Εφαρμογή Τουριστικού Οδηγού μίας περιοχής

Πανεπιστήμιο Πατρών  
Πολυτεχνική Σχολή  
Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών  
Πρότζεκτ στο μάθημα Προγραμματισμός Διαδικτύου (ECE\_ΓΚ802)  
Ομάδα 28

Ελευθέριος Γκλιάτης, 1066548

up1066548@ac.upatras.gr

Πάνος Λελάκης, 1083712

up1083712@ac.upatras.gr

Στην παρούσα έκθεση περιγράφεται ο τρόπος με τον οποίο σχεδιάστηκε η εφαρμογή για την υλοποίηση μίας διαδικτυακής εφαρμογής τουριστικού οδηγού μίας περιοχής. Πιο συγκεκριμένα, αναγράφονται τα Use Cases που καλύπτει η εφαρμογή .............. Συνδυάζοντας το θεωρητικό με το πρακτικό τμήμα της εργασίας, σχεδιάστηκε μία βάση δεδομένων SQLite με χρήση κώδικα SQL για την αποθήκευση των στοιχείων τόσο των άρθρων, όσο και των χρηστών της βάσης. Επιπλέον, αναπτύχθηκε πρόγραμμα σε Python που υλοποιεί μία ευρεία συλλογή από εργαλεία για την αλληλεπίδραση μεταξύ χρήστη και βάσης. Αναλύονται τα σημαντικά σημεία του κώδικα SQL στην δημιουργία της βάσης δεδομένων, στην καταχώρηση εγγραφών σε αυτή και στην εκτέλεση SQL queries που προσομοιώνουν την προσδοκώμενη χρήση της, καθώς και τα σημεία του κώδικα Python για τη λειτουργία του προγράμματος. Σε καθεμία από τις παραπάνω περιπτώσεις, δίνονται σε μορφή screenshot τα αποτελέσματα του κώδικα.

Βασικοί Όροι: • Web programming • Full stack development • Database • Javascript

**Επιπλέον λέξεις-κλειδιά:** Τουριστικός οδηγός,

Ελευθέριος Γκλιάτης, Πάνος Λελάκης. 2024. Εφαρμογή Τουριστικού Οδηγού μίας περιοχής: Πανεπιστήμιο Πατρών, Πολυτεχνική Σχολή, Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών, Πρότζεκτ στο μάθημα Προγραμματισμός Διαδικτύου (ECE\_ΓΚ802), Ομάδα 28. Μάιος 26/5, 2024. Πάτρα, Ελλάδα, ................ σελίδες.

1. Περιληψη

Το θέμα της εργασίας είναι η ανάπτυξη μίας διαδικτυακής εφαρμογής τουριστικού οδηγού για μία περιοχή. Σκοπός της ιστοσελίδας της εφαρμογής είναι η πληροφόρηση των χρηστών της για τις υποδομές της περιοχής, τα σημεία και μαγαζιά που ενδέχεται να τους ενδιαφέρουν, καθώς και η διαφήμιση αυτών. Επιπλέον, δίνεται η δυνατότητα και για προγραμματισμό ταξιδιού, μέσω των bookmarks, εφόσον ο χρήστης έχει φτιάξει λογαριασμό και έχει κάνει login.

Όπως ζητήθηκε από την εκφώνηση της εργασίας, όλα τα δεδομένα της εφαρμογής, δηλαδή οι πληροφορίες των τοποθεσιών, των χρηστών και τα bookmarks, είναι αποθηκευμένα σε μία βάση δεδομένων τύπου SQL. Μετά από την full-stack ανάπτυξη της ιστοσελίδας, έγινε δυνατόν το front-end να επικοινωνεί άρτια με το back-end και την βάση δεδομένων, αλλά και το αντίστροφο, ώστε οι ιστοσελίδες να κατασκευάζονται δυναμικά.

Η αυθεντικοποίηση των χρηστών, επιπλέον, κρίθηκε απαραίτητη, ώστε να τους δίνεται η δυνατότητα να αποθηκεύουν τις τοποθεσίες που τους ενδιαφέρουν στα προσωπικά τους bookmarks. Για την ασφαλή επικοινωνία μεταξύ χρηστών και βάσης δεδομένων, καθώς και την προστασία των στοιχείων των λογαριασμών τους, οι sensitive πληροφορίες κωδικοποιούνται.

Καθώς η εκφώνηση της εργασίας θέτει ελάχιστους περιορισμούς όσον αφορά τον σχεδιασμό του μικρόκοσμού της, για τον σκοπό της επιλέχθηκε να χρησιμοποιηθεί ως διαφημιζόμενος τόπος ένα φανταστικό νησί. Έγινε προσπάθεια ώστε να περιέχει ρεαλιστικό πλήθος δεδομένων στην βάση, να προσφέρει πραγματικά χρήσιμες λειτουργίες αλλά και να έχει ελκυστική εμφάνιση.

Το μεγαλύτερο ενδιαφέρον στην εργασία έχει η επιλογή και υλοποίηση των λειτουργιών της εφαρμογής, με βάση την οπτική γωνία του χρήστη. Αποφασίστηκε συνειδητά να δοθεί περισσότερη έμφαση στην πληροφόρηση των χρηστών της ιστοσελίδας, καθώς οι χρήστες της αναμένεται να είναι επισκέπτες του νησιού. Για αυτό, υλοποιήθηκε responsive χάρτης της περιοχής που περιέχει markers με όλες τις τοποθεσίες της. Λόγω του ότι πιθανότατα οι επισκέπτες θα αναζητήσουν αυτές τις πληροφορίες με το smartphone τους όσο θα βρίσκονται στο νησί, επιλέχθηκε να δοθεί περισσότερη βαρύτητα στον mobile-first σχεδιασμό της από τον desktop-based.

1. Τροποσ επιλυσησ

Οι παρακάτω ενότητες περιγράφουν τη μεθοδολογία, την κατανομή των εργασιών και το χρονοδιάγραμμα που ακολουθήθηκε, όπως επίσης και τον τρόπο με τον οποίο αξιολογήθηκε και τον τύπο των δεδομένων που χρησιμοποιήθηκαν.

* 1. Μεθοδολογία

Για την επίλυση του προβλήματος, αρχικά έγινε αναζήτηση στο διαδίκτυο για ένα πρότυπο, πάνω στο οποίο θα μπορούσε κανείς να βασιστεί και να «φανταστεί» πώς δομείται η βάση δεδομένων πίσω από αυτό. Για τον σκοπό αυτό, επιλέχθηκε η ιστοσελίδα https://ieeexplore.ieee.org/ [1], η οποία δείχνει γραφικά ακριβώς όλες τις λειτουργίες που είναι επιθυμητό να έχει η βάση: την αποθήκευση και αλληλεπίδραση με επιστημονικά άρθρα.

Φυσικά, η ιστοσελίδα του ieee xplore προσφέρει ένα τεράστιο εύρος από εργαλεία, η υλοποίηση των οποίων ωστόσο θα έφευγε εκτός του scope του αρχικού προβλήματος. Οπότε, το επόμενο βήμα ήταν να σημειωθούν σε μία λίστα επακριβώς οι λειτουργίες που είναι επιθυμητό να καλυφθούν, σε μορφή Use Cases. Επιπλέον, όσον αφορά τους χρήστες της βάσης, αυτοί χωρίστηκαν σε δύο κατηγορίες (User Roles): τους admins που θα μπορούσαν να είναι οι υπάλληλοι της εταιρείας, και τους non-admins που είναι οι απλοί χρήστες. Ανάλογα με την κατηγορία στην οποία ανήκει ένας χρήστης, έχει πρόσβαση σε διαφορετικά εργαλεία. Στο παρακάτω σχήμα παρουσιάζονται όλα τα προαναφερόμενα Use Cases:

A diagram of a company

Description automatically generated with medium confidence

Εικ. 1: Διάγραμμα με τα Use Cases

Όπως φαίνεται και στο σχήμα, ένας Non-Admin μπορεί μεταξύ άλλων να κάνει μία αναζήτηση για ένα προφίλ ή άρθρο. Μόλις βρει το προφίλ ή άρθρο που ψάχνει, μπορεί να αλληλεπιδράσει μαζί του. Τα Article Interactions ορίζονται ως οι ενέργειες που μπορεί ένας non-admin να κάνει προς ένα άρθρο ενώ τα Profile Interactions είναι οι ενέργειες προς ένα προφίλ ενός άλλου χρήστη, αντίστοιχα. Εκτός από αυτά, έχει την δυνατότητα να σώζει άρθρα που του αρέσουν σε προσωπικές βιβλιοθήκες (Libraries). Επίσης, έχει το δικαίωμα να ανεβάσει ένα draft άρθρο (Draft Article Upload) με σκοπό αυτό αργότερα να δημοσιευθεί (Published) και να είναι διαθέσιμο σε όλους τους χρήστες. Ο ρόλος ενός Admin είναι να αξιολογεί τα άρθρα που ανεβάζουν οι Non-Admins και να επιτρέπει την δημιοσίευσή τους ή όχι.

Έχοντας πλέον ξεκαθαρίσει το εύρος των λειτουργιών που καλύπτονται από την ΒΔ, σχεδιάστηκε το ERD και ακολούθως το Schema, τα οποία αναλύονται στις ομότιτλες ενότητες 3.1 και 3.2 . Η σχεδίασή τους αποδείχθηκε δυσκολότερη από ό,τι θα περίμενε κανείς, διότι ήταν πολλές οι φορές που χρειάστηκε να σβηστεί ό,τι είχε παραχθεί μέχρι στιγμής και να γίνει επιστροφή πίσω στην αρχή. Αυτό συνέβη λόγω του ότι για το ίδιο πρόβλημα ενδέχεται να υπάρχουν πολλαπλές λύσεις διαφορετικές μεταξύ τους, οπότε το ζητούμενο ήταν να επιλεχθεί η βέλτιστη για την εργασία από αυτές.

Αφότου σχεδιάστηκε το ERD και το Schema, το επόμενο βήμα ήταν να αναπτύχθεί κώδικας SQL για την δημιουργία της ΒΔ. Συγκεκριμένα, ο στόχος ήταν να δημιουργηθούν τα tables από τα οποία η βάση αποτελείται και να οριστούν τα columns που περιέχει. Ο τρόπος με τον οποίο επιτεύχθηκε αυτό περιγράφεται στην ενότητα 4.1 .

Έπειτα, η επόμενη πράξη ήταν να εντοπιστούν πιθανά σφάλματα στην βάση, έτσι ώστε με την αντιμετώπιση των οποίων να εξασφαλιστεί η σωστή λειτουργία της. Για να επιτευχθεί αυτό, καταχωρήθηκαν λίγες σε πλήθος εγγραφές σε κάθε table, όπως αναλύεται στην ενότητα 4.2 , και εκτελέστηκαν πολύ απλά SQL queries. Τα δεδομένα που εισήχθηκαν ήταν τέτοια ώστε να είναι γνωστό εκ των προτέρων τα ορθά αποτελέσματα των queries. Επομένως, συγκρίνοντας τα παραγόμενα αποτελέσματα με τα αναμενόμενα, διορθώθηκαν τα σφάλματα που είχαν προκύψει.

Έχοντας επιβεβαιώσει την σωστή λειτουργία της βάσης, εκτελέστηκε ένα σύνολο από λίγο πιο περίπλοκα SQL queries, τα οποία προσομοιώνουν την προσδοκώμενη χρήση της. Τα ερωτήματα αυτά καθώς και τα αποτελέσματα που λήφθηκαν με την εκτέλεσή τους αναφέρονται στην ενότητα 4.3 .

Τέλος, αναπτύχθηκε πρόγραμμα σε Python που να περιέχει τα εργαλεία που παρουσιάστηκαν ως Use-Cases και να τα προσφέρει σε έναν εξωτερικό χρήστη. Κρίθηκε σημαντικό να προστεθούν ορισμένοι έλεγχοι, έτσι ώστε να εξασφαλίζεται η σωστή εισαγωγή και τροποποίηση δεδομένων στην ΒΔ. Κάποιες από τις λειτουργίες επιλέχθηκε να μην υλοποιηθούν σε βάθος, διότι θα απαιτούσαν περίπλοκο κώδικα και θα ξέφευγαν εκτός του scope της εργασίας.

* 1. Αξιολόγηση

Έχοντας ολοκληρώσει την δημιουργία της βάσης και του προγράμματος σε Python, έγινε μία ανασκόπηση όλων των τμημάτων της εργασίας, με σκοπό την εσωτερική αξιολόγησή της.

Για την αξιολόγηση της εργασίας, χρησιμοποιήθηκαν τα παρακάτω κριτήρια και απαντήθηκαν οι ερωτήσεις που περιέχει το καθένα, ταξινομημένα με φθίνουσα σειρά προτεραιότητας (1 = υψηλότερη σημασία, 4 = χαμηλότερη σημασία):

1. Ορθότητα

• Είναι το ERD και το Schema σωστά σχεδιασμένα;

• Τα tables στην βάση δημιουργούνται σωστά, με τις κατάλληλες σχέσεις μεταξύ τους;

• Είναι σωστά τα αποτελέσματα που παράγουν τα queries;

2. Πληρότητα

• Έχουν καλυφθεί όλα τα απαραίτητα πεδία που απαιτούνται για την υλοποίηση όλων των λειτουργιών της εφαρμογής;

• Τα SQL queries και το πρόγραμμα σε Python καλύπτουν τα Use-Cases;

3. Κατανόηση του θέματος και των ορίων

• Έγινε σωστή επιλογή των ορίων της εργασίας, δηλαδή μέχρι τι εύρος καλύπτει; Το ιδανικό εύρος είναι τέτοιο ώστε να καλύπτονται όλες οι απαιτήσεις του προβλήματος, χωρίς όμως να ξεφεύγει και να δίνει έμφαση σε θέματα που να μην είναι απαραίτητα για την εκπόνησή της.

4. Δομή και καθαρότητα κώδικα

• Ο κώδικας σε SQL και Python είναι κατανοητός και ευανάγνωστος;

• Υπάρχουν επαρκή σχόλια στον κώδικα, ώστε κάποιος άλλος να καταλαβαίνει τον σκοπό κάθε εντολής;

Η μεθοδολογία που επιλέχθηκε να ακολουθηθεί διαμορφώθηκε έτσι ώστε η εργασία να πληροί έως έναν ικανοποιητικό βαθμό τα παραπάνω κριτήρια αξιολόγησης.

* 1. Κατανομή εργασιών

Η εργασία χωρίστηκε στα παρακάτω subtasks:

1. Κατανόηση προβλήματος και Use Cases

2. ERD

3. Schema

4. Προετοιμασία (powerpoint) για την ενδιάμεση παρουσίαση της εργασίας

5. Δημιουργία βάσης με κώδικα SQL

6. Εισαγωγή δεδομένων στην βάση

7. SQL queries για τυπικές αναζητήσεις

8. Πρόγραμμα Python

9. Έκθεση

10. Τελική παρουσίαση (powerpoint) για την τελική εξέταση της εργασίας

Και τα δύο μέλη της ομάδας πήραν μέρος σε όλα τα subtasks, ωστόσο ο καθένας ανέλαβε από ένα υποσύνολο αυτών ως «κύριες» ασχολίες του. Μόλις το μέλος που είχε αναλάβει ένα subtask ήταν κοντά στην ολοκλήρωσή του, το άλλο μέλος έβλεπε την λύση και συνεργαζόταν μαζί του ώστε να τελειοποιηθεί.

Τα subtasks που ανέλαβε το κάθε μέλος είναι τα εξής:

Εξ’ ολοκλήρου από κοινού:

1)Κατανόηση προβλήματος και Use Cases

4)Προετοιμασία (powerpoint) για την ενδιάμεση παρουσίαση της εργασίας

8)Πρόγραμμα Python

9)Έκθεση

Αναστάσιος Κελεσίδης:

3)Schema

5)Δημιουργία βάσης με κώδικα SQL

6)Εισαγωγή δεδομένων στην βάση

Πάνος Λελάκης:

2)ERD

7)SQL queries για τυπικές αναζητήσεις

10)Τελική παρουσίαση (powerpoint) για την τελική εξέταση της εργασίας

* 1. Χρονοδιάγραμμα

Τα subtasks που αναφέρθηκαν στην ενότητα 2.3 μπορούν να τοποθετηθούν χρονικά ως εξής:

A graph with multiple colored squares

Description automatically generated with medium confidence

Εικ. 2: Χρονοδιάγραμμα

* 1. Δεδομένα

Η εισαγωγή δεδομένων στην ΒΔ, αφότου χτίστηκε ο σκελετός της, κρίθηκε απαραίτητη. Όχι μόνο μπορέσαμε αργότερα να πετύχουμε την σωστή λειτουργία της εντοπίζοντας bugs, αλλά μπορέσαμε και να αναδείξουμε τις χρήσεις της υλοποιώντας συγκεκριμένα use cases.

Επιλέξαμε να χρησιμοποιήσουμε ρεαλιστικά δεδομένα, τόσο για τα στοιχεία των χρηστών της ΒΔ όσο και για τα άρθρα που αυτή περιέχει.

• Για τα στοιχεία των χρηστών της χρησιμοποιήθηκε το εργαλείο www.fakepersongenerator.com , το οποίο κάνει generate μη υπαρκτούς ανθρώπους, παρέχοντας προσωπικά στοιχεία (πχ όνομα, ηλικία, επάγγελμα κλπ) καθώς και στοιχεία login (username, password, email κλπ).

• Για τις πληροφορίες των άρθρων χρησιμοποιήθηκε η ιστοσελίδα www.kaggle.com/datasets?search=article , η οποία περιέχει έτοιμα datasets (πίνακες) με ρεαλιστικά στοιχεία για άρθρα. Τα ακριβή datasets που χρησιμοποιήθηκαν για τον σκοπό της εργασίας είναι:

<https://www.kaggle.com/datasets/viniciuslambert/medium-2021-data-science-articles-dataset> [2]

<https://www.kaggle.com/code/cocolicoq4/sampling-500k-small-dataset-data-cleaning> [3]

<https://www.kaggle.com/datasets/aryansingh0909/nyt-articles-21m-2000-present> [4]

Για κάποια από τα columns (πχ file directory) χρειάστηκε να βάλουμε αυθαίρετες τιμές (δηλαδή δικές μας τιμές) διότι δεν βρέθηκε dataset που να τα περιέχει όλα.

• Για τα δεδομένα όλων των υπόλοιπων tables (Review, Saved, Interaction κλπ) καταχωρήσαμε δικές μας εγγραφές με ρεαλιστικά στοιχεία, με τέτοιο τρόπο όμως ώστε να βοηθούν στα αποτελέσματα των SQL queries που αναφέρονται στην ενότητα 4.3 .

Είναι σημαντικό να σημειωθεί ότι σε συγκεκριμένα columns, παρόλο που δεν υπάρχει έλεγχος από την SQL, τα δεδομένα πρέπει να έχουν τιμή από ένα σύνολο τιμών. Πιο συγκεκριμένα:

1) στο table Interaction, στο column Type: εδώ αποθηκεύεται ο τύπος της αλληλεπίδρασης και θα πρέπει να πάρει μία από τις εξής τιμές:

a. Για άρθρα: View, Share, Like, Comment, Download

b. Για προφίλ: View, Share, Follow

c. Για βιβλιοθήκες: Rename, Save, Create, Delete, Remove

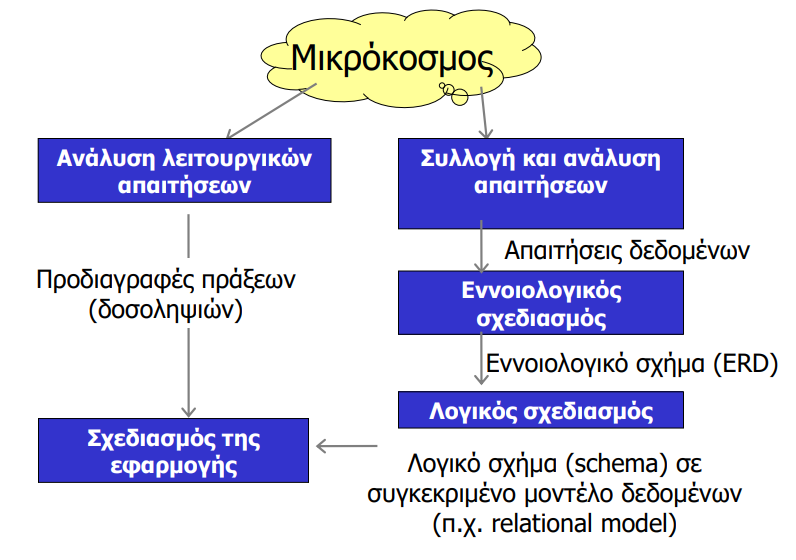
2) στο table Draft, στο column Status: εδώ αποθηκεύεται η κατάσταση στην οποία βρίσκεται το draft article, την οποία αλλάζει μόνο ένας admin, και θα πρέπει να έχει πάντα μία από τις εξής τιμές: Pending, Rejected, Accepted, Published.

1. ΘΕΩΡΗΤΙΚΟΣ ΣΧΕΔΙΑΣΜΟΣ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Στις παρακάτω ενότητες περιγράφεται ο τρόπος με τον οποίο σχεδιάστηκε η βάση δεδομένων σε θεωρητικό επίπεδο, δηλαδή αναλύονται το ERD και το Schema.

* 1. Διάγραμμα οντοτήτων-συσχετίσεων (ERD)

Αφού προσδιορίστηκε και περιγράφηκε η ακριβής δομή της ΒΔ, σχεδιάστηκε το διάγραμμα οντοτήτων-συσχετίσεων (ERD), το οποίο αποτελεί σημαντικό κομμάτι για την πλήρη επισκόπηση του μικρόκοσμου της εφαρμογής μας.



Εικ. 3: Υλοποίηση μικρόκοσμου

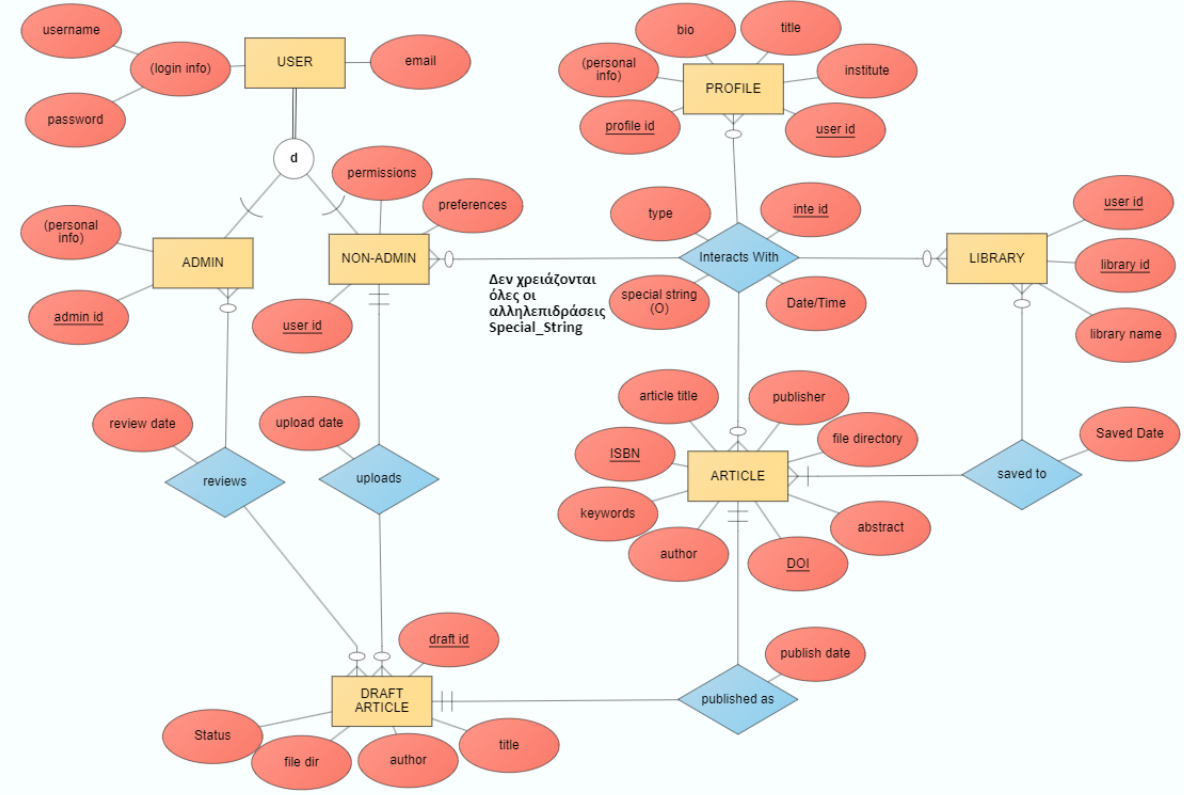
Αρχικά, καταγράφηκαν οι βασικές έννοιες (οντότητες) της εφαρμογής (Χρήστες, Άρθρο κλπ) και έπειτα τα γνωρίσματα τους, τα οποία κρίθηκαν απαραίτητα σύμφωνα πάντα με την δομή και τις παραδοχές της εφαρμογής που έγιναν. Έπειτα, προσθέτοντας τις κατάλληλες συσχετίσεις μεταξύ των οντοτήτων, υλοποιήθηκε μια πρώτη δομή του εννοιολογικού μοντέλου, αναπαριστώντας για παράδειγμα τη διαδικασία δημοσιοποίησης ενός άρθρου, από την υποβολή ενός χρήστη για αξιολόγηση έως την επίσημη δημοσίευση του άρθρου στην εφαρμογή.

A diagram of a draft analysis

Description automatically generated

Εικ. 4: Διαδικασία δημοσίευσης άρθρου

Αφού αποτυπώθηκαν όλες οι λειτουργίες της εφαρμογής στο εννοιολογικό μοντέλο, προστέθηκαν τελικά και οι πληθυκότητες των σχέσεων ολοκληρώνοντας το διάγραμμα.



Εικ. 5: ERD

* 1. Λογικό σχεσιακό μοντέλο (Schema)

Σε πλήρη αντιστοιχία με το εννοιολογικό μοντέλο, δημιουργήθηκε και το λογικό μοντέλο της βάσης δεδομένων, ακολουθώντας κατάλληλα την θεωρία για τα ξένα κλειδιά, τις υποκλάσσεις και όλους τους σχετικούς ορισμούς.

A screenshot of a computer

Description automatically generated

Εικ. 6: Schema

1. ΔΗμιουργια και χρηση βασησ δεδομενων

Στις παρακάτω ενότητες περιγράφεται ο τρόπος με τον οποίο δημιουργήθηκε η βάση δεδομένων SQLite, δηλαδή πώς μεταφράστηκε το ERD και το Schema σε κώδικα SQL. Επίσης, εξηγείται ο κώδικας που αναπτύχθηκε για την επαλήθευση της σωστής λειτουργίας της, ο κώδικας τυπικών queries και το πρόγραμμα σε Python.

* 1. Δημιουργία ΒΔ

Ο σκελετός της βάσης αποτελείται από 10 tables: Admin, Non\_Admin, Review, Profile, Article, Draft, Library, Publication, Saved και Interaction. Βάσει θεωρίας, απαιτείται ένα table για την κάθε οντότητα και συσχέτιση. Η δομή αυτή προκύπτει άμεσα από το schema, που παρουσιάστηκε στην ενότητα 3.2 . Επιλέχθηκε να μην δημιουργηθεί table για την συσχέτιση uploads που εκφράζει την δυνατότητα ενός χρήστη να ανεβάσει ένα draft article και αντ’ αυτού προστέθηκε απλώς το non\_admin\_id στο table draft\_article, ώστε να αποφευχθεί η αχρείαστη επανάληψη εγγραφών.

Για να χτιστεί η ΒΔ, γράφηκε κώδικας SQL έχοντας ως πρότυπο τον κώδικα των παραδειγμάτων που παρουσιάζονται στις αντίστοιχες διαφάνειες του μαθήματος. Όλα τα tables μοιράζονται κοινό κώδικα, οπότε παρακάτω εξηγείται ο κώδικας για τη δημιουργία ενός μόνο από τα tables (Profile table) που περιέχει όλα τα σημαντικά σημεία κώδικα που εμφανίζονται.

Για τη δημιουργία λοιπόν του table Profile, εκτελούμε τον παρακάτω κώδικα SQL:

DROP TABLE IF EXISTS "Profile";

CREATE TABLE IF NOT EXISTS "Profile" (

"Profile\_ID" int(7) NOT NULL DEFAULT '0',

"Non\_Admin\_ID" int(7) NOT NULL DEFAULT '0',

"Bio" text NOT NULL DEFAULT '',

"Title" varchar(20) DEFAULT '',

"Institute" varchar(30) DEFAULT '',

"Last\_Name" varchar(20) NOT NULL DEFAULT '',

"First\_Name" varchar(20) NOT NULL DEFAULT '',

"Age" int(2) NOT NULL DEFAULT '0',

"Sex" varchar(6) NOT NULL DEFAULT '',

CONSTRAINT "Non\_Admin\_ID\_FK" FOREIGN KEY("Non\_Admin\_ID") REFERENCES

"Non\_Admin"("Non\_Admin\_ID") ON UPDATE CASCADE ON DELETE CASCADE,

PRIMARY KEY("Profile\_ID")

);

Αρχικά, με την εντολή DROP TABLE IF EXISTS ελέγχεται αν ήδη υπάρχει table με αυτό το όνομα. Αν υπάρχει, τότε αυτό σβήνεται.

Με την εντολή CREATE TABLE IF NOT EXISTS πάλι ελέγχεται αν ήδη υπάρχει table με αυτό το όνομα. Αυτή τη φορά όμως, αν ΔΕΝ υπάρχει, τότε δημιουργείται ένα.

Στην παρένθεση τοποθετούνται όλα τα columns που είναι επιθυμητό να περιέχει το table, καθώς και τι τύπου δεδομένα αποθηκεύονται στο καθένα. Επίσης, με το NOT NULL DEFAULT τίθενται κάποιοι περιορισμοί για τις τιμές που μπορεί να πάρει το πεδίο της κάθε εγγραφής.

Εάν λάβει κανείς για παράδειγμα το πρώτο column, που είναι το Profile\_ID, παρατηρεί ότι σε αυτό το column θα αποθηκεύονται δεδομένα τύπου int (integer), μεγέθους 7 (δηλαδή θα είναι μέχρι 7 ψηφία). Με το NOT NULL δεν επιτρέπεται να αποθηκευτούν NULL τιμές στο column αυτό. Με το DEFAULT ‘0’ ορίζεται ότι σε περίπτωση που κάποιος προσθέσει μία εγγραφή αλλά δεν δώσει τιμή για το πεδίο αυτό, τότε η βάση του δίνει by default τιμή 0. Οι παραπάνω 2 περιορισμοί είναι σημαντικοί επειδή το Profile\_ID έχει τον ρόλο ενός κλειδιού, και βάσει ορισμού είναι απαραίτητο κάθε εγγραφή να έχει το μοναδικό κλειδί της ώστε να ξεχωρίζει από τις άλλες. Με τους περιορισμούς αυτούς, οπότε, εξασφαλίζεται ακριβώς αυτό.

Αφότου προστέθηκαν όλα τα columns που είναι επιθυμητό να υπάρχουν στο table, τέθηκαν συγκεκριμένοι περιορισμοί για τα κλειδιά. Πιο συγκεκριμένα, εξηγείται στη βάση: ποια από τα columns που δημιουργήθηκαν είναι πρωτεύοντα κλειδιά και ποια είναι ξένα, τα ξένα κλειδιά από πού προέρχονται, καθώς και το τι γίνεται αν η τιμή ενός ξένου κλειδιού αλλάξει ή διαγραφεί.

Με το CONSTRAINT “…” FOREIGN KEY(“Non\_Admin\_ID”) REFERENCES “Non\_Admin” (“…”) ορίζεται το πεδίο Non\_Admin\_ID ως ξένο κλειδί (foreign key) που προέρχεται από το table Non\_Admin (references).

Με το ON UPDATE CASCADE ON DELETE CASCADE εξασφαλίζεται ότι σε περίπτωση που το Non\_Admin\_ID μιας εγγραφής αλλάξει τιμή (on update) ή διαγραφεί (on delete) στο parent table (δηλαδή στο table που το Non\_Admin\_ID είναι πρωτεύον κλειδί, δηλαδή στο table Non\_Admin), τότε θα γίνει ακριβώς η ίδια αλλαγή αυτόματα στο table Profile (cascade).

Τέλος, με το PRIMARY KEY(“Profile\_ID”) τίθεται το column Profile\_ID ως πρωτεύον κλειδί για το table Profile. Αυτό σημαίνει ότι κάθε εγγραφή θα πρέπει να έχει ξεχωριστό Profile\_ID και αν ένα άλλο table περιέχει αυτό το πεδίο, θα πρέπει να το ορίσει ως ξένο κλειδί με τον ίδιο τρόπο που ορίστηκε το Non\_Admin\_ID ως ξένο κλειδί.

Για τη δημιουργία όλων των υπόλοιπων tables χρησιμοποιείται πανομοιότυπος κώδικας, με μόνες διαφορές τα columns και τα κλειδιά. Ο πλήρης κώδικας για την δημιουργία όλων των tables δίνεται στο αρχείο db\_creation.sql .

* 1. Εισαγωγή δεδομένων στην ΒΔ

Αφότου χτίστηκε η βάση, θα πρέπει να γίνει η εισαγωγή ενός συνόλου δεδομένων σε αυτή ώστε αργότερα να μπορεί κανείς να αλληλεπιδράσει μαζί της και να εντοπιστούν ενδεχόμενα bugs.

Αρχικά, πραγματοποιείται εισαγωγή λίγων σε αριθμό εγγραφών σε κάθε table, ώστε να επαληθευτεί ότι όλα λειτουργούν σωστά. Η εισαγωγή δεδομένων σε κάθε table γίνεται με τον ίδιο ακριβώς τρόπο, οπότε παρακάτω εξηγείται η εισαγωγή δεδομένων σε ένα μόνο table.

Για χάρη του παραδείγματος, καταχωρούνται 5 εγγραφές στο table Admin με τον εξής τρόπο:

INSERT INTO 'Admin' ('Email', 'Username', 'Password', 'Admin\_ID', 'Last\_Name', 'First\_Name',

'Address', 'Phone\_Number')

VALUES

('admin1@example.com', 'admin1', 'adminpass1', 1, 'Smith', 'John', '123 Main St', 1234567890),

('admin2@example.com', 'admin2', 'adminpass2', 2, 'Doe', 'Jane', '456 Oak St', 9876543210),

('admin3@example.com', 'admin3', 'adminpass3', 3, 'Johnson', 'Robert', '789 Elm St', 4567890123),

('admin4@example.com', 'admin4', 'adminpass4', 4, 'Williams', 'Emily', '101 Pine St', 3210987654),

('admin5@example.com', 'admin5', 'adminpass5', 5, 'Taylor', 'Ava', '202 Cedar St', 8765432109);

Με το INSERT INTO ‘Admin’ σηματοδοτείται η πρόσθεση εγγραφών στο table Admin και στην παρένθεση δίπλα αναγράφονται τα columns που αυτό περιέχει.

Στο VALUES (…) περιέχονται όλες οι εγγραφές που εισάγονται. Αν λάβει κανείς για παράδειγμα την πρώτη εγγραφή:

('admin1@example.com', 'admin1', 'adminpass1', 1, 'Smith', 'John', '123 Main St', 1234567890)

Ορίζεται η τιμή που παίρνει σε κάθε column με την ίδια σειρά που αυτά έχουν τοποθετηθεί όταν δημιουργήθηκε το εκάστοτε table. Ως πρώτο πεδίο θα πρέπει να δοθεί το email (εδώ admin1@example.com), ως δεύτερο το username (εδώ admin1), ως τρίτο το password (εδώ adminpass1), ως τέταρτο το admin\_id (εδώ 1) που είναι πρωτεύον κλειδί και άρα δεν μπορεί κάποια άλλη εγγραφή να λάβει την ίδια τιμή για κλειδί, ως πέμπτο το επίθετο κλπ.

Ο κώδικας για την εισαγωγή δεδομένων στα υπόλοιπα tables είναι πανομοιότυπος. Ο πλήρης κώδικας για την καταχώρηση εγγραφών σε όλα τα tables δίνεται στο αρχείο db\_populate.sql .

* 1. Τυπικές αναζητήσεις στην ΒΔ

Παρακάτω παρατίθενται μερικά SQL queries που δείχνουν την προσδοκώμενη χρήση της ΒΔ. Για το κάθε query, δίνεται μία σύντομη περιγραφή για την χρήση του όπως και τα αποτελέσματα που επιστρέφει. Όλα τα τμήματα του κώδικα δίνονται και στο αρχείο sql\_queries.sql με όλα τα απαραίτητα σχόλια.

1: Όλα τα draft articles που έχουν status ‘Pending’. Το query αυτό έχει σημασία διότι επιστρέφει όλα τα draft articles που οι admins (υπάλληλοι της εταιρείας) πρέπει να αξιολογήσουν, επομένως εκείνοι ξέρουν εξαρχής ποιος είναι ο όγκος εργασίας τους.

SQL: SELECT D.Draft\_ID, D.Title

FROM Draft AS D

WHERE D.Status = 'Pending'

Αποτελέσματα:

A close up of a label

Description automatically generated

2: Όλα τα draft articles που ο admin με admin\_id = 2 έχει αξιολογήσει μέσα στο 2023.

SQL: SELECT R.Draft\_ID, R.Date

FROM Review AS R

WHERE R.Admin\_ID = '2' AND strftime('%Y', R.Date) = '2023'

Αποτελέσματα:

A close up of numbers

Description automatically generated

3: Τα top 5 draft articles που έχουν status ‘Pending’ και περιμένουν για τον περισσότερο καιρό για να αξιολογηθούν. Τα άρθρα που περιμένουν τις περισσότερες μέρες εμφανίζονται πρώτα.

SQL: SELECT D.Draft\_ID, D.Upload\_Date,

CAST (julianday(date('now')) - julianday(date(D.Upload\_Date)) AS INTEGER) AS Days\_Waiting

FROM Draft AS D

WHERE D.Status = 'Pending'

ORDER BY D.Upload\_Date ASC

LIMIT 5

Αποτελέσματα:

A screenshot of a upload date

Description automatically generated

4: Ο αριθμός των likes που έχει λάβει το article με article\_id = 4.

SQL: SELECT COUNT(I.Inte\_ID) AS Likes

FROM Article AS A

JOIN Interaction AS I ON A.Article\_ID = I.Article\_ID

WHERE I.Type = 'Like' AND A.Article\_ID = '4'

Αποτελέσματα:

A white background with black text

Description automatically generated

5: Top 5 articles με τα περισσότερα likes. Ταξινομούνται με φθίνοντα αριθμό likes.

SQL: SELECT A.Article\_ID, A.Title, COUNT(I.Inte\_ID) AS Likes

FROM Article AS A

JOIN Interaction AS I ON A.Article\_ID = I.Article\_ID

WHERE I.Type = 'Like'

GROUP BY I.Article\_ID

ORDER BY COUNT(I.Inte\_ID) DESC

LIMIT 5

Αποτελέσματα:

A white background with black text

Description automatically generated

6: Όλα τα articles που έχουν σχέση με τεχνητή νοημοσύνη και τεχνολογία (δηλαδή στα keywords να περιέχονται οι όροι Artificial Intelligence και Technology).

SQL: SELECT A.Article\_ID, A.Title, A.Authors

FROM Article AS A

WHERE (A.Keywords LIKE '%ai%' OR A.Keywords LIKE '%artificial intelligence%') AND

A.Keywords LIKE '%technology%'

Αποτελέσματα:

A white background with black text

Description automatically generated

7: Freshly-published articles (δηλαδή να έχουν δημοσιευθεί τις τελευταίες 7 ημέρες). Τα πιο πρόσφατα δημιοσιευμένα articles εμφανίζονται πρώτα.

SQL: SELECT A.Article\_ID, A.Title, P.Publi\_Date, A.Authors

FROM Article AS A

JOIN Publication AS P ON A.Article\_ID = P.Article\_ID

WHERE julianday('now') - julianday(P.Publi\_Date) <= 7

ORDER BY P.Publi\_Date DESC

Αποτελέσματα:

A white background with black text

Description automatically generated

8: Top 10 trending articles (δηλαδή να είναι freshly-published, ταξινομημένα με τον αριθμό των views που έχουν πάρει). Ταξινομούνται με φθίνοντα αριθμό views.

SQL: SELECT A.Article\_ID, A.Title, P.Publi\_Date, A.Authors, COUNT(I.Inte\_ID) AS Views

FROM Article AS A

JOIN Publication AS P ON A.Article\_ID = P.Article\_ID

JOIN Interaction AS I ON A.Article\_ID = I.Article\_ID

WHERE julianday(date('now')) - julianday(date(P.Publi\_Date)) <= 7

AND I.Type = 'View'

GROUP BY I.Article\_ID

ORDER BY COUNT(I.Inte\_ID) DESC

LIMIT 10

Αποτελέσματα:

A group of numbers on a white background

Description automatically generated

1. χρηση βασησ και προγραμμα python

Στην παρακάτω ενότητα αναλύεται το πρόγραμμα Python που αναπτύχθηκε για την υλοποίηση των λειτουργιών της εφαρμογής, συνοδευόμενο από παραδείγματα χρήσης του.

* 1. Λειτουργίες Admin

Κατά την υλοποίηση της τελευταίας φάσης της εργασίας δημιουργήθηκε μια εφαρμογή σε γλώσσα Python, την οποία χρησιμοποιούν είτε ένας admin για να αξιολογήσει draft articles, είτε ένας non-admin για να αλληλεπιδράσει με την βάση. Η εφαρμογή στηρίχτηκε στην βάση δεδομένων και στον κώδικα SQL που αναπτύχθηκαν, όπως αναφέρθηκε παραπάνω, και ανανεώνει τα δεδομένα (update τα tables), αναλόγως με τις ενέργειες του εκάστοτε χρήστη. Ο πλήρης κώδικας που αναπτύχθηκε δίνεται στο αρχείο db\_program.py .

Στην αρχή της εφαρμογής ζητείται από τον χρήστη να πληκτρολογήσει αν θέλει να συνδεθεί ως Admin ή ως Non-Admin. Έπειτα, για να συνδεθεί πρέπει να πληκτρολογήσει τα username & password, όπως αυτά είναι αποθηκευμένα στην βάση δεδομένων. Αν δεν βάλει τα σωστά δεδομένα η εφαρμογή δεν τον αφήνει να προχωρήσει. Μετά την σωστή ταυτοποίηση, η εφαρμογή τον καλωσορίζει ανακτώντας το όνομά του από την βάση και του απαριθμεί τις επιλογές του.

A screenshot of a computer

Description automatically generated

Εικ. 7: Στοιχεία login ενός admin για το παράδειγμα

A screenshot of a computer program

Description automatically generated

Εικ. 8: Επιτυχής είσοδος του admin

A black background with white text

Description automatically generated

Εικ. 9: Ανεπιτυχής είσοδος του admin

Μετά την επαλήθευσή του, ένας Admin έχει τις εξής επιλογές:

1. να αξιολογήσει ένα ανεβασμένο Draft Άρθρο (επιλογή ‘1’)

2. να πραγματοποιήσει κάποια άλλη λειτουργία η οποία δεν υλοποιείται στα πλαίσια αυτής της εργασίας (επιλογή ‘2’)

3. να βγει από την εφαρμογή (επιλογή ‘3’).

Με την επιλογή ‘1’ ο Admin καλείται να διαλέξει το Draft το οποίο επιθυμεί να αξιολογήσει χρησιμοποιώντας το Draft\_ID. Έπειτα, μόλις αλλάξει την κατάσταση του Draft ενημερώνονται στην ΒΔ οι πίνακες Draft (το Draft.Status) και Review (προστίθεται μια αξιολόγηση στον πίνακα Review).

A screen shot of a computer

Description automatically generatedA screenshot of a table

Description automatically generated

Εικ. 10: Αξιολόγηση του Draft1 από τον Admin1

* 1. Λειτουργίες non-admin

Στον non-admin παρέχεται πρόσβαση σε μία ευρεία συλλογή από εργαλεία, με τα οποία μπορεί αυτός να αλληλεπιδράσει με την ΒΔ. Πιο συγκεκριμένα, μετά την επαλήθευσή του ένας non-admin έχει τις εξής επιλογές:

1. Να δει τα διαθέσιμα δημοσιοποιημένα άρθρα, τις πληροφορίες τους και αν θέλει να επιλέξει ένα (χρησιμοποιώντας το Article\_ID) και να αλληλεπιδράσει με αυτό (επιλογή ‘1’)  
Επιλέγοντας ένα άρθρο για να αλληλεπιδράσει ο χρήστης έχει τις εξής επιλογές:

* Να διαβάσει το άρθρο (τυπώνεται το κείμενο από κάποιο storage)
* Να κατεβάσει το άρθρο
* Να κάνει ‘Like’ στο άρθρο
* Να σχολιάσει το άρθρο
* Να κοινοποιήσει το άρθρο (το πρόγραμμα κάνει generate ένα URL και αυτό αντιγράφεται στο clipboard)
* Να αποθηκεύσει το άρθρο σε μια βιβλιοθήκη του (αν δεν έχει βιβλιοθήκες του δίνεται η επιλογή να δημιουργήσει μία, ενημερώνοντας κατάλληλα την βάση δεδομένων).

Μετά από οποιαδήποτε αλληλεπίδραση από τις παραπάνω ενημερώνεται ο πίνακας Interaction της βάσης δεδομένων κάθε φορά με τις κατάλληλες πληροφορίες(ειδικά στο Special\_String για τον σχολιασμό αποθηκεύεται το σχόλιο).

2. Να ανεβάσει ένα Draft άρθρο προς δημοσιοποίηση (επιλογή ‘2’)

3. Να δημοσιοποιήσει ένα ‘Accepted’ άρθρο το οποίο έχει αξιολογηθεί (αν υπάρχει) (επιλογή ‘3’)

4. Να διαχειριστεί τις προσωπικές του βιβλιοθήκες (επιλογή ‘4’)

5. Να βγει από την εφαρμογή (επιλογή ‘5’)

Λόγω του εύρους των επιλογών που έχει ο non-admin, παρακάτω εξηγείται ένα μόνο παράδειγμα χρήσης. Για χάρη του παραδείγματος, αναλύεται η περίπτωση που ο χρήστης επιλέξει να διαχειριστεί τις προσωπικές του βιβλιοθήκες (επιλογή ‘4’).

A screenshot of a computer program

Description automatically generated

Εικ. 11: Μενού non-admin και παράδειγμα χρήσης επιλογής 6 (αποθήκευση άρθρου σε βιβλιοθήκη)

1. συνοψη

Η ανάπτυξη και υλοποίηση της Βάσης Δεδομένων για την αποθήκευση και ανάκτηση επιστημονικών άρθρων και περιοδικών αξιοποίησε τον συνδυασμό τεχνολογιών και εργαλείων ανάπτυξης: το ERD, το schema, τη βάση δεδομένων τύπου SQLite και την επικοινωνία αυτής με τη γλώσσα Python.

Παρόλο που είναι περιορισμένη σε εύρος, προσέφερε χρήσιμη και πολύτιμη εμπειρία με τις τεχνολογίες αυτές. Δεν δόθηκαν ακριβή ζητούμενα από την εκφώνηση της εργασίας, πράγμα που κατέστησε την αναζήτηση στο διαδίκτυο, για τον τρόπο με τον οποίο είναι δομημένες πραγματικές βάσεις που εκτελούν τις ίδιες λειτουργίες, απαραίτητη.

Τέλος, η βάση και το πρόγραμμα σε python σχεδιάστηκαν με απλές, μη-επαγγελματικές προδιαγραφές ως προς το εύρος των δυνατοτήτων που προσφέρουν, καθώς δόθηκε περισσότερη έμφαση στην σωστή λειτουργία τους. Φυσικά, τα όριά τους μπορούν να διευρυνθούν σημαντικά μετά από την ανάλυση των απαιτήσεων της εταιρείας που πρόκειται να τα χρησιμοποιήσει. Ορισμένα παραδείγματα επιπλέον λειτουργειών που θα μπορούσαν να προστεθούν είναι: η δυνατότητα για common (shared) libraries, public chat rooms, article recommendation engine, subscription/membership, εργαλεία για τους admins για την απευθείας τροποποίηση των στοιχείων ενός άρθρου κλπ.

ΠΗΓΕΣ

1. <https://ieeexplore.ieee.org/> (πρότυπο για λειτουργίες εφαρμογής)
2. <https://www.kaggle.com/datasets/viniciuslambert/medium-2021-data-science-articles-dataset> (dataset για δεδομένα βάσης)
3. <https://www.kaggle.com/code/cocolicoq4/sampling-500k-small-dataset-data-cleaning> (dataset για δεδομένα βάσης)

1. <https://www.kaggle.com/datasets/aryansingh0909/nyt-articles-21m-2000-present> (dataset για δεδομένα βάσης)