίδιο του Συστήματος

1. Περιγραφή Συστήματος

Το σύστημα αποτελείται από μια διαδικτυακή πλατφόρμα που λειτουργεί σε αρχιτεκτονική client-server. Το frontend είναι υλοποιημένο σε Next.js (React / TypeScript) και εξυπηρετείται στη θύρα 3000. Το backend είναι υλοποιημένο σε FastAPI (Python) και εξυπηρετείται στη θύρα 8000. Ο Apache web server λειτουργεί ως reverse proxy, δρομολογώντας τα αιτήματα στην κατάλληλη υπηρεσία. Η βάση δεδομένων είναι MariaDB και τα αρχεία αποθηκεύονται στο filesystem του server.

2. Αρχιτεκτονική Εφαρμογής

Η εφαρμογή αποτελείται από τα εξής βασικά τμήματα:

- Frontend (Next.js / React / TypeScript)
- Backend API (FastAPI / Python)
- Βάση Δεδομένων (MariaDB)
- Μηχανισμοί Ανάλυσης (DetectCodeGPT, ML pipeline, rule-based detector)
- Apache Reverse Proxy (HTTPS)
- Filesystem (αποθήκευση αρχείων)

Κάθε τμήμα λειτουργεί ανεξάρτητα και επικοινωνεί μέσω REST API ή σύνδεση βάσης δεδομένων.

3. Προαπαιτούμενα Server

- Λειτουργικό σύστημα: Rocky Linux 9
- Python: 3.12+
- Node.js: 18+
- MariaDB: 10.5+
- Apache: 2.4+
- LibreOffice: για μετατροπή DOCX σε PDF

4. Οδηγίες Εγκατάστασης

4.1 Αντιγραφή Αρχείων Εφαρμογής

Τα αρχεία της εφαρμογής αντιγράφονται στον server στον κατάλληλο φάκελο (π.χ. /sys-data/ai-detection/). Η δομή πρέπει να περιέχει δύο κύριους φακέλους και το αρχείο requirements.txt:

- ai-text-detector/ (backend)
- frontend/
- requirements.txt

4.2 Εγκατάσταση Backend

```
cd /sys-data/ai-detection/ai-text-detector
python3 -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
```

4.3 Ρύθμιση Μεταβλητών Περιβάλλοντος (Backend)

Δημιουργείται αρχείο .env στον φάκελο ai-text-detector/ με τα εξής:

DATABASE_URL = mysql+pymysql://username:password@localhost/ai_detection_db

JWT_SECRET_KEY = ένα_μυστικό_κλειδί

CORS_ORIGINS = https://ai-detection.in.cs.ucy.ac.cy

BREVO_API_KEY = κλειδί_brevo

BREVO_SENDER_EMAIL = noreply@example.com

INVITE_CODE = κωδικός_πρόσκλησης

UPLOAD_DIR = /sys-data/ai-detection/ai-text-detector/storage/uploads

PREVIEW_DIR = /sys-data/ai-detection/ai-text-detector/storage/previews

4.4 Εγκατάσταση Frontend

```
cd /sys-data/ai-detection/frontend
```

```
npm install
```

4.5 Ρύθμιση Μεταβλητών Περιβάλλοντος (Frontend)

Δημιουργείται αρχείο .env.production στον φάκελο frontend/ με τα εξής:

NEXT_PUBLIC_API_BASE=https://ai-detection.in.cs.ucy.ac.cy/api

4.6 Build Frontend

```
cd /sys-data/ai-detection/frontend
```

```
npm run build
```

5. Ρύθμιση Apache (Reverse Proxy)

Δημιουργείται αρχείο /etc/httpd/conf.d/ai-detection.conf:

```
<VirtualHost *:80>
    ServerName ai-detection.in.cs.ucy.ac.cy
    RewriteEngine On
    RewriteRule ^/?(.*)$ https://%{HTTP_HOST}/$1 [R=301,L]
</VirtualHost>
<VirtualHost *:443>
    ServerName ai-detection.in.cs.ucy.ac.cy
    SSLEngine on
    SSLCertificateFile    /path/to/certificate.pem
    SSLCertificateKeyFile /path/to/private.key
    SSLCertificateChainFile /path/to/certificate.pem
    ProxyPreserveHost On
    ProxyRequests Off
    ProxyTimeout 600
    RequestHeader set X-Forwarded-Proto "https"
    RequestHeader set X-Forwarded-Host "ai-detection.in.cs.ucy.ac.cy"
    LimitRequestBody 524288000
    ProxyPass    /api/ http://127.0.0.1:8000/
    ProxyPassReverse /api/ http://127.0.0.1:8000/
    ProxyPass    / http://127.0.0.1:3000/
    ProxyPassReverse / http://127.0.0.1:3000/
    ErrorLog /var/log/httpd/ai-detection-error.log
    CustomLog /var/log/httpd/ai-detection-access.log combined
</VirtualHost>
```

Ενεργοποίηση και επανεκκίνηση:

```
sudo systemctl restart httpd
```

6. Εκκίνηση Υπηρεσιών

6.1 Εκκίνηση Backend

```
cd /sys-data/ai-detection/ai-text-detector  
source .venv/bin/activate  
uvicorn app.main:app --host 127.0.0.1 --port 8000 \  
    --proxy-headers --forwarded-allow-ips = 127.0.0.1 \  
    --root-path /api
```

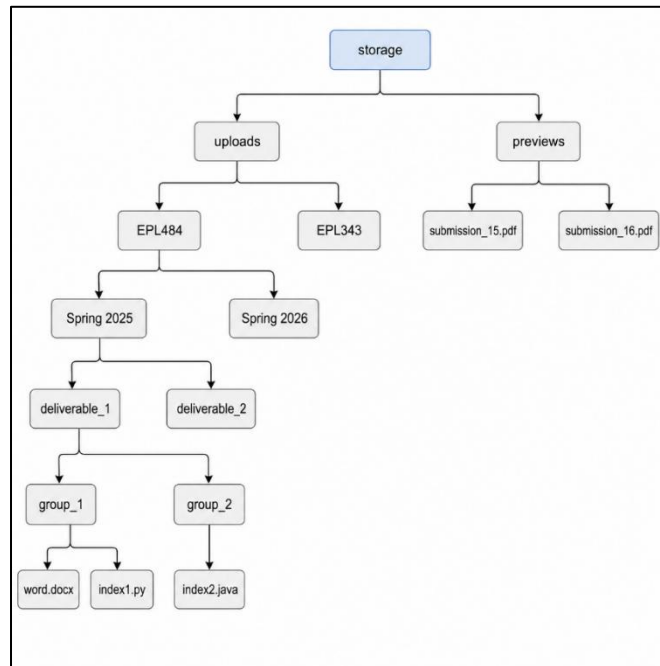
6.2 Εκκίνηση Frontend

```
cd /sys-data/ai-detection/frontend  
npm run start -- -p 3000
```

Για μόνιμη εκτέλεση μετά από αποσύνδεση SSH μπορεί να χρησιμοποιηθεί nohup:

```
nohup uvicorn app.main:app --host 127.0.0.1 --port 8000 \  
    --proxy-headers --forwarded-allow-ips = 127.0.0.1 \  
    --root-path /api &  
nohup npm run start -- -p 3000 &
```

7. Δομή Φακέλων Αποθήκευσης



Τα previews δημιουργούνται αυτόματα κατά το άνοιγμα αναφοράς DOCX, μέσω του LibreOffice.

8. Αντιμετώπιση Προβλημάτων

Πρόβλημα: 502 Proxy Error κατά τη μαζική υποβολή

Αιτία: Ο Apache κλείνει τη σύνδεση πριν ολοκληρωθεί η ανάλυση.

Λύση: Αύξηση του ProxyTimeout στο Apache config (π.χ. ProxyTimeout 600).

Πρόβλημα: Η ανάλυση κώδικα είναι πολύ αργή

Αιτία: Ο server δεν διαθέτει GPU. Το DetectCodeGPT τρέχει σε CPU.

Λύση: Η ανάλυση μπορεί να εκτελείται χωρίς αυτόματο AI check κατά το upload, και να γίνεται χειροκίνητα αργότερα.

Πρόβλημα: Η μετατροπή DOCX σε PDF δεν λειτουργά

Αιτία: Το LibreOffice δεν είναι εγκατεστημένο.

Λύση: `sudo dnf install libreoffice-headless`

Πρόβλημα: Αποτυχία σύνδεσης στη βάση δεδομένων

Αιτία: Λανθασμένο DATABASE_URL στο .env ή η MariaDB δεν τρέχει.

Λύση: Έλεγχος με `sudo systemctl status mariadb` και επαλήθευση στοιχείων σύνδεσης.

Παράρτημα Δ

Παραδείγματα Υλοποίησης

Το παρόν παράρτημα παρουσιάζει ενδεικτικά αποσπάσματα κώδικα από βασικά σημεία της υλοποίησης του συστήματος. Στόχος δεν είναι η πλήρης ανάλυση ολόκληρου του κώδικα, αλλά η παρουσίαση επιλεγμένων τμημάτων που βοηθούν στην καλύτερη κατανόηση της αρχιτεκτονικής και της λειτουργίας της εφαρμογής.

Τα παραδείγματα που ακολουθούν καλύπτουν τόσο το backend όσο και το frontend του συστήματος. Συγκεκριμένα, παρουσιάζονται σημεία που αφορούν την αρχικοποίηση της εφαρμογής, τη σύνδεση με τη βάση δεδομένων, τον ορισμό των βασικών μοντέλων, την αυθεντικοποίηση των χρηστών, την αναγνώριση τύπου αρχείων, την εκτέλεση της διαδικασίας ανίχνευσης AI-generated περιεχομένου, την επικοινωνία μεταξύ frontend και backend, καθώς και τη δομή των components στο περιβάλλον του React/Next.js.

Η επιλογή των συγκεκριμένων παραδειγμάτων έγινε επειδή αντιπροσωπεύουν κεντρικές λειτουργίες του συστήματος. Μέσα από αυτά φαίνεται ο τρόπος με τον οποίο συνεργάζονται τα διάφορα υποσυστήματα, από την υποβολή ενός αρχείου μέχρι την ανάλυσή του και την αποθήκευση των αποτελεσμάτων στη βάση δεδομένων. Παράλληλα, αναδεικνύεται και η λογική διαχωρισμού ευθυνών που ακολουθήθηκε κατά την ανάπτυξη, με ξεχωριστά αρχεία και modules για την αυθεντικοποίηση, τη διαχείριση υποβολών, τη δομή των μαθημάτων, τα στατιστικά στοιχεία και την επικοινωνία με τη βάση.

Η υλοποίηση βασίζεται σε FastAPI για το backend, SQLAlchemy για την επικοινωνία με τη βάση δεδομένων, JWT για την αυθεντικοποίηση χρηστών, καθώς και React/Next.js για το frontend. Η παρουσίαση των αποσπασμάτων ακολουθεί θεματική σειρά, ώστε κάθε τμήμα να συνδέεται άμεσα με τη λειτουργία που εξυπηρετεί μέσα στο συνολικό σύστημα.

Δ.1 Αρχικοποίηση Εφαρμογής και Routing (FastAPI)

Η εφαρμογή ξεκινά από το `main.py`, όπου δηλώνεται η FastAPI, ρυθμίζεται το CORS και εγγράφονται τα routers:

```
# app/main.py
import os
from dotenv import load_dotenv
load_dotenv()

from fastapi import FastAPI
from fastapi.middleware.cors import CORSMiddleware
from .routes_auth import router as auth_router
from .routes_submissions import router as submissions_router
from .routes_admin import router as admin_router
from .routes_structure import router as structure_router
from .routes_analytics import router as analytics_router

app = FastAPI(title="AI Detector API")

_origins_raw = os.getenv("CORS_ORIGINS", _default_origins)
ALLOWED_ORIGINS = [o.strip() for o in _origins_raw.split(",") if o.strip()]

app.add_middleware(
    CORSMiddleware,
    allow_origins=ALLOWED_ORIGINS,
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

app.include_router(auth_router)
app.include_router(submissions_router)
app.include_router(admin_router)
app.include_router(structure_router)
app.include_router(analytics_router)
```

Κάθε router αντιστοιχεί σε μια ομάδα endpoints. Ο `auth_router` χειρίζεται login, register και OTP, ενώ ο `submissions_router` χειρίζεται upload, delete και reports.

Α.2 Σύνδεση Βάσης Δεδομένων

Η σύνδεση γίνεται μέσω SQLAlchemy, με τη διεύθυνση να διαβάζεται από μεταβλητή περιβάλλοντος:

```
# app/db.py
DATABASE_URL = os.getenv("DATABASE_URL", "sqlite:///./app.db")

engine = create_engine(
    DATABASE_URL,
    connect_args=connect_args,
    pool_pre_ping=True,
    pool_recycle=3600,
)
SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)

def get_db():
    db = SessionLocal()
    try:
        yield db
    finally:
        db.close()
```

Η `get_db()` χρησιμοποιείται ως dependency injection στα endpoints, εξασφαλίζοντας ότι κάθε αίτημα αποκτά δική του σύνδεση που κλείνει αυτόματα.

Α.3 Μοντέλα Βάσης Δεδομένων (SQLAlchemy ORM)

Οι πίνακες ορίζονται ως Python classes. Ακολουθεί ο πίνακας submissions:

```
# app/models_db.py (απόσπασμα)
class Submission(Base):
    __tablename__ = "submissions"

    id = Column(Integer, primary_key=True, index=True)
    owner_id = Column(Integer, ForeignKey("users.id"), nullable=False)
    deliverable_id = Column(Integer, nullable=True)
    group_id = Column(Integer, nullable=True)

    original_filename = Column(String, nullable=False)
    stored_path = Column(String, nullable=False)
    file_type = Column(String, nullable=False)
    file_size = Column(Integer, nullable=False)

    final_label = Column(String, nullable=True)
    trustworthiness = Column(Integer, nullable=True)
    percentage_ai = Column(Float, nullable=True)
    summary_json = Column(JSON, nullable=True)
    paragraphs_json = Column(JSON, nullable=True)

    ai_check_status = Column(String, nullable=True)
    ai_check_error = Column(Text, nullable=True)
```

Τα αποτελέσματα ανάλυσης αποθηκεύονται, χωρίς ξεχωριστό πίνακα. Τα πεδία summary_json και paragraphs_json περιέχουν ολόκληρη την αναφορά ως JSON.

Δ.4 Αυθεντικοποίηση (JWT)

Η δημιουργία και επαλήθευση tokens γίνεται στο security.py:

```
# app/security.py (απόσπασμα)
SECRET_KEY = os.getenv("JWT_SECRET_KEY")
ALGORITHM = "HS256"
pwd_context = CryptContext(schemes=["bcrypt"], deprecated="auto")

def create_access_token(sub: str, expires_minutes: int = 1440) -> str:
    expire = datetime.utcnow() + timedelta(minutes=expires_minutes)
    return jwt.encode({"sub": sub, "exp": expire}, SECRET_KEY,
                      algorithm=ALGORITHM)
```

```
def decode_token(token: str) -> Optional[str]:
    try:
        payload = jwt.decode(token, SECRET_KEY, algorithms=[ALGORITHM])
        return payload.get("sub")
    except JWTError:
        return None
```

Η προστασία endpoints γίνεται μέσω dependency:

```
# app/deps_auth.py (απόσπασμα)
def require_approved_user(current_user = Depends(get_current_user)):
    if current_user.status != "APPROVED":
        raise HTTPException(status_code=403,
                             detail="Your request is being reviewed by the admin.")
    return current_user
```

Δ.5 Αναγνώριση Τύπου Αρχείου

Όταν ανεβαίνει αρχείο, το σύστημα αποφασίζει τη μέθοδο ανάλυσης μέσω του file_router:

```
# app/file_router.py
DOCX_EXTENSIONS = {".docx"}
CODE_EXTENSIONS = {".py", ".java", ".c", ".cpp", ".js", ".go", ".php",
".rb", ...}

def detect_file_kind(path) -> str:
    ext = Path(path).suffix.lower()
    if ext in DOCX_EXTENSIONS: return "DOCX"
    if ext in CODE_EXTENSIONS: return "CODE"
    if ext in IGNORE_EXTENSIONS: return "IGNORE"
    return "UNKNOWN"
```

Δ.6 Pipeline Ανίχνευσης AI

Η κεντρική λογική βρίσκεται στη `run_detection`, που δρομολογεί κάθε αρχείο στη σωστή μέθοδο:

```
# app/services_detector.py (απόσπασμα)
def run_detection(file_path, user_thresholds=None):
    kind = detect_file_kind(file_path)

    if kind == "DOCX":
        result = analyze_file_structured(file_path,
                                         custom_thresholds=user_thresholds)
        return {"summary": result.get("summary"),
                "paragraphs": result.get("paragraphs", [])}

    if kind == "CODE":
        # Πρώτα DetectCodeGPT
        detectcodegpt_result =
detect_ai_generated_code_detectcodegpt(file_path)
        if _is_detectcodegpt_success(detectcodegpt_result):
            return detectcodegpt_result
        # Fallback: hybrid (40% rules + 60% ML)
        rules_result = detect_ai_generated_code(file_path)
        ml_result = detect_ai_generated_code_ml(file_path)
        return _build_hybrid_code_result(rules_result, ml_result)

    return {"summary": {"final_label": "IGNORED"}, "paragraphs": []}
```

Για τον κώδικα ακολουθείται στρατηγική δύο επιπέδων: πρώτα DetectCodeGPT, και αν αποτύχει, ο υβριδικός detector (40% rules + 60% ML).

Η ανάλυση εκτελείται ως background task μέσω της `run_ai_check_for_submission`, που φορτώνει τα κατώφλια του χρήστη, τρέχει τη `run_detection` και αποθηκεύει τα αποτελέσματα στη βάση.

Δ.7 Επικοινωνία Frontend-Backend (apiFetch)

Η επικοινωνία γίνεται μέσω μιας κεντρικής συνάρτησης που προσθέτει αυτόματα το JWT token:

```
// lib/api.ts (απόσπασμα)
const API_BASE = process.env.NEXT_PUBLIC_API_BASE ??
"http://127.0.0.1:8000";

export async function apiFetch<T>(path: string, options: RequestInit = {}):
Promise<T> {
  const token = getToken();
  const headers = new Headers(options.headers);
  if (token) headers.set("Authorization", `Bearer ${token}`);

  const res = await fetch(`${API_BASE}${path}`, { ...options, headers });
  if (!res.ok) {
    const data = await res.json().catch(() => null);
    throw new ApiError(data?.detail || `Request failed: ${res.status}`,
res.status);
  }
  return await res.json() as T;
}
```

Αρχεία όπως submissions.ts ορίζουν εξειδικευμένες κλήσεις:

```
// lib/submissions.ts (απόσπασμα)
export async function bulkUploadZip(params: {
  deliverable_id: number;
  zip_file: File;
  run_ai_check?: boolean;
}) {
  const fd = new FormData();
  fd.append("zip_file", params.zip_file);
  const qs = new URLSearchParams();
  qs.set("deliverable_id", String(params.deliverable_id));
  qs.set("run_ai_check", String(params.run_ai_check ?? false));
  return apiFetch<BulkUploadZipResult>(
    `/submissions/bulk-upload-zip?${qs.toString()}`,
    { method: "POST", body: fd }
  );
}
```

Δ.8 Component Structure (React / Next.js)

Το frontend ακολουθεί το App Router pattern. Οι σελίδες ομαδοποιούνται σε route groups, δηλαδή (user) για καθηγητή και (admin) για διαχειριστή. Κάθε group έχει δικό του layout:

```
// app/(user)/layout.tsx
import UserNavbar from "../components/UserNavbar";

export default function UserLayout({ children }: { children:
React.ReactNode }) {
  return (
    <>
      <UserNavbar />
      {children}
    </>
  );
}
```

Κάθε σελίδα μέσα στο (user) εμφανίζει αυτόματα το navbar χωρίς να χρειάζεται κάθε component να το δηλώσει ξεχωριστά.