



## **Εθνικό Μετσόβιο Πολυτεχνείο**

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

### **Βάσεις Δεδομένων**

Αναφορά Εξαμηνιαίας Εργασίας

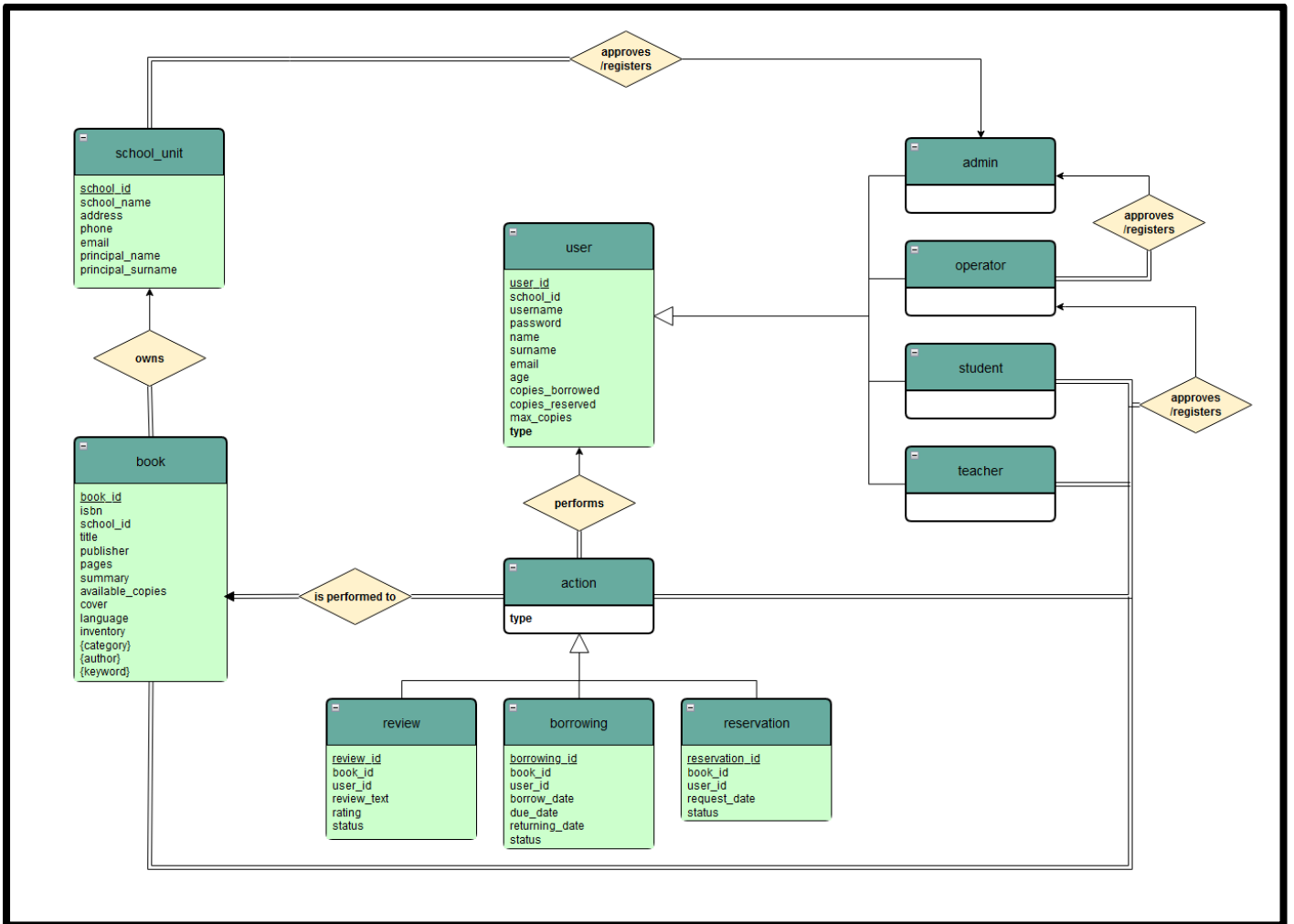
Εαρινό Εξάμηνο 2022-2023

### **Ομάδα Project 17**

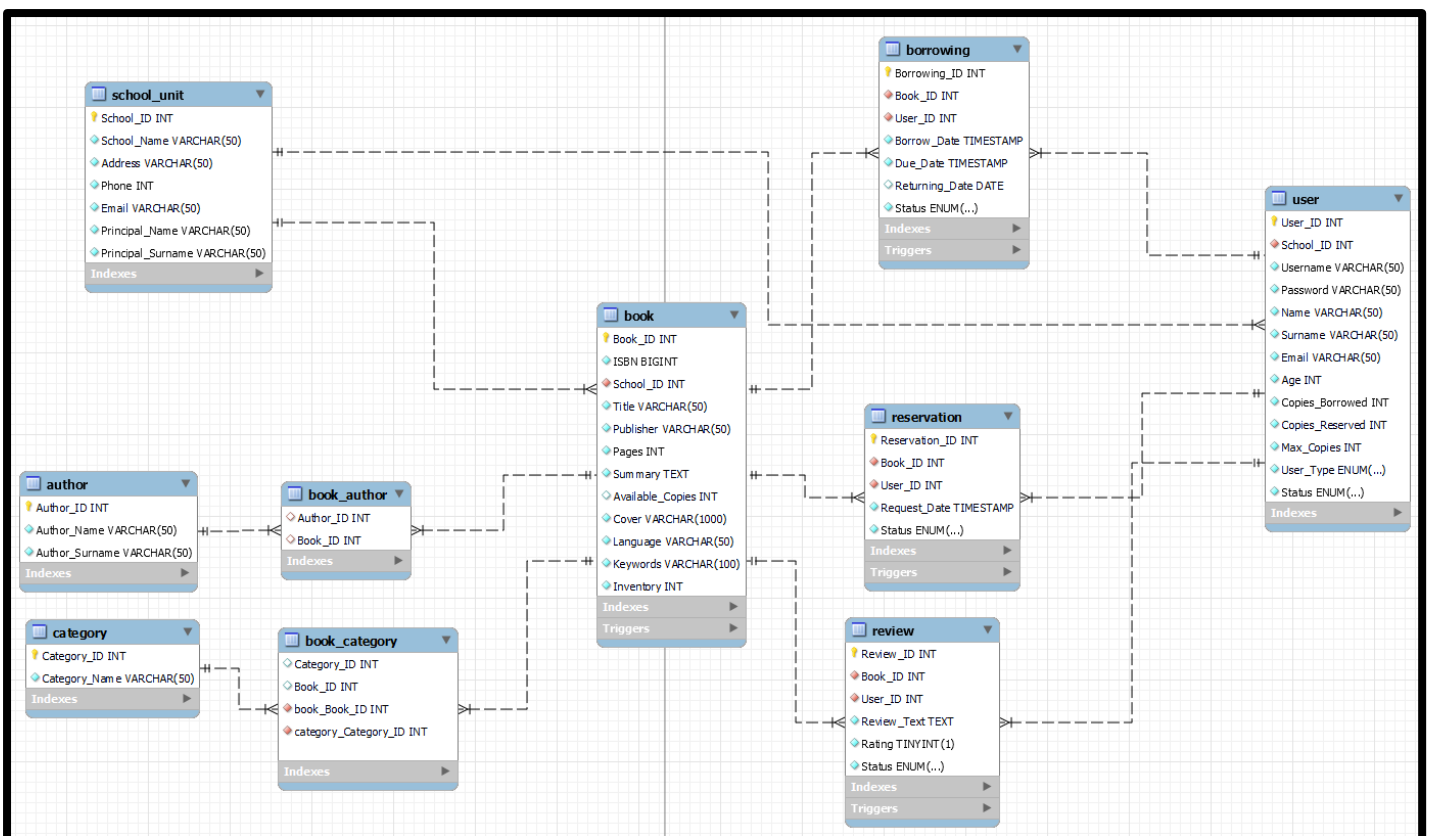
Αλέξανδρος Σκούρας, 03120105

Ιωάννης Τσαντήλας, 03120883

## Ενότητα 1: Διάγραμμα ER



## Ενότητα 2: Σχεσιακό Διάγραμμα



## **Ενότητα 3: Σύνοψη Περιγραφή Οντοτήτων και Σχέσεων**

### **3.1 Οντότητες**

Η βάση δεδομένων μας αποτελείται συνολικά από 10 οντότητες: 6 «βασικές» οντότητες, 2 «βοηθητικές» και 2 «συνδετικές».

Οι «βασικές» οντότητες είναι οι: School\_Unit, User, Book, Borrowing, Reservation, Review που ουσιαστικά είναι το σώμα της βάσης, οι «βοηθητικές» είναι οι: Author, Category που μας επιτρέπουν να εξηγήσουμε καλύτερα συγκεκριμένη ιδιότητα της οντότητας «Book» (κάθε βιβλίο μπορεί να ανήκει σε πολλές κατηγορίες και να έχει συγγραφεί από πολλούς συγγραφείς), ενώ οι «συνδετικές» είναι οι: Book\_Author, Book\_Category, που μας επιτρέπουν να αντιστοιχίσουμε το κάθε βιβλίο με πολλαπλές κατηγορίες και συγγραφείς (και αντίστροφα). Αναλυτικότερα για τις «βασικές» οντότητες:

#### ***School\_Unit***

Η σχολική μονάδα που αντιπροσωπεύει την αντίστοιχη βιβλιοθήκη. Περιγράφεται από ένα ID και τα βασικά στοιχεία της, δηλαδή, το όνομα, την διεύθυνση, το τηλέφωνο, το email και το ονοματεπώνυμο του διευθυντή της.

#### ***User***

Οι χρήστες που μπορούν να χρησιμοποιήσουν την βάση. Κάθε χρήστης χαρακτηρίζεται, εκτός από ένα ID, από τα βασικά στοιχεία του, δηλαδή username, password, ονοματεπώνυμο, email, ηλικία και το ID του σχολείου στο οποίο είναι εγγεγραμμένος.

Επιπλέον, ανάλογα το είδος του χρήστη (Administrator, Operator, Student ή Teacher), ο κάθε ένας έχει δικαίωμα για συγκεκριμένο πλήθος δανεισμών και κρατήσεων, τα οποία εξασφαλίζονται μέσω των attributes: copies\_borrowed, copies\_reserved, max\_copies (οι Administrator και Operators έχουν max\_copies = 0, οι Students έχουν max\_copies = 2 ενώ οι Teachers έχουν max\_copies = 1).

Τέλος, κάθε χρήστης, αφού κάνει αίτημα εγγραφής, έχει ένα attribute «status» που δηλώνει το αν έχει γίνει approved από τον αντίστοιχο ανώτερο του.

#### ***Book***

Κάθε βιβλίο, εκτός από το ID του, περιγράφεται από τα βασικά στοιχεία του, δηλαδή 13-digit ISBN αριθμό, ID του σχολείου στο οποίο ανήκει, τίτλος, εκδότης, πλήθος σελίδων, περίληψη, πλήθος διαθέσιμων αντιτύπων, εξώφυλλο, γλώσσα συγγραφής, λέξεις-κλειδιά, συνολικό απόθεμα.

#### ***Borrowing***

Η οντότητα «δανεισμός» χαρακτηρίζεται από το ID της, το ID του βιβλίου στο οποίο ασκείται ο δανεισμός, το ID του χρήστη που αιτείται την κράτηση, την ημερομηνία που έγινε το αίτημα, την ημερομηνία που πρέπει να επιστραφεί το βιβλίο, την (τελική)

ημερομηνία που επιστράφηκε το βιβλίο καθώς και ένα attribute «status», το οποίο δηλώνει το κατά πόσο έγινε αποδεκτό το αίτημα από τον αντίστοιχο operator.

### ***Reservation***

Η οντότητα «κράτηση» χαρακτηρίζεται από το ID της, το ID του βιβλίου στο οποίο ασκείται η κράτηση, το ID του χρήστη που αιτείται την κράτηση, την ημερομηνία που έγινε το αίτημα καθώς και ένα attribute «status», το οποίο δηλώνει το κατά πόσο έγινε αποδεκτό το αίτημα από τον αντίστοιχο operator.

### ***Review***

Η οντότητα «αξιολόγηση» χαρακτηρίζεται από το ID της, το ID του βιβλίου στο οποίο ασκείται η αξιολόγηση, το ID του χρήστη που αιτείται την αξιολόγηση, την αξιολόγηση και βαθμολογία, καθώς και ένα attribute «status», το οποίο δηλώνει το κατά πόσο έγινε αποδεκτό το αίτημα από τον αντίστοιχο operator.

## **3.2 Σχέσεις**

Οι σχέσεις που διέπουν τη βάση μας, όπως φαίνονται και στο ER-Diagram, είναι οι εξής:

- Ένα School\_Unit έχει Books.
- Οι Users μπορεί να είναι: Administrator, Operator, Student, Teacher.
- Ο Administrator προσθέτει νέα Σχολεία και εγκρίνει νέους Operators.
- Ο Operator εγκρίνει νέους Students και Teachers, ενώ προσθέτει και νέα Books στο School\_Unit που του αντιστοιχεί. Επίσης, εγκρίνει τα Borrowings, τα Reservations και τα Reviews.
- Κάθε Student και Teacher μπορεί να αιτηθεί Borrowing, Reservation και Review, αναλόγως των ορίων που έχουν θεσπιστεί.
- Κάθε «Action», δηλαδή Borrowing, Reservation και Review, επηρεάζει το αντίστοιχο βιβλίο.

## **Ενότητα 4: Indexing**

Αρχικά, είναι σημαντικό να αναφέρουμε πως στην MySQL τα indexes δημιουργούνται αυτόματα τόσο για τα primary keys όσο και για τα foreign keys που συνδέουν παρόμοια attributes μεταξύ των οντοτήτων. Για αυτό, έχουμε:

- Primary Keys: School\_Unit.School\_ID, Book.Book\_ID, User.User\_ID, Category.Category\_ID, Author.Author\_ID, Borrowing.Borrowing\_ID, Reservation.Reservation\_ID, Review.Review\_ID.
- Foreign Keys: Book. School\_ID, User. School\_ID, Book\_Author.Book\_ID, Book\_Category.Book\_ID, Borrowing.Book\_ID, Reservation.Book\_ID, Review.Book\_ID,

Borrowing.User\_ID, Reservation.User\_ID, Review.User\_ID, Book\_Author.Author\_ID, Book\_Category.Category\_ID.

Το indexing μας βοηθάει ώστε να έχουμε γρήγορη πρόσβαση σε αυτά τα στοιχεία σε queries και triggers που τα χρησιμοποιούν συχνά. Επιπλέον, προσθέσαμε μερικά indexes τύπου unique, όπου κρίναμε πως το αντίστοιχο attribute πρέπει να είναι μοναδικό και συγκριμένα:

- Unique Indexes: School.School\_Name, School.Addresss, School.Phone, School.Email, User.Username, User.Email, Category.Category\_Name.

### **Ενότητα 5: Constraints, Checks and Other**

Όπως προαναφέρθηκε, αναλόγως το είδος του User, ο κάθε ένας έχει συγκεκριμένο πλήθος αντίτυπων που μπορεί να δανειστεί και να αιτηθεί κράτηση. Για αυτό, στην δημιουργία ενός νέου User, το attribute «max\_copies» παίρνει τιμή 2 σε περίπτωση Μαθητή και 1 σε περίπτωση Καθηγητή (διαφορετικά είναι 0).

Ταυτόχρονα, είναι λογικό να ελέγχουμε πως τα copies\_borrowed, copies\_reserved και available\_copies να είναι πάντα μεγαλύτερα ή ίσα από το μηδέν, ενώ το inventory να είναι πάντα θετικό (δεν είναι λογικό να υπάρχει ένα βιβλίο με κανένα αντίτυπο στο απόθεμα μας). Σημαντικό είναι επίσης τα διαθέσιμα αντίτυπα να μην ξεπεράσουν το απόθεμα.

Τέλος, η βαθμολόγηση της αξιολόγησης θα πρέπει να είναι μεταξύ 1 και 5, ενώ όλοι οι νέοι Χρήστες, Δανεισμοί, Κρατήσεις και Αξιολογήσεις θα πρέπει να περιμένουν την έγκριση του Operator (και για τους νέους Operators, την έγκριση του Administrator), επομένως το Status τους είναι by default «On Hold».

## Ενότητα 6: DDL και DML

Το DDL ορίζει την δημιουργία των οντοτήτων (create, drop, alter) ενώ το DML επεξεργάζεται τα δεδομένα της βάσης (insert, update, delete, select). Ενδεικτικά, επισυνάπτουμε μέρη του DDL, της δημιουργίας των πινάκων «Book» και «User»:

```
CREATE TABLE IF NOT EXISTS `Book` (  
  `Book_ID` INT(50) NOT NULL AUTO_INCREMENT,  
  `ISBN` BIGINT(13) NOT NULL,  
  `School_ID` INT(50) NOT NULL,  
  `Title` VARCHAR(50) NOT NULL,  
  `Publisher` VARCHAR(50) NOT NULL,  
  `Pages` INT(50) NOT NULL,  
  `Summary` TEXT NOT NULL,  
  `Available_Copies` INT(50) CHECK (Available_Copies >= 0),  
  `Cover` VARCHAR(1000) NOT NULL DEFAULT 'https://hotemoji.com/images/dl/1/orange-book-emoji-by-twitter.png',  
  `Language` VARCHAR(50) NOT NULL,  
  `Keywords` VARCHAR(100) NOT NULL,  
  `Inventory` INT(50) NOT NULL CHECK (Inventory > 0),  
  PRIMARY KEY (`Book_ID`),  
  CONSTRAINT `chk_available_copies` CHECK (`Available_Copies` <= `Inventory`),  
  CONSTRAINT `fk_book_school_id` FOREIGN KEY (`School_ID`) REFERENCES `School_Unit` (`School_ID`) ON DELETE RESTRICT ON UPDATE CASCADE  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

```
CREATE TABLE IF NOT EXISTS `User` (  
  `User_ID` int(50) NOT NULL AUTO_INCREMENT,  
  `School_ID` int(50) NOT NULL,  
  `Username` varchar(50) NOT NULL UNIQUE,  
  `Password` varchar(50) NOT NULL,  
  `Name` varchar(50) NOT NULL,  
  `Surname` varchar(50) NOT NULL,  
  `Email` varchar(50) NOT NULL UNIQUE,  
  `Age` int(3) NOT NULL,  
  `Copies_Borrowed` int(50) NOT NULL DEFAULT 0,  
  `Copies_Reserved` int(50) NOT NULL DEFAULT 0,  
  `Max_Copies` INT(50) NOT NULL DEFAULT (CASE WHEN `User_Type` = 'Student' THEN 2  
                                              WHEN `User_Type` = 'Teacher' THEN 1  
                                              ELSE 0  
                                              END),  
  `User_Type` enum('Administrator', 'Operator', 'Teacher', 'Student') NOT NULL,  
  `Status` enum('Approved', 'On Hold') NOT NULL,  
  PRIMARY KEY (`User_ID`),  
  CHECK (`Copies_Borrowed` >= 0),  
  CHECK (`Copies_Reserved` >= 0),  
  CONSTRAINT `fk_user_id_school_unit` FOREIGN KEY (`School_ID`) REFERENCES `School_Unit` (`School_ID`) ON DELETE RESTRICT ON UPDATE CASCADE  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Καθώς και μερικά inserts:

```
INSERT INTO `School_Unit` (`School_ID`, `School_Name`, `Address`, `Phone`, `Email`, `Principal_Name`, `Principal_Surname`) VALUES  
(1, 'School A', 'Address A', 123456789, 'schoola1@school.com', 'John', 'Smith'),  
(2, 'School B', 'Address B', 234567891, 'schoolb2@school.com', 'Emily', 'Johnson'),  
(3, 'School C', 'Address C', 345678912, 'schoolc3@school.com', 'Daniel', 'Brown'),  
(4, 'School D', 'Address D', 456789123, 'schoold4@school.com', 'Michael', 'Davis'),  
(5, 'School E', 'Address E', 567891234, 'school5@school.com', 'Emma', 'Miller'),  
(6, 'School F', 'Address F', 678912345, 'schoolf6@school.com', 'Jose', 'Garcia'),  
(7, 'School G', 'Address G', 789456123, 'schoolg7@school.com', 'Jonathan', 'Venezuela'),  
(8, 'School H', 'Address H', 891234567, 'schoolh8@school.com', 'Mario', 'Ruiz'),  
(9, 'School I', 'Address I', 912345678, 'schooli9@school.com', 'Carlos', 'Esteban');
```

```

INSERT INTO `Category` (`Category_ID`, `Category_Name`)
VALUES
    (1, 'Fiction'),
    (2, 'Non-fiction'),
    (3, 'Mystery'),
    (4, 'Science fiction'),
    (5, 'Fantasy'),
    (6, 'Romance'),
    (7, 'Thriller'),
    (8, 'Historical fiction'),
    (9, 'Biography'),
    (10, 'Self-help'),
    (11, 'Science'),
    (12, 'Poetry'),
    (13, 'Travel'),
    (14, 'Horror'),
    (15, "Children's"),
    (16, 'Young adult'),
    (17, 'Humor'),
    (18, 'Philosophy'),
    (19, 'Art and photography'),
    (20, 'Business and finance');

```

## Ενότητα 7: Queries

```

1  -- -----
2  -- ----- ADMIN -----
3  -- -----
4  -- -----
5  -- ----- QUERY 1 - 3.1.1 ----- List with the total borrowings per school (Search criteria: year, calendar month).
6  -- -----
7  SELECT
8      s.School_Name,
9      YEAR(b.Borrow_Date) AS Borrow_Year,
10     MONTH(b.Borrow_Date) AS Borrow_Month,
11     COUNT(*) AS Total_Borrowings
12 FROM
13     School_Unit s
14     INNER JOIN User u ON s.School_ID = u.School_ID
15     INNER JOIN Borrowing b ON u.User_ID = b.User_ID
16 WHERE
17     YEAR(b.Borrow_Date) = <year> AND           -- Replace with the desired year
18     MONTH(b.Borrow_Date) = <month>           -- Replace with the desired calendar month
19 GROUP BY
20     s.School_ID,
21     YEAR(b.Borrow_Date),
22     MONTH(b.Borrow_Date)
23 ORDER BY
24     s.School_Name;

```

```

26  -- -----
27  -- ----- QUERY 2 - 3.1.2 ----- For a given book category (user-selected), which authors belong to it and which teachers have borrowed books from that category in the last year?
28  -- -----
29  SELECT DISTINCT a.Author_Name, a.Author_Surname
30  FROM Author a
31  JOIN Book_Author ba ON a.Author_ID = ba.Author_ID
32  JOIN Book_Category bc ON ba.Book_ID = bc.Book_ID
33  JOIN Category c ON bc.Category_ID = c.Category_ID
34  WHERE c.Category_Name = 'category_name';           -- Replace with the desired category_name
35
36  SELECT DISTINCT U.Name, U.Surname
37  FROM User U
38  JOIN Borrowing B ON U.User_ID = B.User_ID
39  JOIN Book_Category BC ON B.Book_ID = BC.Book_ID
40  JOIN Category C ON BC.Category_ID = C.Category_ID
41  WHERE U.User_Type = 'Teacher'
42  AND C.Category_Name = 'category_name'           -- Replace with the desired category_name
43  AND B.Borrow_Date >= DATE_SUB(CURRENT_DATE(), INTERVAL 1 YEAR);

```

```

45  -- -----
46  -- ----- QUERY 3 - 3.1.3 ----- Find young teachers (age < 40 years) who have borrowed the most books and the number of books.
47  -- -----
48  SELECT U.User_ID, U.Name, U.Surname, U.Age, COUNT(*) AS NumOfBooksBorrowed
49  FROM User U
50  JOIN Borrowing B ON U.User_ID = B.User_ID
51  JOIN Book BK ON B.Book_ID = BK.Book_ID
52  WHERE U.User_Type = 'Teacher' AND U.Age < 40
53  GROUP BY U.User_ID
54  ORDER BY NumOfBooksBorrowed DESC;
55
56  -- -----
57  -- ----- QUERY 4 - 3.1.4 ----- Find authors whose books have not been borrowed.
58  -- -----
59  SELECT a.Author_ID, a.Author_Name, a.Author_Surname
60  FROM Author a
61  JOIN Book_Author ba ON a.Author_ID = ba.Author_ID
62  LEFT JOIN Book b ON ba.Book_ID = b.Book_ID
63  LEFT JOIN Borrowing bor ON b.Book_ID = bor.Book_ID
64  GROUP BY a.Author_ID, a.Author_Name, a.Author_Surname
65  HAVING COUNT(DISTINCT bor.Borrowing_ID) = 0;

```

```

68  -- -----
69  -- ----- QUERY 5 - 3.1.5 ----- Which operators have loaned the same number of books in a year with more than 20 loans?
70  -- -----
71  SELECT sq.School_ID, sq.School_Name, u.Name AS Operator_Name, u.Surname AS Operator_Surname, sq.Total_Borrowings
72  FROM (
73      SELECT su.School_ID, su.School_Name, COUNT(*) AS Total_Borrowings
74      FROM School_Unit su
75      INNER JOIN User u ON su.School_ID = u.School_ID
76      INNER JOIN Borrowing b ON u.User_ID = b.User_ID
77      WHERE YEAR(b.Borrow_Date) = <user_selected_year>           -- Replace it!
78      GROUP BY su.School_ID
79      HAVING COUNT(*) >= 20
80  ) AS sq
81  INNER JOIN (
82      SELECT Total_Borrowings
83      FROM (
84          SELECT COUNT(*) AS Total_Borrowings
85          FROM School_Unit su
86          INNER JOIN User u ON su.School_ID = u.School_ID
87          INNER JOIN Borrowing b ON u.User_ID = b.User_ID
88          WHERE YEAR(b.Borrow_Date) = <user_selected_year>           -- Replace it!
89          GROUP BY su.School_ID
90          HAVING COUNT(*) >= 20
91      ) AS t
92      GROUP BY Total_Borrowings
93      HAVING COUNT(*) > 1
94  ) AS t2 ON sq.Total_Borrowings = t2.Total_Borrowings
95  INNER JOIN User u ON sq.School_ID = u.School_ID
96  WHERE u.User_Type = 'Operator'
97  ORDER BY sq.Total_Borrowings DESC;

```



```

99  -- -----
100 -- ----- QUERY 6 - 3.1.6 ----- Many books cover more than one category; among category pairs (e.g., history and poetry) that are common in books, find the top-3 pairs that appeared in borrowings.
101 -- -----
102 SELECT C1.Category_Name AS Category1, C2.Category_Name AS Category2, COUNT(*) AS BorrowingCount
103 FROM Book_Category BC1
104 JOIN Book_Category BC2 ON BC1.Book_ID = BC2.Book_ID AND BC1.Category_ID < BC2.Category_ID
105 JOIN Book B ON BC1.Book_ID = B.Book_ID
106 JOIN Borrowing BR ON B.Book_ID = BR.Book_ID
107 JOIN Category C1 ON BC1.Category_ID = C1.Category_ID
108 JOIN Category C2 ON BC2.Category_ID = C2.Category_ID
109 GROUP BY C1.Category_Name, C2.Category_Name
110 ORDER BY BorrowingCount DESC
111 LIMIT 3;
112
113 -- -----
114 -- ----- QUERY 7 - 3.1.7 ----- Find all authors who have written at least 5 books less than the author with the most books.
115 -- -----
116 SELECT A.Author_ID, A.Author_Name, A.Author_Surname, COUNT(*) AS Books_Written
117 FROM Author A
118 JOIN Book_Author BA ON A.Author_ID = BA.Author_ID
119 GROUP BY A.Author_ID, A.Author_Name, A.Author_Surname
120 HAVING Books_Written <= (
121     SELECT COUNT(*) - 5
122     FROM Book_Author
123     GROUP BY Author_ID
124     ORDER BY COUNT(*) DESC
125     LIMIT 1
126 )
127 ORDER BY Books_Written DESC;

```

```

129 -- -----
130 -- ----- OPERATOR -----
131 -- -----
132 -- -----
133 -- ----- QUERY 8 - 3.2.1 ----- Find all books by Title, Author (Search criteria: title/ category/ author/ copies).
134 -- -----
135 SELECT b.Book_ID, b.Title, GROUP_CONCAT(CONCAT(a.Author_Name, ' ', a.Author_Surname) SEPARATOR ', ') AS Authors, b.Pages, b.Available_Copies, b.Inventory
136 FROM Book b
137 JOIN Book_Author ba ON b.Book_ID = ba.Book_ID
138 JOIN Author a ON ba.Author_ID = a.Author_ID
139 WHERE b.Title = '<title>' -- Replace with the desired element
140 AND b.School_ID = <operator_school_id> -- Replace with the desired element
141 GROUP BY b.Book_ID;
142
143 SELECT b.Book_ID, b.Title, GROUP_CONCAT(CONCAT(a.Author_Name, ' ', a.Author_Surname) SEPARATOR ', ') AS Authors, b.Pages, b.Available_Copies, b.Inventory
144 FROM Book b
145 JOIN Book_Author ba ON b.Book_ID = ba.Book_ID
146 JOIN Author a ON ba.Author_ID = a.Author_ID
147 WHERE a.Author_Name = '<author_name>' -- Replace with the desired element
148 OR a.Author_Surname = '<author_surname>' -- Replace with the desired element
149 AND b.School_ID = <operator_school_id> -- Replace with the desired element
150 GROUP BY b.Book_ID;

```

```

152 SELECT b.Book_ID, b.Title, GROUP_CONCAT(CONCAT(a.Author_Name, ' ', a.Author_Surname) SEPARATOR ', ') AS Authors, b.Pages, b.Available_Copies, b.Inventory
153 FROM Book b
154 JOIN Book_Author ba ON b.Book_ID = ba.Book_ID
155 JOIN Author a ON ba.Author_ID = a.Author_ID
156 JOIN Book_Category bc ON b.Book_ID = bc.Book_ID
157 JOIN Category c ON bc.Category_ID = c.Category_ID
158 WHERE c.Category_Name = 'category_name' -- Replace with the desired element
159 AND b.School_ID = <operator_school_id> -- Replace with the desired element
160 GROUP BY b.Book_ID;
161
162 SELECT b.Book_ID, b.Title, GROUP_CONCAT(CONCAT(a.Author_Name, ' ', a.Author_Surname) SEPARATOR ', ') AS Authors, b.Pages, b.Available_Copies, b.Inventory
163 FROM Book b
164 JOIN Book_Author ba ON b.Book_ID = ba.Book_ID
165 JOIN Author a ON ba.Author_ID = a.Author_ID
166 WHERE b.Available_Copies = <available_copies> -- Replace with the desired element
167 AND b.School_ID = <operator_school_id> -- Replace with the desired element
168 GROUP BY b.Book_ID;

```

```

170  -- -----
171  -- ----- QUERY 9 - 3.2.2 ----- Find all borrowers who own at least one book and have delayed its return. (Search criteria: First Name, Last Name, Delay Days).
172  -- -----
173  SELECT u.User_ID, u.Name, u.Surname, DATEDIFF(CURDATE(), bor.Due_Date) AS Delay_Days
174  FROM User u
175  JOIN Borrowing bor ON u.User_ID = bor.User_ID
176  WHERE DATEDIFF(CURDATE(), bor.Due_Date) > 0
177        AND u.Name = '<name>' -- Replace with the desired element
178        AND u.Surname = '<surname>' -- Replace with the desired element
179        AND u.School_ID = <operator_school_id> -- Replace with the desired element
180        AND bor.Returning_Date IS NULL
181  GROUP BY u.User_ID, u.Name, u.Surname;
182
183  SELECT u.User_ID, u.Name, u.Surname, DATEDIFF(CURDATE(), bor.Due_Date) AS Delay_Days
184  FROM User u
185  JOIN Borrowing bor ON u.User_ID = bor.User_ID
186  WHERE DATEDIFF(CURDATE(), bor.Due_Date) > 0
187        AND DATEDIFF(CURDATE(), bor.Due_Date) > <delay_days> -- Replace with the desired element
188        AND u.School_ID = <operator_school_id> -- Replace with the desired element
189        AND bor.Returning_Date IS NULL
190  GROUP BY u.User_ID, u.Name, u.Surname;

```

```

192  -- -----
193  -- ----- QUERY 10 - 3.2.3 ----- Average Ratings per borrower and category (Search criteria: user/category)
194  -- -----
195  SELECT u.User_ID, u.Name, u.Surname, AVG(r.Rating) AS Average_Rating
196  FROM User u
197  JOIN Review r ON u.User_ID = r.User_ID
198  WHERE (u.User_Type = 'Student' OR u.User_Type = 'Teacher')
199        AND u.username = '<username>' -- Replace with the desired element
200        AND u.School_ID = <operator_school_id> -- Replace with the desired element
201  GROUP BY u.User_ID, u.Name, u.Surname;
202
203  SELECT u.User_ID, u.Name, u.Surname, AVG(r.Rating) AS Average_Rating
204  FROM User u
205  JOIN Review r ON u.User_ID = r.User_ID
206  JOIN Book b ON r.Book_ID = b.Book_ID
207  JOIN Book_Category bc ON b.Book_ID = bc.Book_ID
208  JOIN Category c ON bc.Category_ID = c.Category_ID
209  WHERE (u.User_Type = 'Student' OR u.User_Type = 'Teacher')
210        AND c.Category_Name = '<category_name>' -- Replace with the desired element
211        AND b.School_ID = <operator_school_id> -- Replace with the desired element
212  GROUP BY u.User_ID, u.Name, u.Surname;

```

```

214  -- -----
215  -- ----- USER -----
216  -- -----
217  -- -----
218  -- ----- QUERY 11 - 3.3.1 ----- List with all books (Search criteria: title/category/author), ability to select a book and create a reservation request.
219  -- -----
220  SELECT b.Book_ID, b.Title, GROUP_CONCAT(a.Author_Name, ' ', a.Author_Surname) AS Authors, b.Pages, b.Available_Copies, b.Inventory
221  FROM Book b
222  JOIN Book_Author ba ON b.Book_ID = ba.Book_ID
223  JOIN Author a ON ba.Author_ID = a.Author_ID
224  WHERE b.Title = '<title>' -- Replace with the desired element
225  GROUP BY b.Book_ID, b.Title, b.Pages, b.Available_Copies, b.Inventory;
226
227  SELECT b.Book_ID, b.Title, GROUP_CONCAT(a.Author_Name, ' ', a.Author_Surname) AS Authors, b.Pages, b.Available_Copies, b.Inventory
228  FROM Book b
229  JOIN Book_Author ba ON b.Book_ID = ba.Book_ID
230  JOIN Author a ON ba.Author_ID = a.Author_ID
231  JOIN Book_Category bc ON b.Book_ID = bc.Book_ID
232  JOIN Category c ON bc.Category_ID = c.Category_ID
233  WHERE c.Category_Name = '<category_name>' -- Replace with the desired element
234  GROUP BY b.Book_ID, b.Title, b.Pages, b.Available_Copies, b.Inventory;
235
236  SELECT b.Book_ID, b.Title, GROUP_CONCAT(a.Author_Name, ' ', a.Author_Surname) AS Authors, b.Pages, b.Available_Copies, b.Inventory
237  FROM Book b
238  JOIN Book_Author ba ON b.Book_ID = ba.Book_ID
239  JOIN Author a ON ba.Author_ID = a.Author_ID
240  WHERE a.Author_Name = '<author_name>' -- Replace with the desired element
241        AND a.Author_Surname = '<author_surname>' -- Replace with the desired element
242  GROUP BY b.Book_ID, b.Title, b.Pages, b.Available_Copies, b.Inventory;

```

```
244      -- -----
245      -- ----- QUERY 12 - 3.3.2 ----- List of all books borrowed by this user.
246      -- -----
247      SELECT b.Book_ID, b.Title, GROUP_CONCAT(a.Author_Name, ' ', a.Author_Surname) AS Authors
248      FROM Borrowing bor
249      JOIN Book b ON bor.Book_ID = b.Book_ID
250      JOIN Book_Author ba ON b.Book_ID = ba.Book_ID
251      JOIN Author a ON ba.Author_ID = a.Author_ID
252      WHERE bor.User_ID = '<logged_in_user_id>' -- Replace with the desired element
253      GROUP BY b.Book_ID, b.Title;
```

## Ενότητα 8: User Interface

*User Login and Sign Up:*

### Login

Login

Not Registered?

[Sign Up Here](#)

### Sign up

User Type

School Unit

First Name

Last Name

Email

Age

Username

Password

Submit

# Administrator

## Home Interface:

Library - Home Page

Logout Home

View Books

Show all Books in all Libraries

Show

View School Units

Show all registered School Units

Show

New Operators

Accept New Operators

Show

Borrowing per School\_Unit

Show Borrowings Per School Unit (Query 3.1.1)

Show

Authors Per Category

Show Authors Per Category (Query 3.1.2a)

Show

Teachers Per Category

Show list of Teachers who have borrowed books from a specific Category (Query 3.1.2b)

Show

Young Teachers

Show the young teachers (age<40) who have borrowed the most books and the number of books borrowed (Query 3.1.3)

Show

Lonely Authors

Show Authors whose books have not been borrowed (Query 3.1.4)

Show

Same Loans

Show the operators that have loaned the same number of books in a year, with more than 20 loans (Query 3.1.5)

Show

Best Pairs

Among category pairs, shot the top-3 pairs in borrowings (Query 3.1.6)

Show

5 Books Less Than Max Author

Show all authors who have written at least 5 books less than the author with the most books (Query 3.1.7)

Show

## Show Schools:

Library - Schools							Logout	Home
							Add New School	
School ID	School Name	Address	Phone	Email	Principal Name	Principal Surname		
1	School A	Address A	123456789	schoola1@school.com	John	Smith		
2	School B	Address B	234567891	schoolb2@school.com	Emily	Johnson		
3	School C	Address C	345678912	schoolc3@school.com	Daniel	Brown		
4	School D	Address D	456789123	schoold4@school.com	Michael	Davis		
5	School E	Address E	567891234	school5@school.com	Emma	Miller		
6	School F	Address F	678912345	schoolf6@school.com	Jose	Garcia		
7	School G	Address G	789456123	schoolg7@school.com	Jonathan	Venezuela		
8	School H	Address H	891234567	schoolh8@school.com	Mario	Ruiz		
9	School I	Address I	912345678	schooli9@school.com	Carlos	Esteban		

## Add School:

### Add School

School Name

Address

Phone

Email

Principal Name

Principal Surname

Submit

## New Operators:

Library - New Operators							Logout	Home
User ID	Name	Surname	Email	Age				
452	name452	surname452	name452surname452@gmail.com	38	Accept		Reject	
453	name453	surname453	name453surname453@gmail.com	28	Accept		Reject	
454	name454	surname454	name454surname454@gmail.com	32	Accept		Reject	
455	name455	surname455	name455surname455@gmail.com	31	Accept		Reject	
456	name456	surname456	name456surname456@gmail.com	32	Accept		Reject	

## Query Example: 3.1.4 – Lonely Authors

Library - Lonely Authors			Logout	Home
Author ID	Name	Surname		
61	Lilith	Brown		
62	Logan	Black		
63	Leo	Valdez		
64	Lucas	Jackson		
65	Lola	Chase		

## Query Example: 3.1.6 – Best Pairs

Library - Best Pairs			Logout	Home
Category 1	Category 2	Total Borrowings		
Fantasy	Business and finance	18		
Historical fiction	Biography	13		
Biography	Poetry	11		

## Operator

Home Interface:

Library - Home Page

Logout

Home

View Books

Show all books in my Library  
(Query 3.2.1)

Show

View Borrowings

Show borrowings in my School

Show

View Reservations

Show Reservations in my School

Show

Delayed Borrowings

Show late borrowed books  
(Query 3.2.2)

Show

Average Rating

Average ratings per borrower  
and category (Query 3.2.3)

Show

New Users

Approve New Users

Show

Approve Borrowings

Approve New Borrowings

Show

Approve Reservations

Approve New Reservations

Show

Approve Reviews

Approve New Reviews

Show

## New Users for Operator:

Library - New Users						Logout	Home
User ID	Name	Surname	Email	Age	User Type		
461	name461	surname461	name461surname461@gmail.com	25	Student	Accept	Reject
467	name467	surname467	name467surname467@gmail.com	19	Student	Accept	Reject
472	name472	surname472	name472surname472@gmail.com	22	Student	Accept	Reject
478	name478	surname478	name478surname478@gmail.com	20	Student	Accept	Reject
489	name489	surname489	name489surname489@gmail.com	25	Student	Accept	Reject
492	name492	surname492	name492surname492@gmail.com	21	Student	Accept	Reject
493	name493	surname493	name493surname493@gmail.com	18	Student	Accept	Reject
496	name496	surname496	name496surname496@gmail.com	18	Student	Accept	Reject
498	name498	surname498	name498surname498@gmail.com	25	Student	Accept	Reject
508	name508	surname508	name508surname508@gmail.com	22	Student	Accept	Reject
512	name512	surname512	name512surname512@gmail.com	25	Student	Accept	Reject
532	name532	surname532	name532surname532@gmail.com	26	Student	Accept	Reject
537	name537	surname537	name537surname537@gmail.com	47	Teacher	Accept	Reject
558	name558	surname558	name558surname558@gmail.com	54	Teacher	Accept	Reject

## Query Example: 3.2.2 – Delayed Borrowings

Library - Late Borrowed					Logout	Home
User ID	Username	Name	Surname	Delay Days		
103	user103	name103	surname103	8		
111	user111	name111	surname111	8		
120	user120	name120	surname120	8		
124	user124	name124	surname124	8		
135	user135	name135	surname135	8		

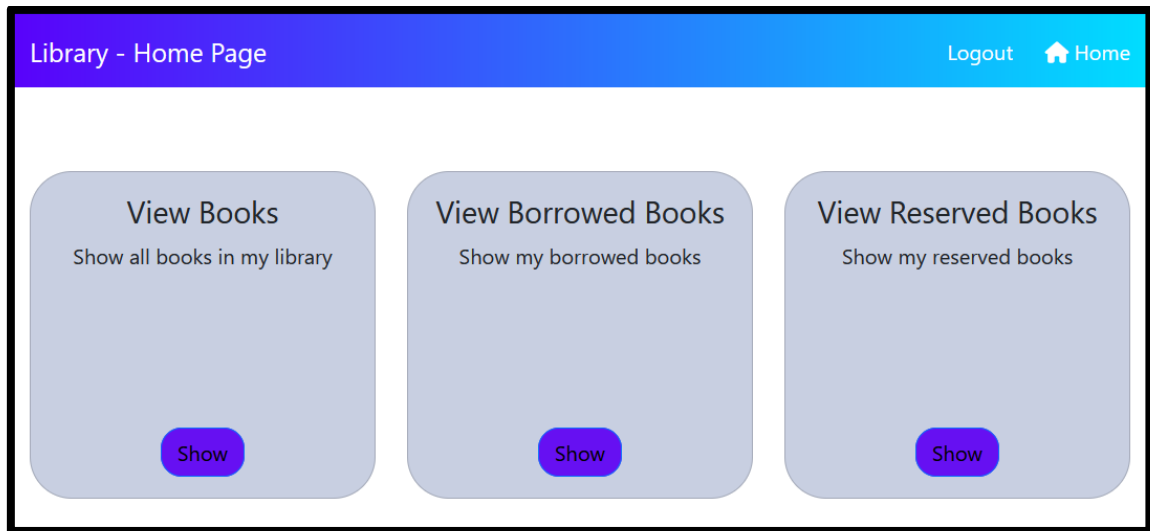
## Approve Reservations:

Library - New Reservations						Logout	Home
Reservation ID	Title	Username	Name	Surname	Request Date		
41	The Communist Dream	user617	name617	surname617	2023-06-05	Accept	Reject
42	The Communist Dream	user618	name618	surname618	2023-06-05	Accept	Reject

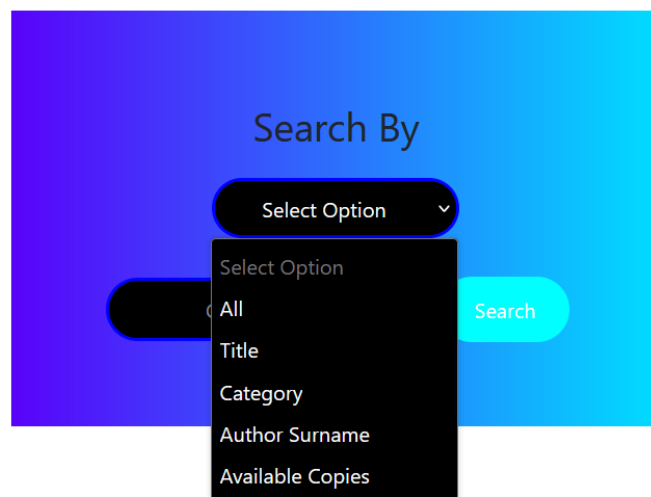
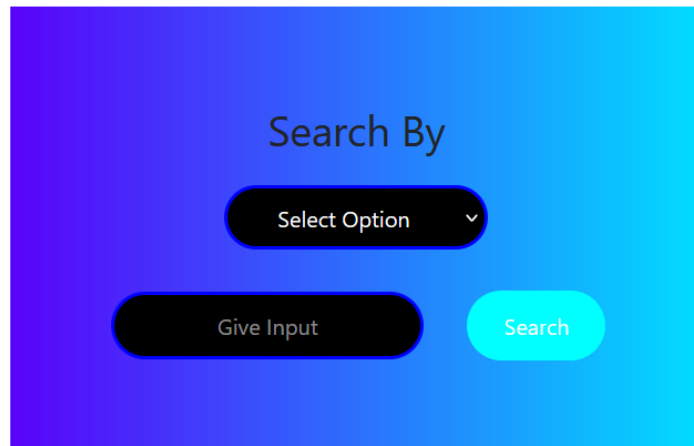


# Student

*Home Interface:*



*View Books in My Library:*



Book ID	Title	Authors	Pages	Available Copies	Inventory			
2	Alice's Adventures in Wonderland	J.K. Rowling	451	13	13	Borrow	Reserve	Review
25	Dracula	C.S. Lewis, Aldous Huxley	975	14	14	Borrow	Reserve	Review
29	To Kill a Mockingbird	H.G. Wells, George R.R. Martin, H.P. Lovecraft	751	0	18	Borrow	Reserve	Review
42	Sense and Sensibility	Terry Pratchett, Oscar Wilde	395	10	10	Borrow	Reserve	Review
44	Tess of the d'Urbervilles	Rudyard Kipling	155	12	12	Borrow	Reserve	Review
58	Black Beauty	Donna Tartt	390	13	13	Borrow	Reserve	Review
61	The House of the Seven Gables	William Shakespeare	874	12	12	Borrow	Reserve	Review
62	Lord of the Flies	J.K. Rowling	184	13	13	Borrow	Reserve	Review
68	The Stranger	Mark Twain	356	13	13	Borrow	Reserve	Review
71	Kidnapped: The Adventures of David Balfour	Fyodor Dostoevsky	622	10	10	Borrow	Reserve	Review
81	A Connecticut Yankee in King Arthur's Court	Oscar Wilde	886	16	16	Borrow	Reserve	Review
82	White Fang	William Faulkner, Agatha Christie	688	14	14	Borrow	Reserve	Review
83	Fathers and Sons	Franz Kafka	272	12	12	Borrow	Reserve	Review
89	Persuasion	H.G. Wells, Emily Bronte	236	14	14	Borrow	Reserve	Review

View My Borrowings:

Search Past or Active Borrowings

HistoryActive


Borrowing ID	Book ID	Book Title	Username	Borrower Name	Borrower Surname	Borrower Type	Borrow Date	Due Date	Returning Date
138	16	Little Women	user150	name150	surname150	Student	2022-08-03	2022-08-10	2022-08-04
156	40	The Picture of Dorian Gray	user150	name150	surname150	Student	2022-08-19	2022-08-26	2022-08-26
382	300	Don Quixote Vol.2	user150	name150	surname150	Student	2023-05-20	2023-05-27	None

Borrowing ID	Book ID	Book Title	Username	Borrower Name	Borrower Surname	Borrower Type	Borrow Date	Due Date	Returning Date	Status
382	300	Don Quixote Vol.2	user150	name150	surname150	Student	2023-05-20	2023-05-27	None	Approved

# Teacher

*Home Interface:*

Library - Home Page

Logout  Home

View Books

Show all books in my Library

Show

View Borrowed Books

Show my borrowed books

Show

View Reserved Books

Show my reserved books

Show

Profile

Update my Profile

Show

*Updating Profile from Teacher:*

Update Profile

name351

surname351

name351surname351@gmail

61

user351

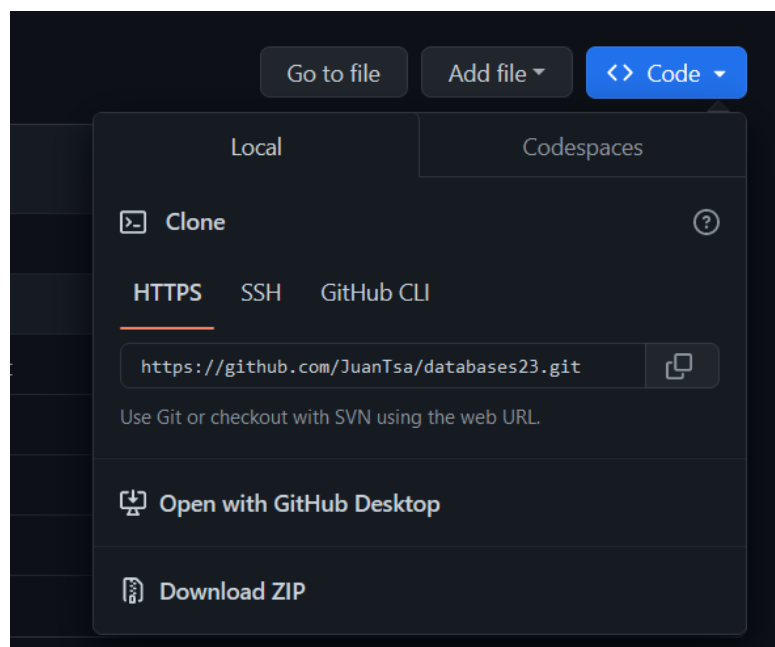
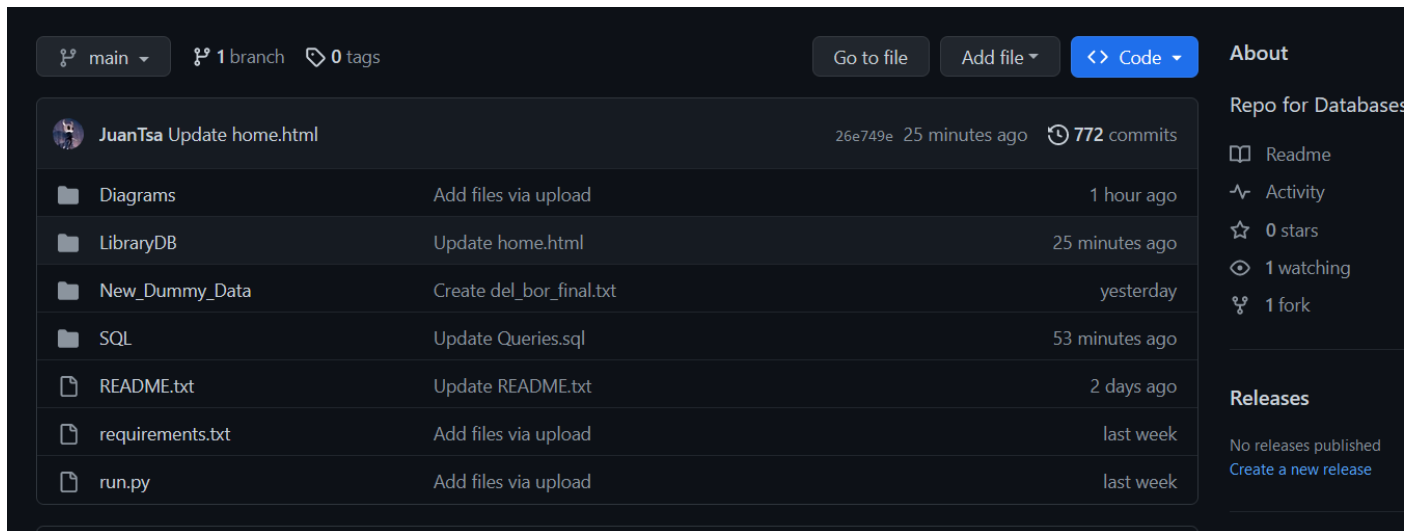
●●●●●●

Submit

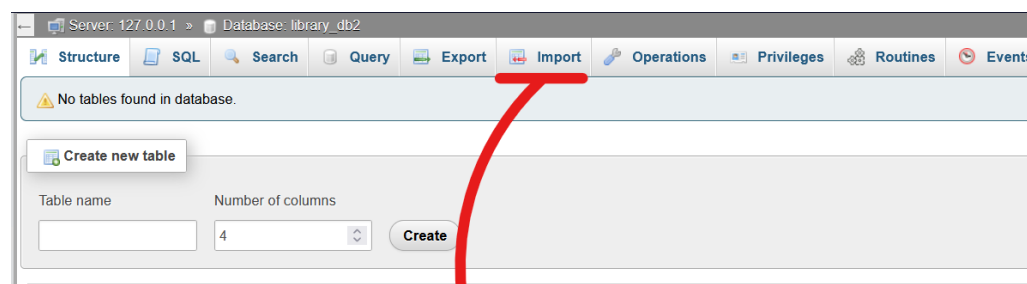
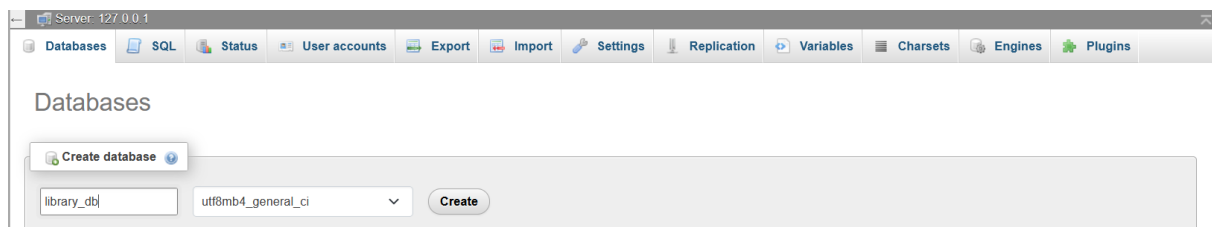
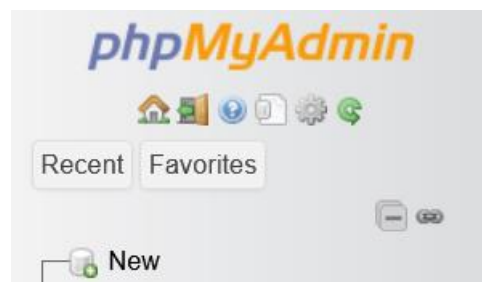
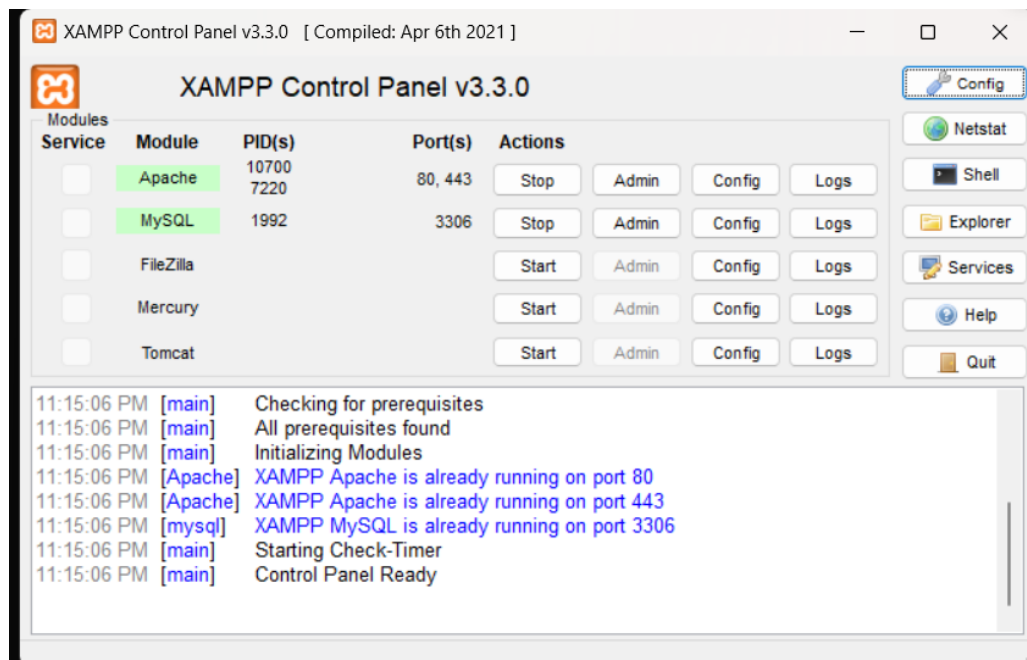
## Ενότητα 9: Installation Guide

Αρχικά, ακολουθείστε το παρακάτω link και κατεβάστε το git repository πατώντας στο «Code» και μετά «Download ZIP»:

<https://github.com/JuanTsa/databases23>



Κάντε extract all σε έναν φάκελο της επιλογής σας (ενδεικτικά με όνομα «app»). Χρησιμοποιώντας το XAMPP (Apache και MySQL activated), συνδεθείτε στο phpMyAdmin και δημιουργήστε μία νέα βάση με όνομα «library\_db» (είναι πολύ σημαντικό να προσέξετε την ορθογραφία να είναι ακριβής).



Ανοίξτε το terminal σας, πλοηγηθείτε στο directory που αποθηκεύσατε τον φάκελο app και εκτελέστε την εντολή: **pip install -r requirements.txt**

Όντας εντός της βάσης, πατήστε Import > Browse > app > DDL&DML.sql για να δημιουργήσετε την βάση.

File to import:

File may be compressed (gzip, bzip2) or uncompressed.  
A compressed file's name must end in **[format].[compression]**. Example: **.sql.zip**

Browse your computer: (Max: 40MiB)

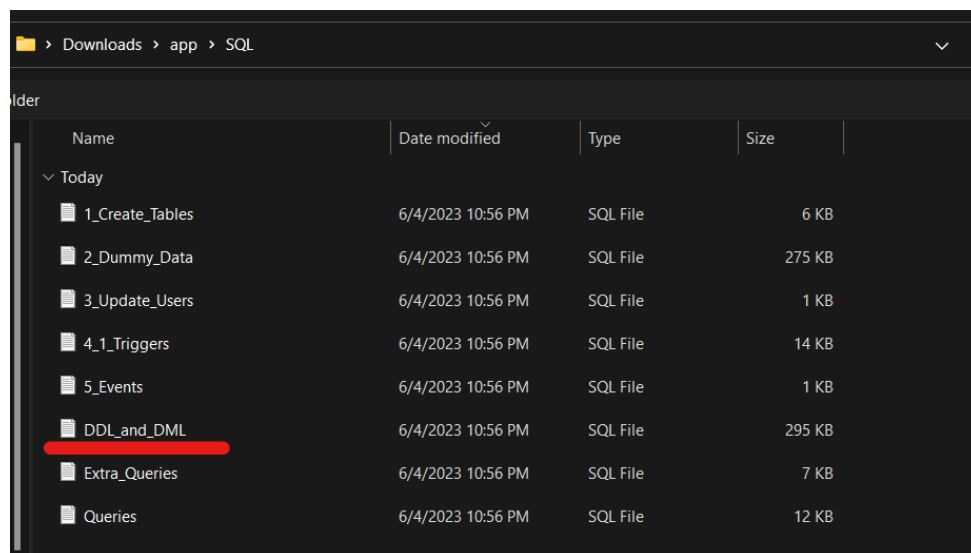
Browse...

No file selected.

You may also drag and drop a file on any page.

Character set of the file:

utf-8



Name	Date modified	Type	Size
▼ Today			
1_Create_Tables	6/4/2023 10:56 PM	SQL File	6 KB
2_Dummy_Data	6/4/2023 10:56 PM	SQL File	275 KB
3_Update_Users	6/4/2023 10:56 PM	SQL File	1 KB
4_1_Triggers	6/4/2023 10:56 PM	SQL File	14 KB
5_Events	6/4/2023 10:56 PM	SQL File	1 KB
DDL_and_DML	6/4/2023 10:56 PM	SQL File	295 KB
Extra_Queries	6/4/2023 10:56 PM	SQL File	7 KB
Queries	6/4/2023 10:56 PM	SQL File	12 KB

Στην συνέχεια, μέσω του φακέλου `app`, τρέξτε το αρχείο `run.py`, όπου θα σας εμφανιστεί το εξής παράθυρο με οδηγίες:

```
C:\Program Files\WindowsAp x + -
* Serving Flask app 'LibraryDB'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://localhost:3000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 385-305-545
```

Ακολουθήστε το link που παρουσιάζεται (με copy-paste σε κάποιο browser):

<http://localhost:3000>