# Instructions to Run

Install Python and Pip (Pip comes preinstalled with python, as does Sqlite3)

https://www.python.org/downloads/release/python-3121/

1. pip install pillow
2. py main.py

Common Error: If it says you already have pip installed but then says you do not have it installed when trying to run the main.py file, you probably have two python installations and you need to remove the oldest one.

# UST Book Fetch User Interface

Our UI contains a student, customer service, and an admin module. The student module gives the user the ability to perform actions relating to the student. The user can create a new student, create a cart, create a new order based on a cart, and create a new rating. The student module also lets the user update their cart and cancel an order. The customer service module allows the user to create trouble tickets and update them by assigning them to an admin. The Administrator module has two subsections, administrator, and super administrator. The administrator can create a new book and a new university. The super administrator can create a new customer service employee as well.

Libraries used: Pillow, sqlite3, tkinter

New indicies added: Assuming this is data added that was not provided:

Courses:

- Added 6 courses
- (2, 1, "CISC 450", 2023, 2),
- (2, 2, "STAT 450", 2023, 2),
- (2, 3, "ENGL 450", 2023, 1),
- (2, 4, "DIMA 450", 2023, 1),
- (1, 5, "CISC 450", 2023, 1),
- (3, 6, "STAT 450", 2023, 2);

Departments:

- Added 6 departments
- (1, 2, "CS", 1),
- (2, 2, "EN", 2),

- (3, 2, "ST", 3),
- (4, 2, "DI", 4),
- (5, 1, "CS", 5),
- (6, 3, "ST", 6);

Instructors:

- Added 6 instructors
- (1, 1, 2, "Sawin", null),
- (2, 2, 2, "James", null),
- (3, 3, 2, "John", null),
- (4, 4, 2, "Sue", null),
- (5, 5, 1, "Louise", null),
- (6, 6, 3, "Pumpernickle", null);

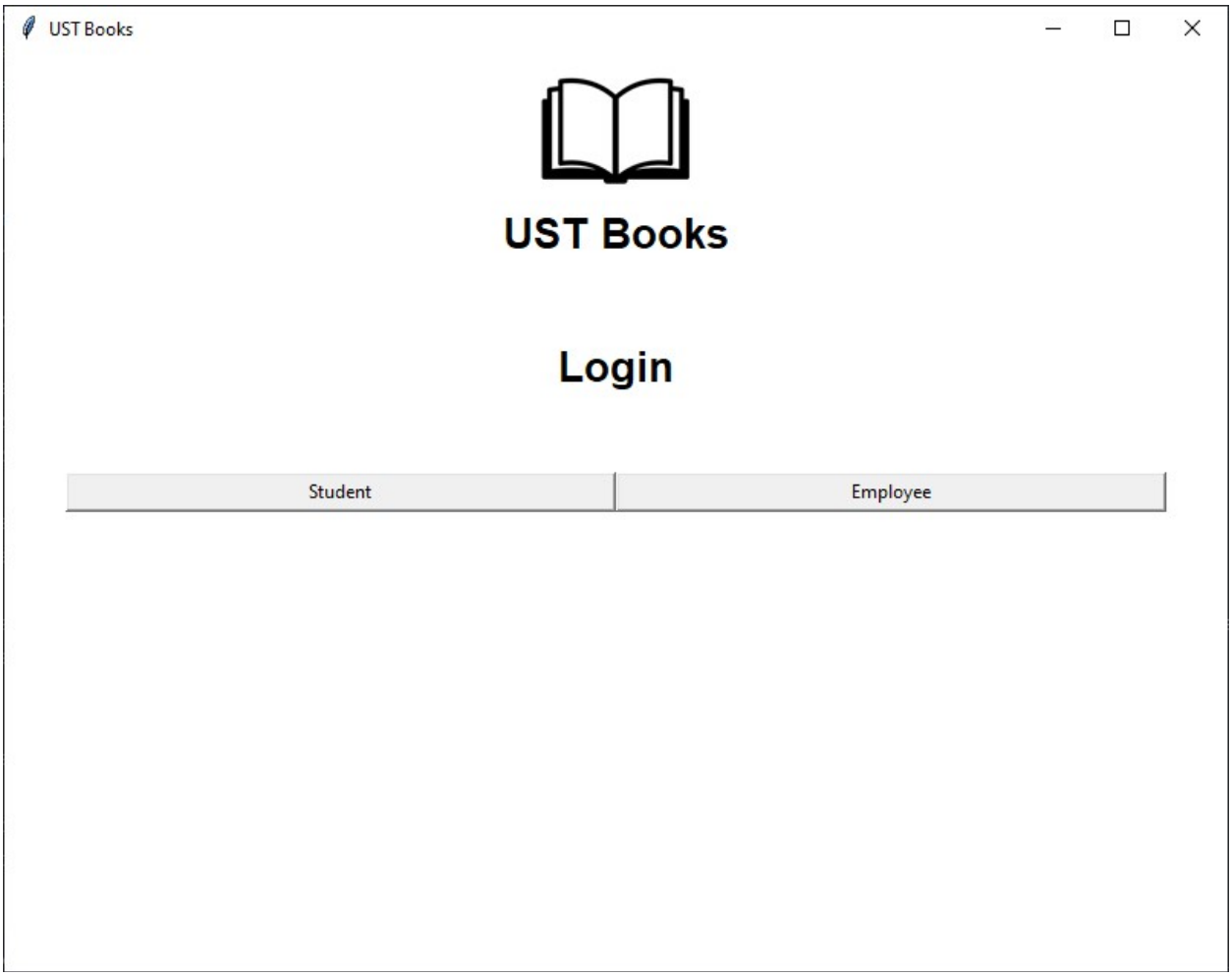
Categories:

- Added 4 categories
- ("Computer Science"),
- ("English"),
- ("Statistics"),
- ("Digital Media");

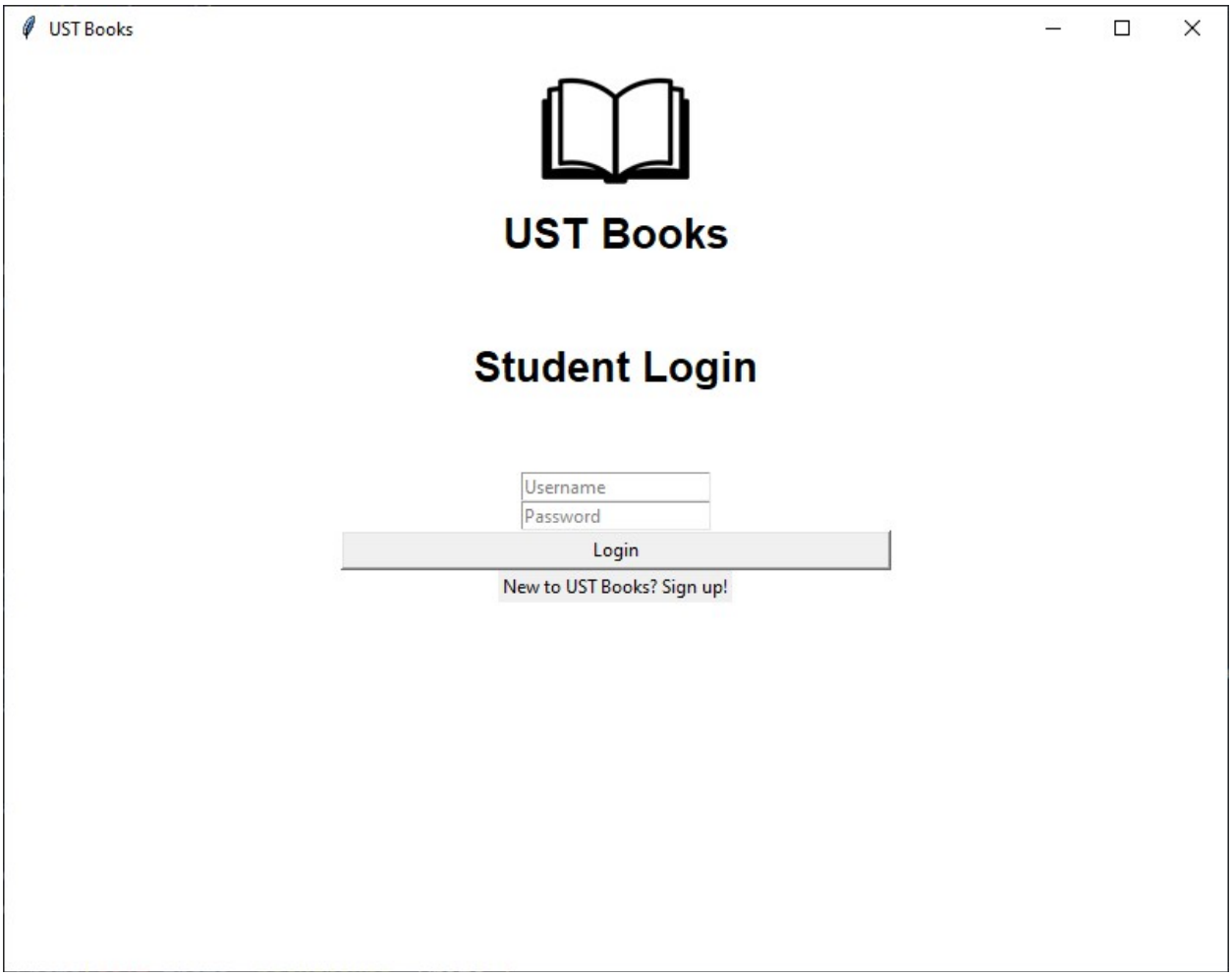
Recommendations:

- Added 3 recommendations
- (1, 1),
- (2, 2),
- (4, 11);

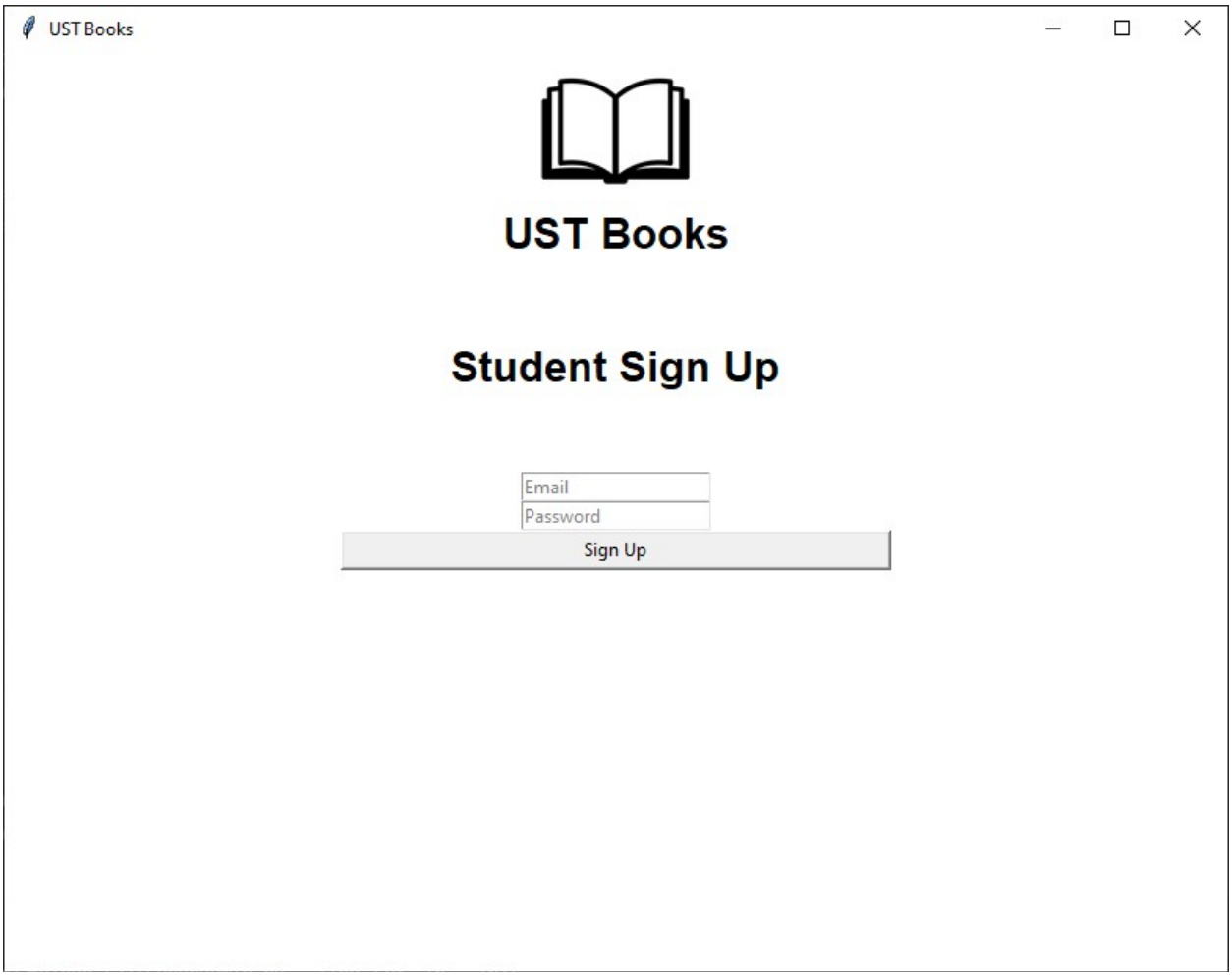
Changed one review to 5 stars, and added completed dates to tickets so certain queries would produce results.



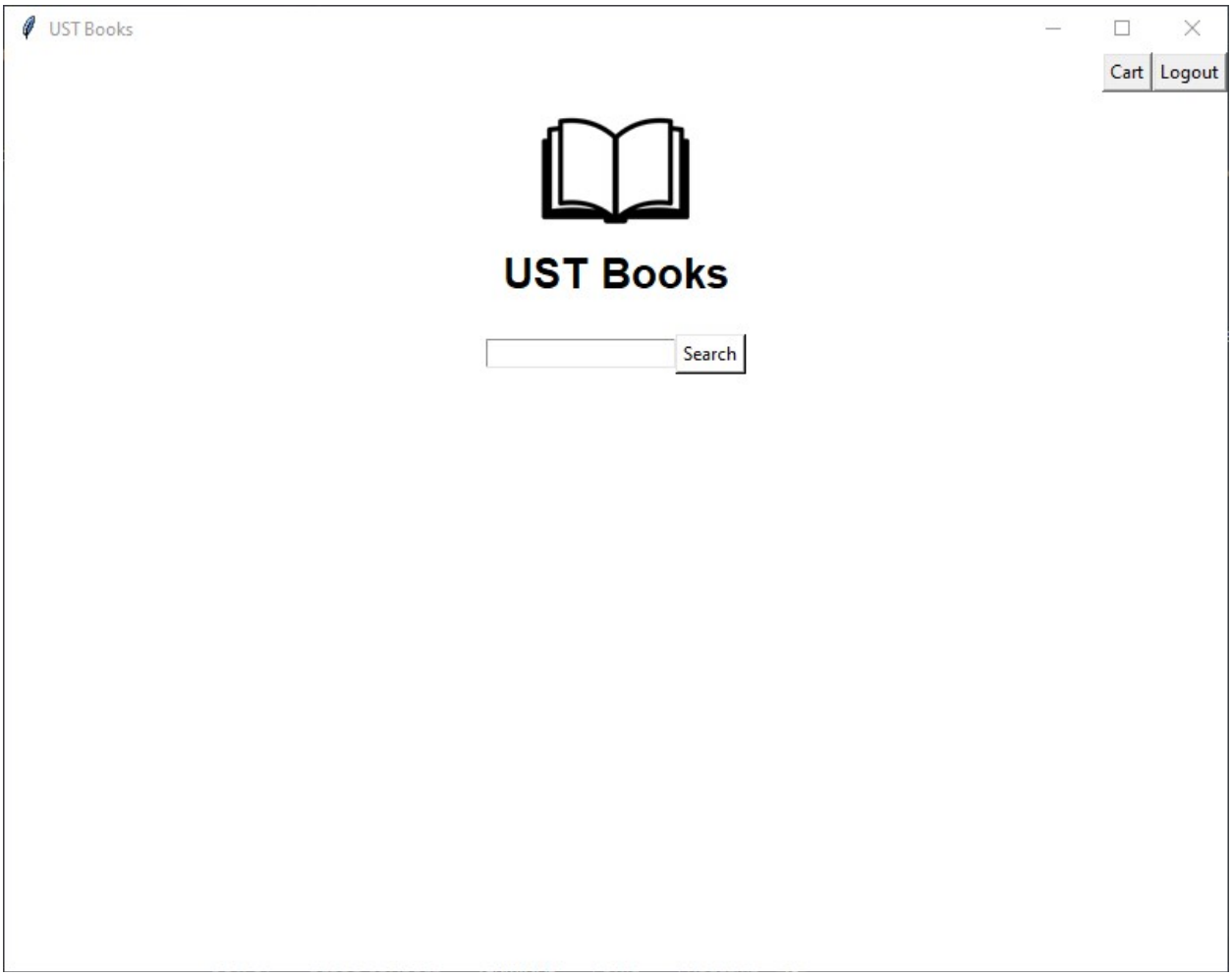

This page gives the user the option to login as a student or an employee. Clicking on either of these buttons will navigate the page to the login page of the button that was clicked.



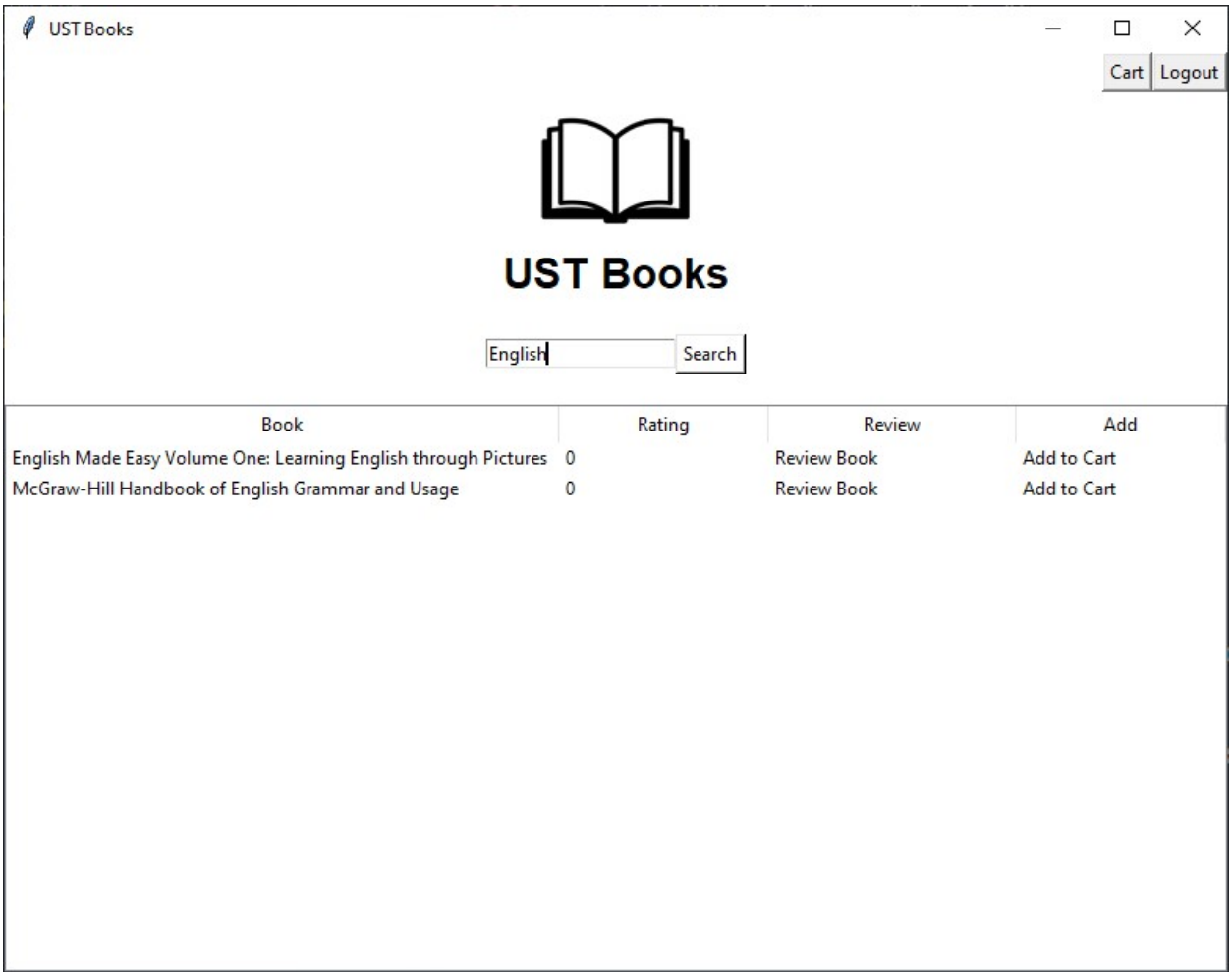

This page is the sign in page for student users. This page has two text entries for a username and a password. Below the text entries are two buttons. One is a submit button for the login form, labeled “Login”. The other leads to a sign up page for students when clicked.



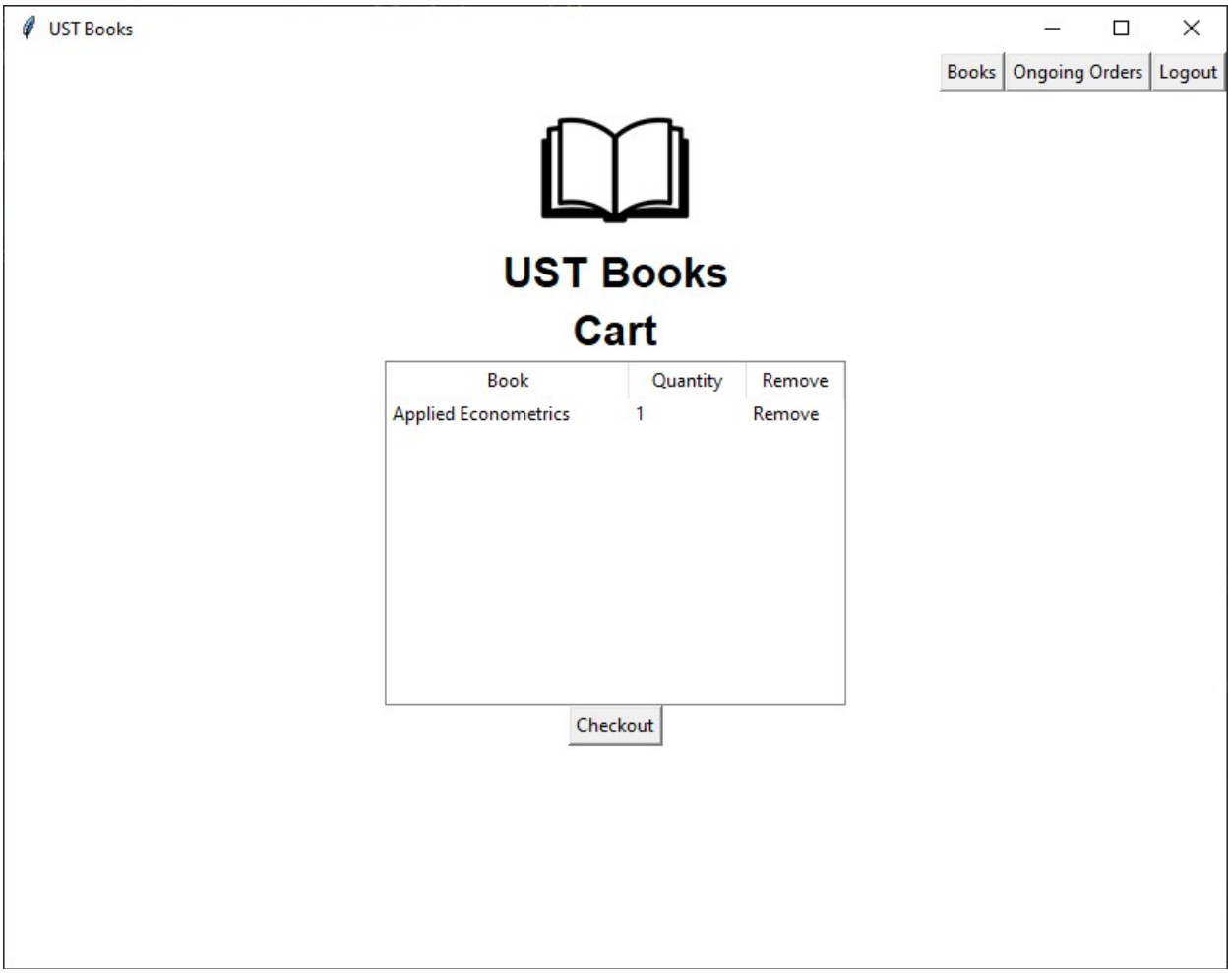

After the sign up button is clicked, the page navigates to the sign up page for students. This page has two text entries, one for email and one for password. Below the text entries is a sign up button that will create a new student user and log in.



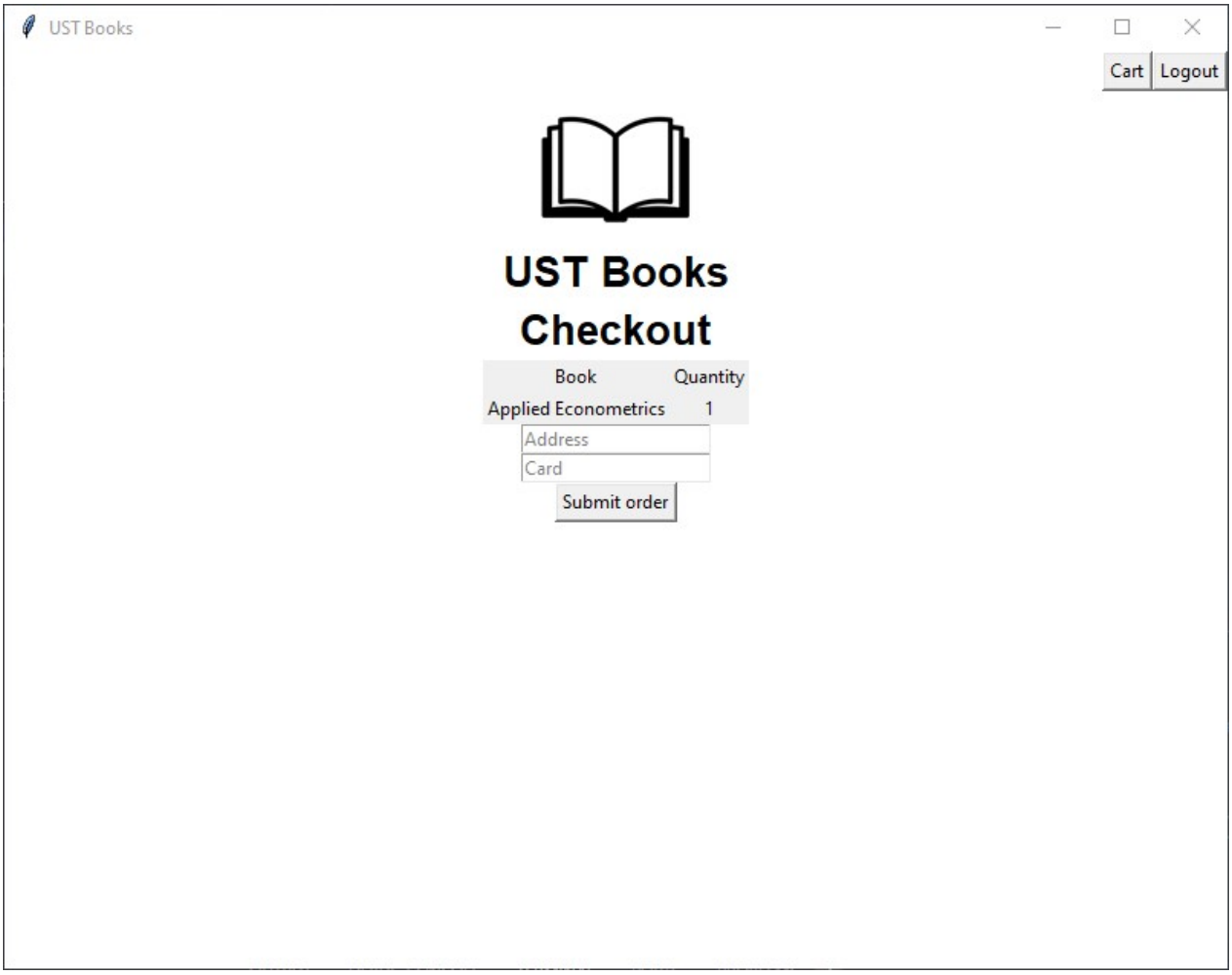

After a student logs in, the page navigates to the search page. This page has a text entrie and a search buttons that executes the search. This page allows students to search for books in the database.



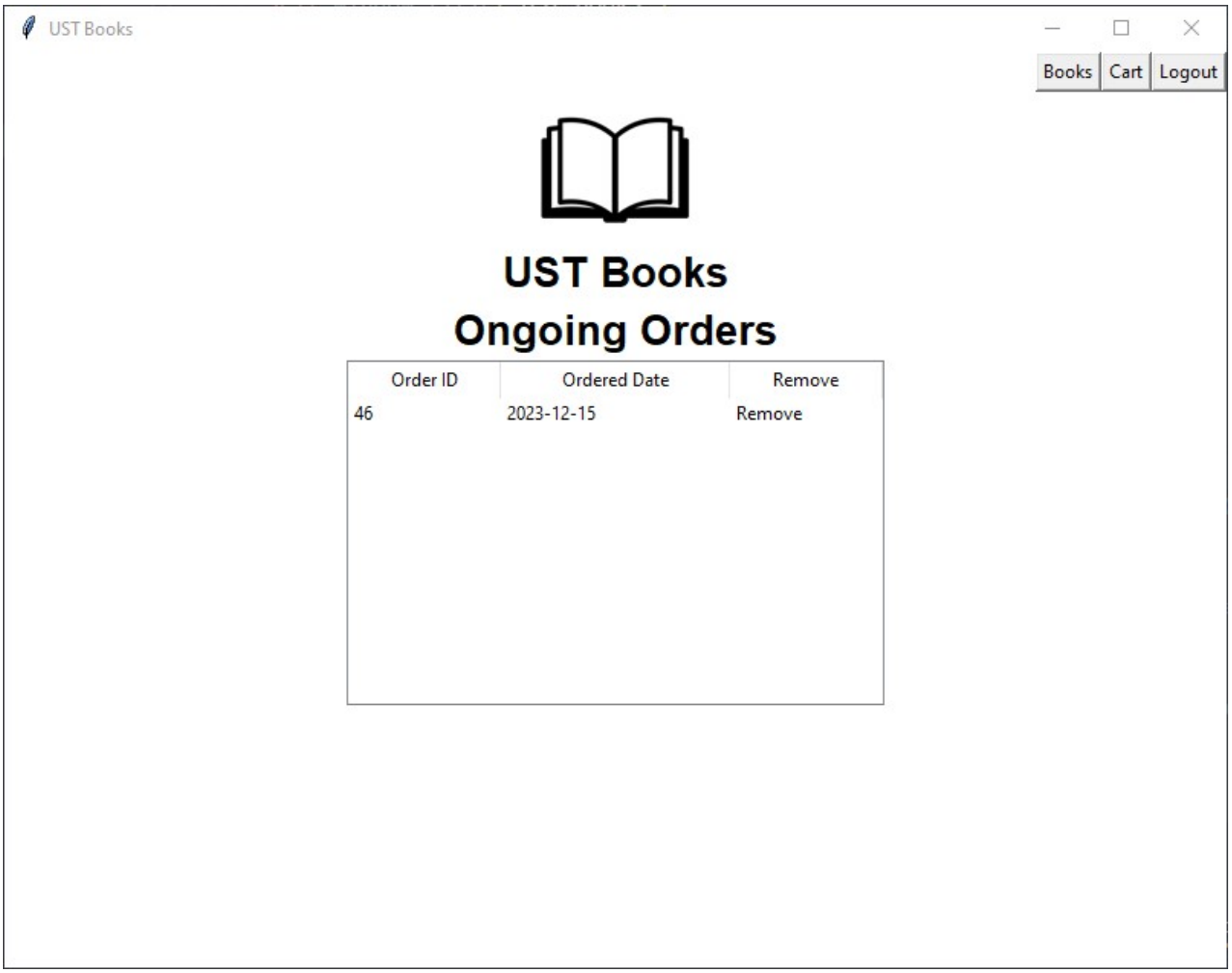

After executing a search on the search page, A list of matching books will appear in a list. Each book item displays the book title and rating, along with buttons for starting a review and adding the book to the cart. The student user can execute another search by changing the text in the box and searching again.



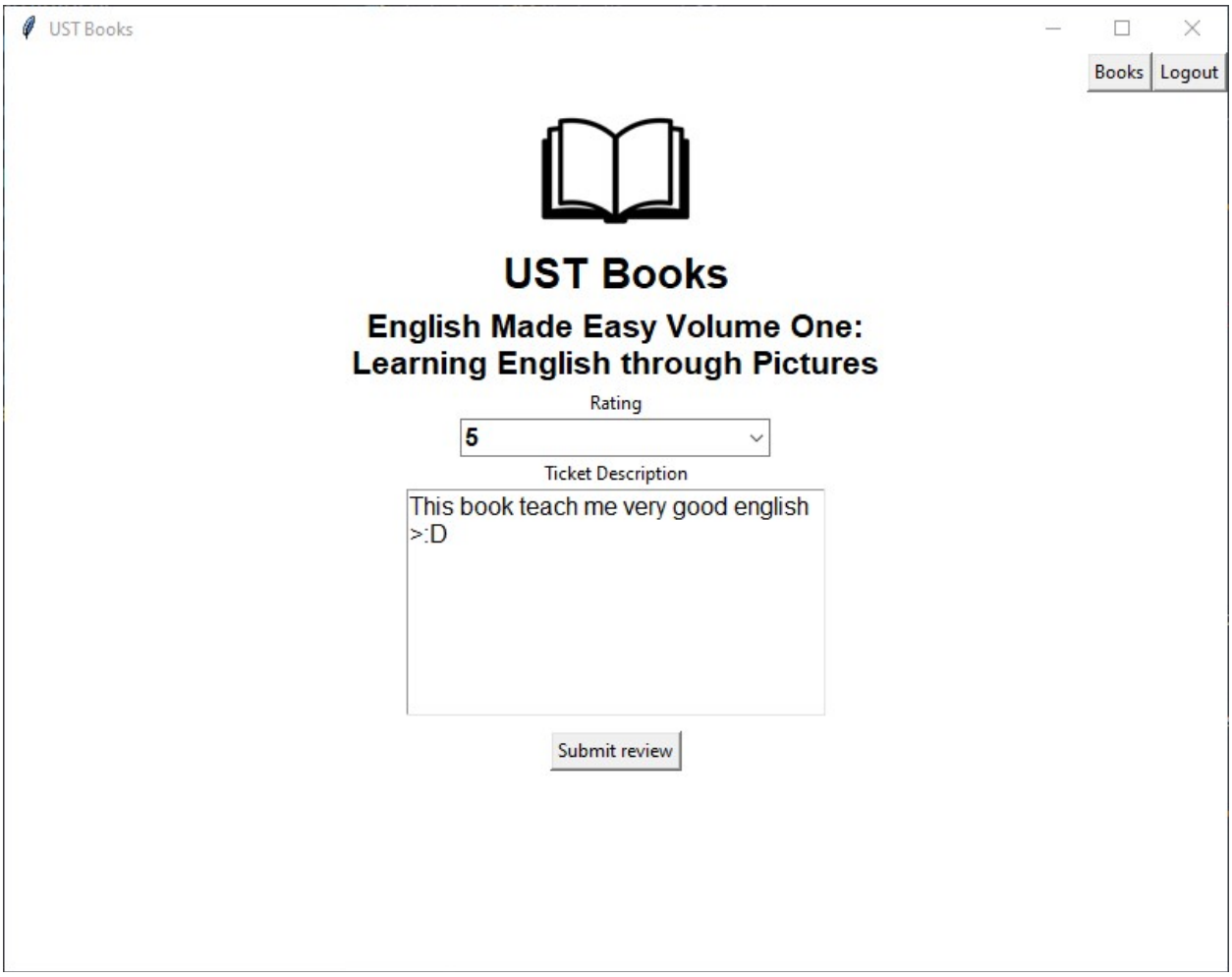

After a student has added one or more books to their cart, they can click on the cart button at the top of the page. This navigates the page to the cart. On this page, the student user can see the titles of each book in their cart, along with options to change the quantity of each book, and remove it from the cart. Under the list of books is a button labeled “checkout”.



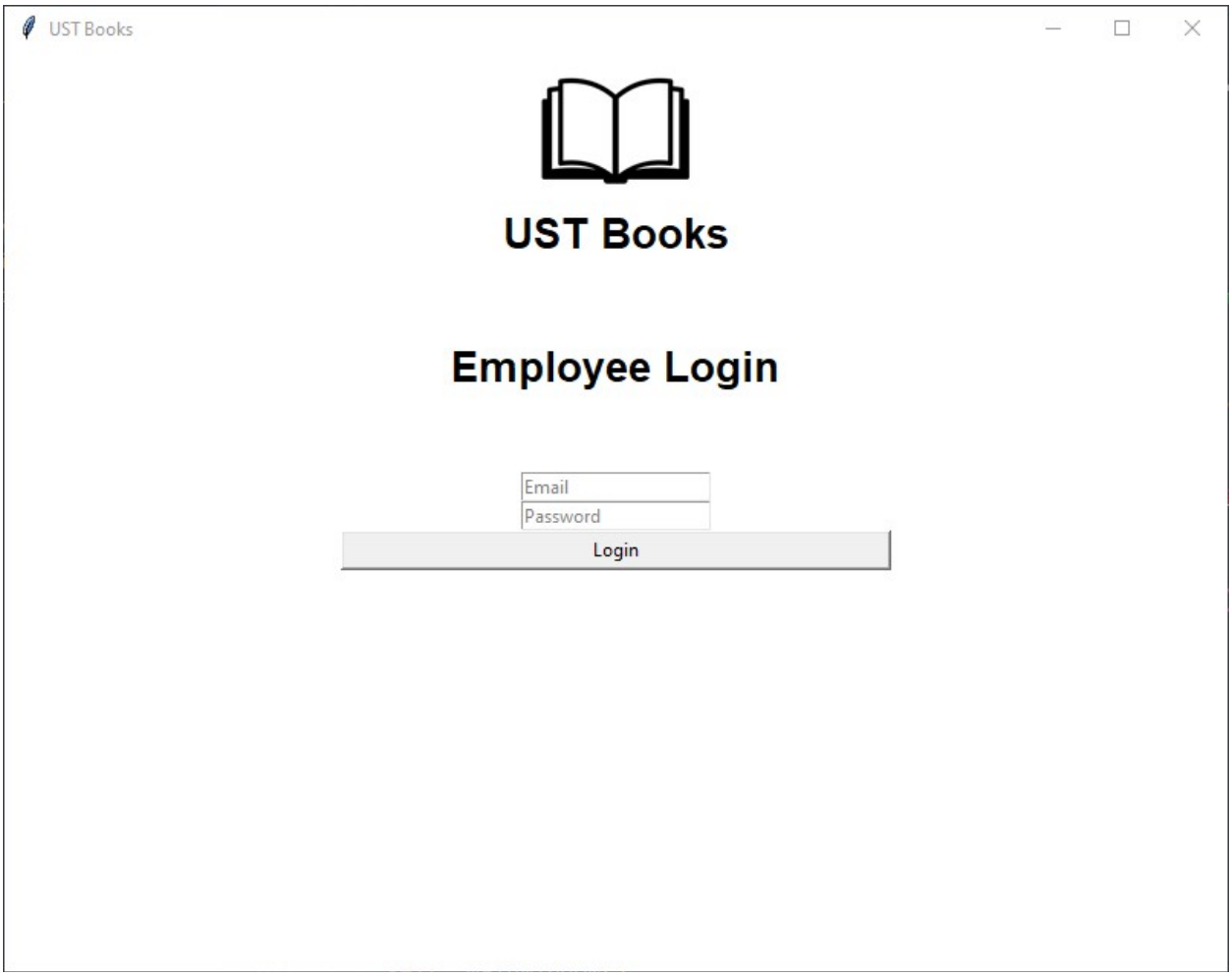

After the checkout button is clicked, two text entries and a button appear. These text entries are for an address and card number. The button is labeled “submit order” and will create an order from the cart.



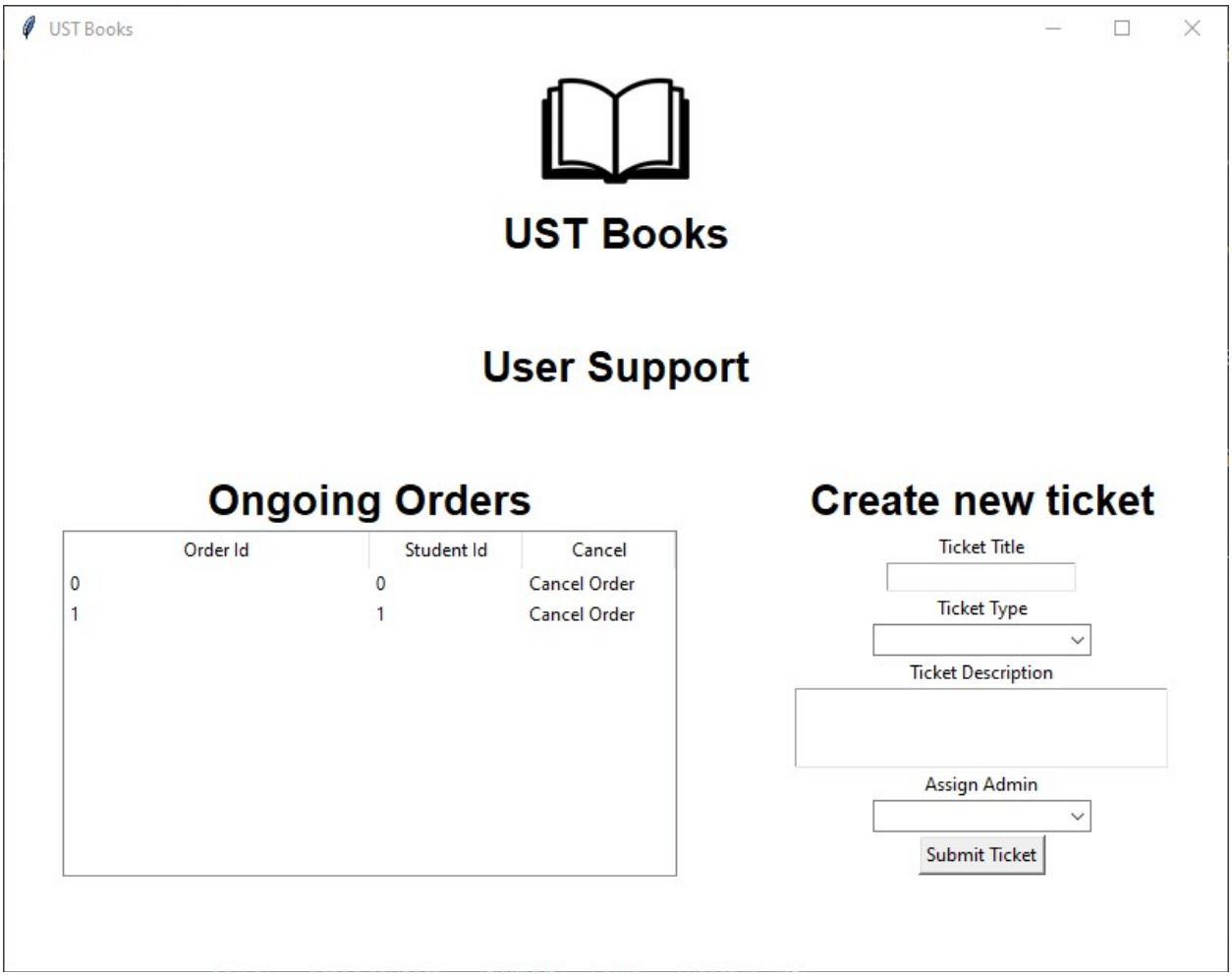

Going back to the Cart or any pervious page, there is a button for “ongoing orders”. This page with all previous ongoing orders will populate. There is also a button, “cancel” that allows students to cancel ongoing orders.



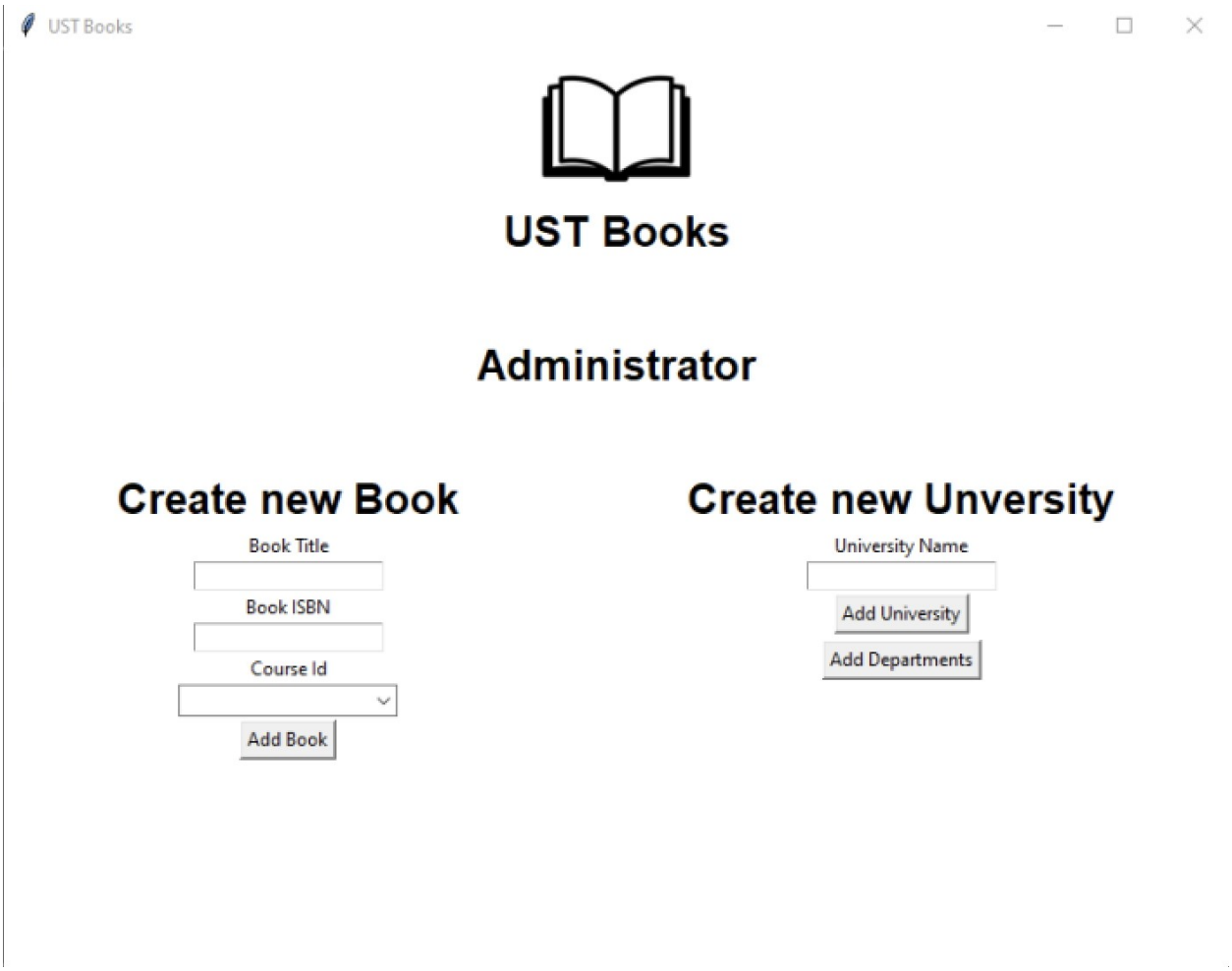

After pressing the “Review Book” button, the page navigates to the review page. On this page, the title of the book is displayed at the top. Below the title is a dropdown menu for the numerical rating of the book. Below the dropdown is a large text entry for the review. At the bottom is a button labeled “Submit Review”. Clicking this button will create a review for the book in the database.



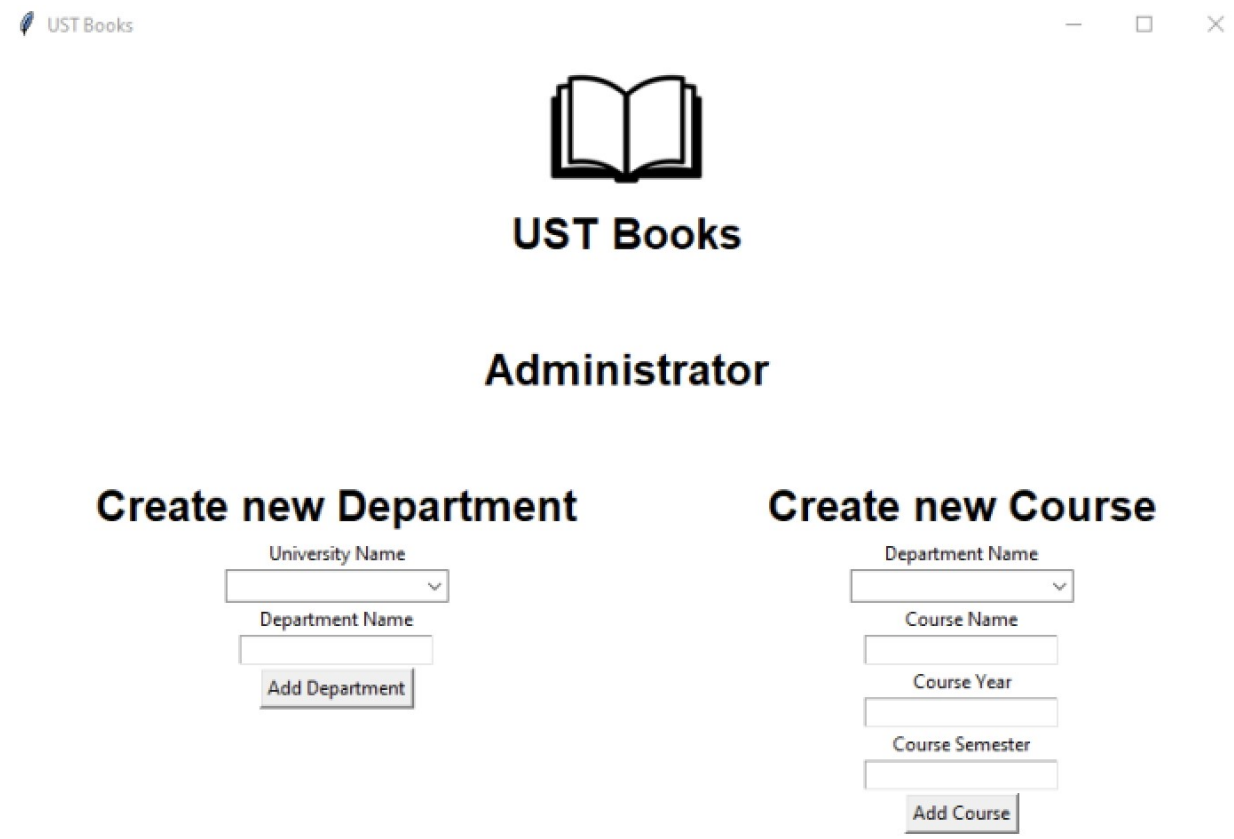

This page is the employee login page. It has two text entries for email and password. Below them is the Login button which will log the user in as an employee.



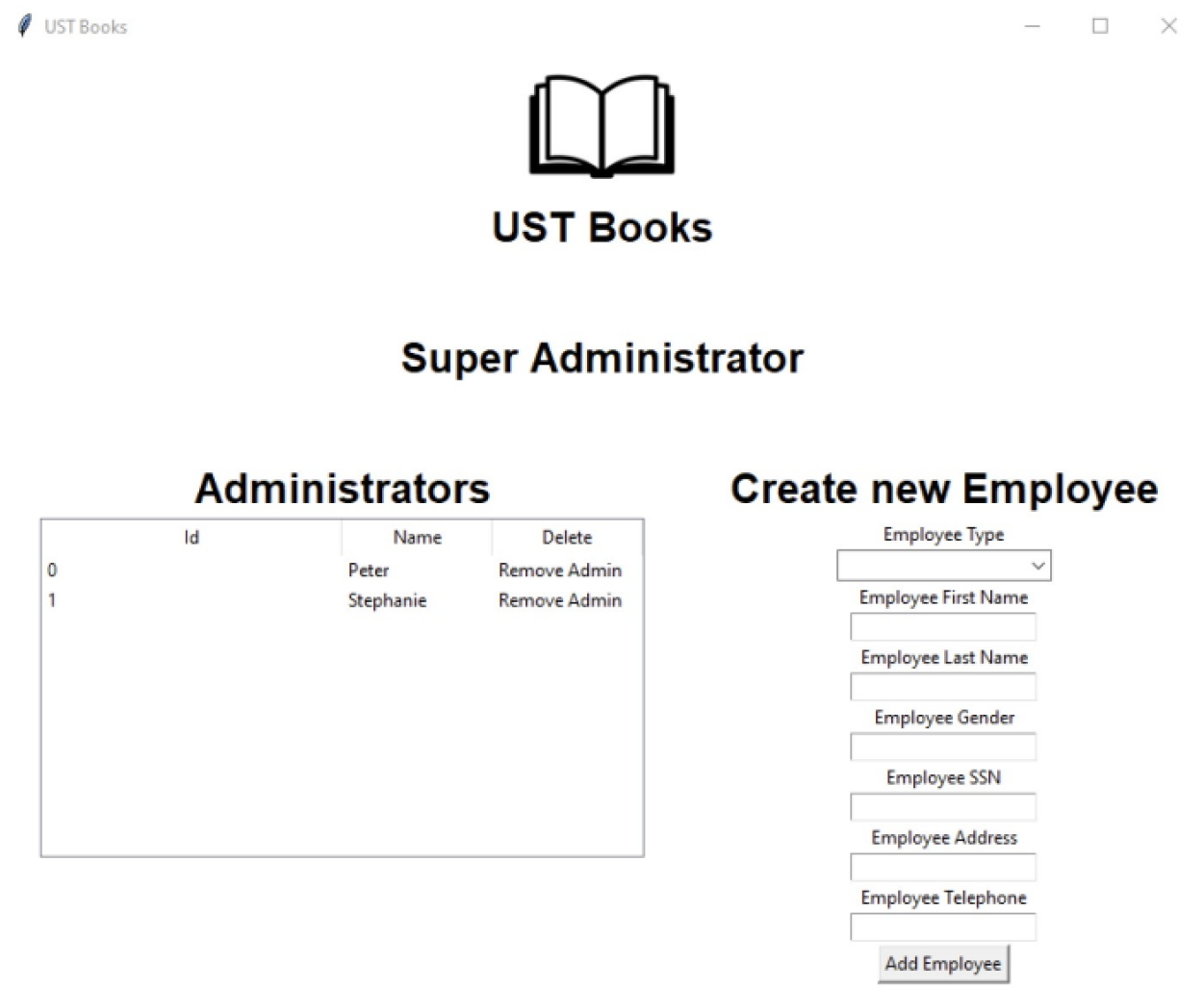

This page is the customer support page. This page shows a list of orders on the left with a form to create a ticket on the right. The table of orders displays the orders ID, the student's ID, and a button to cancel the order. The form has a text entry for the title of the ticket, a dropdown with options for the type of ticket, a large text box for the description of the ticket, and another drop down to assign an admin to the ticket. Below the form is a button to create the ticket in the database.




After logging in to an administrator account, the page is navigated to the administrator page. On this page, the user is able to create a new book with the title, ISBN, and the ID of the course it is associated with, and a “Add Book” button to create the book in the database. The user is also able to create a new university with the university name and a “Add University” button to create the university. The user can then click the button labeled “Add departments” to navigate to the departments page.




After navigating to the departments page, the administrator user can create a new department with the university name, department name, and a “Add Department” button to create the department. The user can also create a new course with the department name, course name, course year, course semester, and a “Add Course” button to create the course in the database.

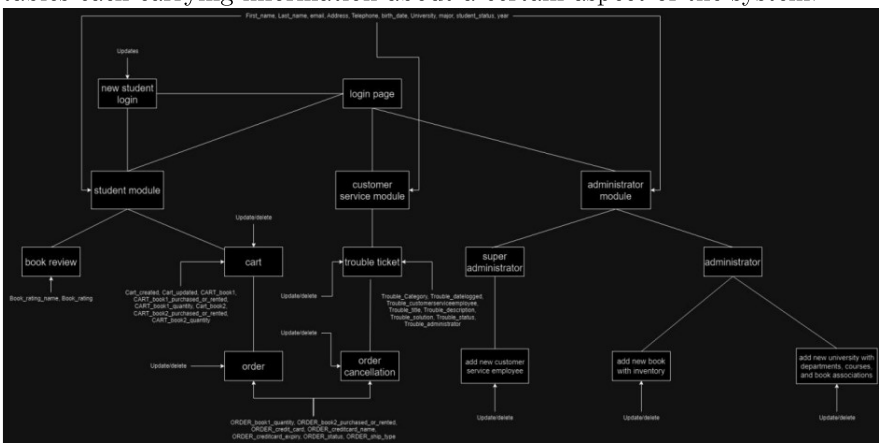



After logging in to a super administrator account, the page displays a list of administrators, and a form to create a new employee. In the list of administrators, each administrator is displayed with their id, name, and a button to remove the person as an administrator. The form allows the user to create a new employee with the employee type, first and last name, gender, SSN, address, phone number, and a “Add Employee” button to create the new employee.

# Phase 1 Documentation Project: Book Fetch

We have selected the Book Fetch project to complete. Book Fetch is a system designed to be used by students to buy or rent books. The system also allows for employee access, with there being two types of employees, customer support employees and administrators. Books can be added to the collection by administrators and can be designated as course books for classes across many universities. The system must handle there being multiple universities, each with their own selection of departments and subsequent courses for a given semester or year. Students are therefore able to buy and rent books for courses they take, as well as books not associated with their courses. The final step is the customer support tickets that can be created by both the customer support employees as well as the students.

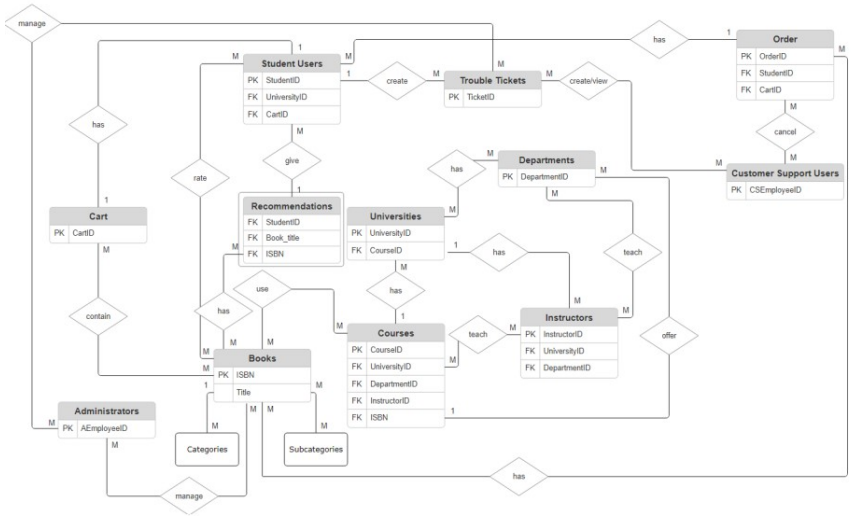
Our plan for working on the project is to initially design a high-level diagram listing the tables we think we need, as well as the connections between them. We can then start designating attributes to the tables and eliminating anomalies we can foresee occurring. Our initial thoughts are that the books themselves will make up a cluster of tables, the universities will make up a cluster, and then the users will make up yet another cluster. Each cluster will contain multiple tables each carrying information about a certain aspect of the system.




**Questions:**

1. Our group was wondering why we have the attributes (`CART_book1`, `CART_book1_purchased_or_rented`, `CART_book1_quantity`) and (`Cart_book2`, `CART_book2_purchased_or_rented`, `CART_book2_quantity`). To us, this implies that there can only be two books in a cart or two books in an order (`ORDER_book1`, `ORDER_book2`). Is this the case, or are we misinterpreting the attributes?
2. The database doesn't include user login/password information. How do we deal with logins?

# Phase 2 Documentation Project: Entity-Relation Model




**Constraints:**

**Student Users**

- Student Users give Recommendations
- Student Users create Trouble Tickets
- Student Users have Order/s
- Student Users have a Cart
- Student Rates Books

**Customer Support Users**

- Customer Support Users create/views Trouble Tickets
- Customer Support Users cancel Orders

**Administrators**

- Administrators Manage Trouble Tickets
- Administrators Manage Books

**Books**

- Books have Recommendations
- Books are used in Courses

- Books are in Categories and Subcategories

**Universities**

- Universities have Departments
- Universities have Instructors
- Universities have Courses

**Departments**

- Departments are taught by Instructors
- Departments offer Courses

**Instructors**

- Instructors teach Courses

**Courses**

- Courses have Trouble Tickets

**Trouble Tickets**

- Trouble Tickets are created by Student Users and Customer Support Users

**Cart**

- Cart contains Books

**Order**

- Order/s have Books

**Recommendations**

**Our Design Approach:**

**Entities:**

1. Student Users

- Represents a Student User by storing data about the user.
- Has StudentID as a primary key and CartID and UniversityID as foreign keys to represent the student's corresponding cart and university.

2. Instructors

- Represents an Instructor of a course by storing data about the instructor.
- Has InstructorID as a primary key and DepartmentID and UniversityID as foreign keys to represent the instructor's department and university.

3. Courses

- Represents a course by storing data about the course.
- Has CourseID as a primary key and UniversityID, DepartmentID, InstructorID, and ISBN to identify its university, department, instructor, and book.

4. Universities

- Represents a university by storing data about the university.
- Has UniversityID as a primary key and CourseID as a foreign key to support its course relations.

5. Order

- Represents an order by storing data about the order.
- Has OrderID as a primary key and StudentID and CartID to identify its student owner and corresponding cart.

6. Departments

- Represents a department.
- Has DepartmentID as its primary key.

7. Administrators

- Represents an administrator user.
- Has EmployeeID as a primary key.

8. Cart

- Represents a student user's cart.
- Has CartID as a primary key.

9. Books

- Represents a book.
- Has ISBN as a primary key.

10. Customer Support Users

- Represents a customer support user.
- Has CSEmployeeID as a primary key.

11. Trouble Tickets

- Represents customer support tickets submitted by customer support or a student.
- Has TicketID as a primary key.

12. Recommendations

- Represents a book recommended to a student user.
- Has StudentID, Book_title, and ISBN to identify the student it was recommended to, its title, and its ISBN.

**Relationships:**

1. Recommendation given to student

- A book can be recommended to many students.

2. Student user creates trouble ticket

- A student user can create many trouble tickets.

3. Student User rate book

- Many student users can give many books a rating.

4. Student User has cart

- A student user can have one cart.

5. Student User has order

- Many student users can have many orders.

6. Course has university

- A university can have many courses.

7. University has Department

- A university can have many departments.

8. University has Instructor

- A university can have many instructors.

9. Instructor teaches course

- Many instructors can teach many courses.

10. Instructor teaches department

- Many instructors can teach many departments.

11. Course uses books

- Many courses can use many books.

12. Book has order

- Many books can have many orders.

13. Administrator manages book

- Many books can have many administrators.

14. Cart contains book

- Many carts can contain many books.

15. Customer Support Users create/view trouble tickets

- Many CSEmployee can create and view many trouble tickets.

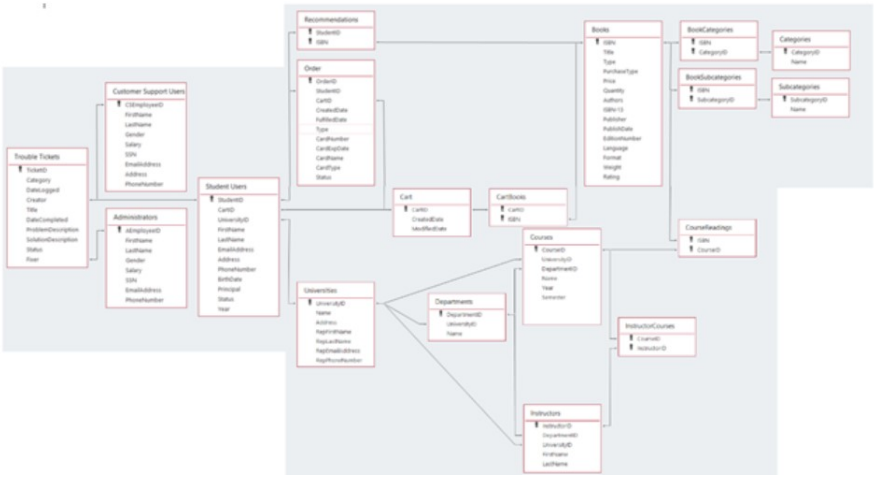
16. Book has recommendation

- Many books can be recommended many times

17. Administrator manages trouble ticket

- Many administrators can manage many trouble tickets

# Phase 3 Documentation Project: Relational Model



**Entities:**

**Student Users**

- The primary key is `StudentID`
- The foreign keys are `UniversityID` (`Universities.UniversityID`) and `CartID` (`Carts.CartID`)

**Cart**

- The primary key is `CartID`
- No foreign keys

**CartBooks**

- The primary key is [`CartID`, `ISBN`]
- The foreign keys are `CartID` (`Carts.CartID`) and `ISBN` (`Books.ISBN`)

**Administrators**

- The primary key is `AEmployeeID`
- No foreign keys

**Recommendations**

- The primary key is [`StudentID`, `ISBN`]
- The foreign keys are `StudentID` (`Students.StudentID`) and `ISBN` (`Books.ISBN`)

**Books**

- The primary key is `ISBN`
- No foreign keys

**Courses**

- The primary key is `CourseID`
- The foreign keys are `UniversityID` (`Universities.UniversityID`) and `DepartmentID` (`Departments.DepartmentID`)

**CourseReadings**

- The primary key is [`CourseID`, `ISBN`]
- The foreign keys are `CourseID` (`Courses.CourseID`) and `ISBN` (`Books.ISBN`)

**Universities**

- The primary key is `UniversityID`
- No foreign keys

**Departments**

- The primary key is `DepartmentID`
- The foreign key is `UniversityID` (`Universities.UniversityID`)

**Instructors**

- The primary key is `InstructorID`
- The foreign keys are `UniversityID` (`Universities.UniversityID`) and `DepartmentID` (`Departments.DepartmentID`)

**InstructorCourses**

- The primary key is [`InstructorID`, `CourseID`]
- The foreign keys are `InstructorID` (`Instructors.InstructorID`) and `CourseID` (`Courses.CourseID`)

**Order**

- The primary key is `OrderID`
- The foreign keys are `StudentID` (`Students.StudentID`) and `CartID` (`Carts.CartID`)

**Customer Support Users**

- The primary key is `CSEmployeeID`
- No foreign keys

**Administrators**

- The primary key is `AEmployeeID`
- No foreign keys

**Trouble Tickets**

- The primary key is `TicketID`
- The foreign keys are `Creator` (`CustomerSupportUsers.CSEmployeeID` or `Students.StudentID`) and `Fixer` (`Administrators.AEmployeeID`)

**Categories**

- The primary key is `CategoryID`
- No foreign keys

**BookCategories**

- The primary key is [`CategoryID`, `ISBN`]
- The foreign keys are `CategoryID` (`Categories.CategoryID`) and `ISBN` (`Books.ISBN`)

**Subcategories**

- The primary key is `SubcategoryID`
- No foreign keys

**BookSubcategories**

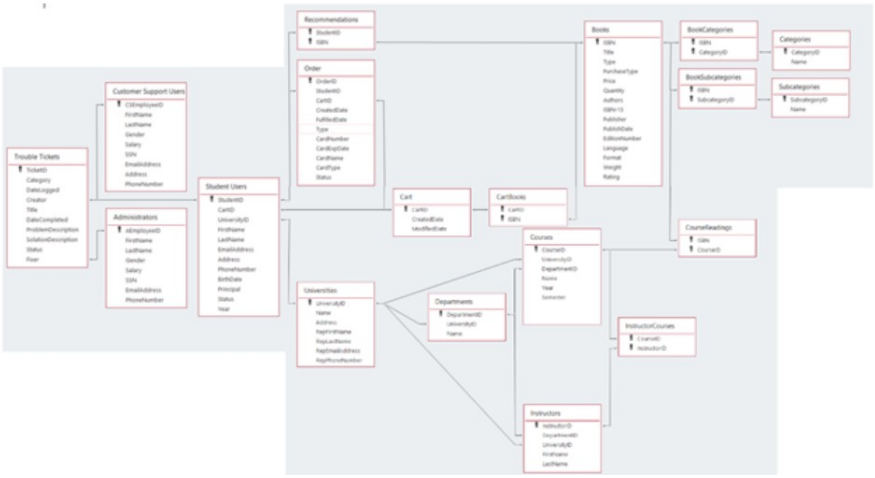
- The primary key is [`SubcategoryID`, `ISBN`]
- The foreign keys are `SubcategoryID` (`Subcategories.SubcategoryID`) and `ISBN` (`Books.ISBN`)

**Constraints:**

- Student Users give Recommendations
- Student Users create Trouble Tickets
- Student Users have Order/s
- Student Users have a Cart
- Student Rates Books
- Customer Support Users create/views Trouble Tickets
- Customer Support Users cancel Orders

- Administrators Manage Trouble Tickets
- Administrators Manage Books
- Books have Recommendations
- Books are used in Courses
- Books are in Categories and Subcategories
- Universities have Departments
- Universities have Instructors
- Universities have Courses
- Departments are taught by Instructors
- Departments offer Courses
- Instructors teach Courses
- Cart contains Books
- Orders have Books

# Phase 4 Documentation Project: Physical Design




**This above is Before the Changes**


**This above is After the Changes**

**Changes:**

- Added missing variables in the database (e.g., Cart and ordered books).
- Redesigned the instructor relation with the Course so that only one instructor can teach a course.
- Added a Reviews table.
- Added a Super-admin table.
- Changed the Order table to allow the deletion of a cart.

**UI Design:**

We will be using Electron and Node.js to create a GUI-based experience on Windows that is similar to a website. Our database will be served by https://supabase.com/Supabase, which offers a very generous free tier for us to use and will make connecting our application to our database very simple and straightforward. It's worth noting Supabase uses PostgreSQL as their SQL server, which is a little bit different from MySQL, so minor syntax is different (serial instead of $\text{auto}_i ncrement$).

The user will be presented with a login form, where they will login as one of student, customer service employee, administrator, or super administrator using their email and password. Upon logging in, the UI will change depending on what role the individual has.

**User:**

For users, they will be shown a search bar to search for books that will appear as they type. There will also be a catalog of books that appear below the search bar at all times.

While viewing books, their price and descriptions will be shown along with an icon to add the book to their cart.

They will have an icon on the top right of a cart, that allows them to view their cart upon pressing.

On this screen, they will be able to remove books from their cart with an "x" icon and change the quantity via the drop-down menu.

While viewing books, they will all have an icon that when pressed allows students to submit a review and/or rate the book.

There will be a drop-down settings icon at the top right that would allow students to view their order history. The drop-down will also have a help button which will allow students to submit tickets for any issues they might have. The settings dropdown will also include a sign-out button.

**Customer Service Employees:**

Customer service employees will be shown a screen with all the current active tickets so that they will be able to monitor the ticket queue.

Clicking on a ticket will show more details about the ticket (e.g., Title, description, student who submitted the ticket's information). Here they will be able to submit solutions to the tickets.

They will also be able to click student icons and be able to do mundane tasks like reset a user's password should they forget it or canceling an order should a user want that.

**Administrators:**

The Administrators will be shown an administrator page that will allow them to select specific parts of the database that they might want to change or add to. They will be able to add new books to the inventory, universities, departments, and courses. For example, if an administrator would like to add a book to the catalogue, they can click a button that will bring up a form where they are able to submit a new book to be added to the catalogue.

**Super Administrators:**

Super administrators will have the added benefit of being able to add new customer service employees via an icon to add employees.