

DSCI-D 532 Applied Database Technologies

Project – Online Library Management System

Final Project Part 3

Team 02

Team Names – Hemraj Yadav, Mustafa Alsaegh

Web App Design

Part 1. Web App Architecture

Overview

Model: We are using Sqlite3 database for storing data and using Django's ORM to define database schema and create model for book management.

View: We are using Django as web framework in Python which having inbuilt features for creating web applications including a powerful template engine for generating HTML views for the front end. Django widget tweaks is used to render form fields in templates. We will be creating views for the different actions the users can perform, such as adding, editing, viewing, searching for the book. Each view maps to a URL, which contains the logic for retrieving data from the database, processing user input, and rendering the appropriate HTML template.

Controller: We have used Django as controller as well. Django is having powerful ORM (Object-Relational Mapping) for interacting with databases, and a robust system for handling URL routing, request handling, and middleware. We have also used sqlparse a non-validating SQL parser for Python to provide support for parsing, splitting and formatting SQL statements. On top of that using pytz module allows for date-time conversion and time zone and ASGI for Python asynchronous web apps and servers to communicate with each other.

1. Architecture

Our App Architecture is based on below components using Django, Python, and SQLite3 as the database. It is based on Model-View-Controller (MVC) design pattern. **Figure 1**

- Model
- Views
- Template
- Forms
- URLs Configuration
- Static Files

- Settings

Online Library Management System

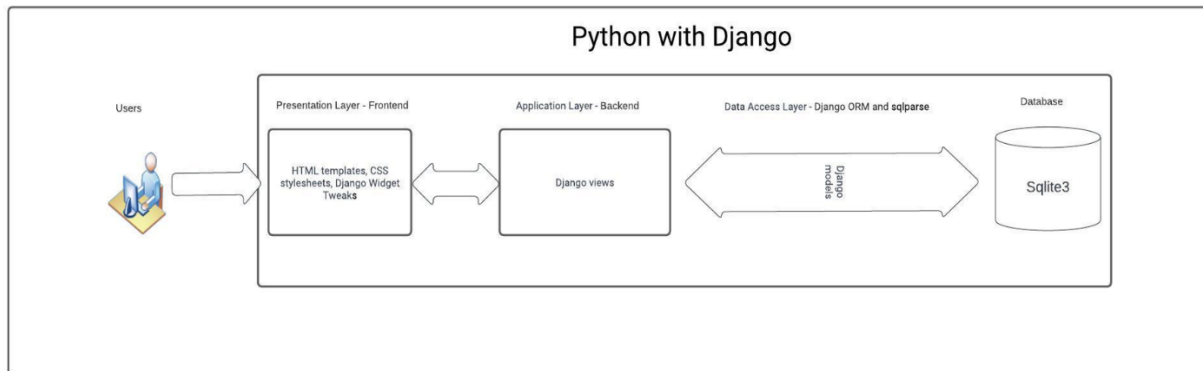


Figure 1. Web architecture

2. Back- End

At the backend, we used Django to build the functionality of our web application. Django allowed us to create a clean and structured codebase, making it easy to manage and maintain our application. We used Django's built-in ORM (Object-Relational Mapping) to map our database schema to Python objects, making it easier to read and manipulate data from the database. Additionally, we implemented various features such as authentication, CRUD (Create, Read, Update, Delete) operations.

```
def is_admin(user):
    return user.groups.filter(name='ADMIN').exists()

def afterlogin_view(request):
    if is_admin(request.user):
        return render(request, 'library/adminafterlogin.html')
    else:
        return render(request, 'library/studentafterlogin.html')

@login_required(login_url='adminlogin')
@user_passes_test(is_admin)
def addbook_view(request):
    #now it is empty book form for sending to html
    form=forms.BookForm()
    if request.method=='POST':
        #now this form have data from html
        form=forms.BookForm(request.POST)
        if form.is_valid():
            user=form.save()
            return render(request, 'library/bookadded.html')
    return render(request, 'library/addbook.html',{'form':form})
```

Figure 2. Back-End fun

2.1 Database

We are using Sqlite3 for storing our data. We are using **SQLite3** a popular lightweight, serverless, and self-contained relational database management system. We chose SQLite3 as it is particularly useful for

our applications needed a local database which can be easily embedded within the application. SQLite3 does not require a separate server process to be running and part of the application deployment itself, For local development, we are using the below code to connect our backend Django application to the locally stored Sqlite3 database. See **Figure 2.1**.

```
# Database
# https://docs.djangoproject.com/en/3.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

Figure 2.1. SQLite3 engine

Name	Type	Schema
Tables (14)		
> auth_group		CREATE TABLE "auth_group" ("Id" Integer NO
> auth_group_permissions		CREATE TABLE "auth_group_permissions" ("
> auth_permission		CREATE TABLE "auth_permission" ("Id" Integ
> auth_user		CREATE TABLE "auth_user" ("Id" Integer NO
> auth_user_groups		CREATE TABLE "auth_user_groups" ("Id" Int
> auth_user_user_permissions		CREATE TABLE "auth_user_user_permissions
> django_admin_log		CREATE TABLE "django_admin_log" ("Id" Int
> django_content_type		CREATE TABLE "django_content_type" ("Id"
> django_migrations		CREATE TABLE "django_migrations" ("Id" Int
> django_session		CREATE TABLE "django_session" ("session_
> library_book		CREATE TABLE "library_book" ("Id" Integer N
> library_issuedbook		CREATE TABLE "library_issuedbook" ("Id" Int
> library_studentextra		CREATE TABLE "library_studentextra" ("Id" li

Figure 2.2. SQLite3 data tables

2. Front - End

we used Django's built-in templating system to create HTML templates that are rendered by the backend. These templates include dynamic content using template tags and variables that are provided by the backend. To create a user interface for the web application, we used **HTML**, and **CSS** to design and style the templates. The frontend communicates with the backend using **HTTP** requests and **SQLite** data.

```

.signup-form form a {
  color: #5ca1d3;
  text-decoration: none;
}
.signup-form form a:hover {
  text-decoration: underline;
}
</style>
</head>
<body>
  {% include "library/navbar.html" %}
  <br>
  <br>
  <center><h3 style="margin-bottom: 0px;"class = 'alert alert-success'>IU Faculty Members</h3></center>
  <div class="signup-form">
    <form method="post" class="form-horizontal">
      <center><h2>Admin Login</h2></center>

```

Figure 3. Frontend design HTML + CSS

5. App Deployment/ Connections

We are deploying the web application locally, which means that it will only be accessible from our local computer. To do so, we use the command-line interface and navigate to the project directory. From there, we run the command "\$python manage.py runserver". This command starts the development server and allows us to access the application using a web browser at <http://localhost:8000>. We can test the application and make sure that everything is working as expected. **Figure 4**

```

D-532-DBT $cd librarymanagement
librarymanagement $python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
April 16, 2023 - 01:46:58
Django version 3.0.5, using settings 'librarymanagement.settings'
Starting development server at http://127.0.0.1:8000/

```

Figure 4. Deploying web app locally

6. App Interactivity

This app is going to be very interactive for admin and users. Admin will be able to perform all below user tasks along with Admin work like management of books lifecycle and user management. For example create and manage admin account and login, view and add book and delete. Users will be able to create account and Login, view book from listings, and delete existing books.

Part 2. Web App Layout

We have used HTML to design web page layouts for our application. There will be a total of 4 main pages, as shown in the included figures below.

The architecture of the web pages is as follows:

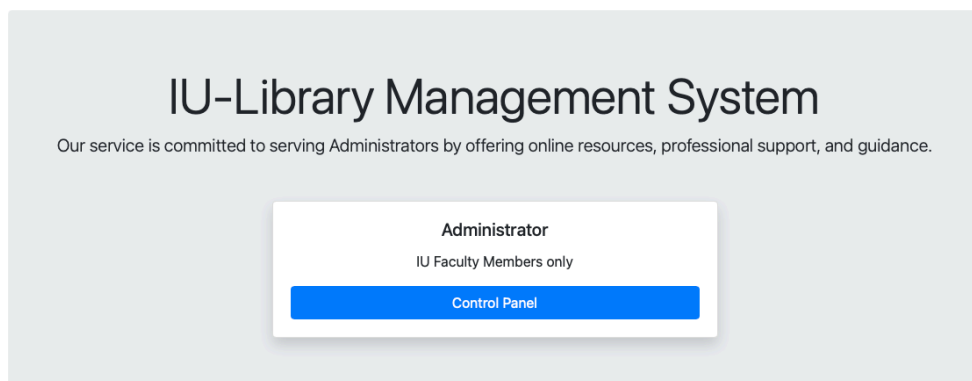
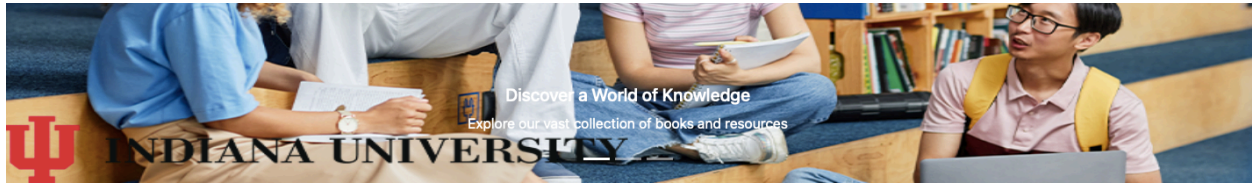
- 1) User will be able to access the control panel from the Home Page.

- 2) User will be able to log in to the application from the “login” page.
- 3) User will be able to add, view, and delete a record from the “Control Panel” following a successful log in.

The color scheme of the application, as well as the components of each page are shown below.

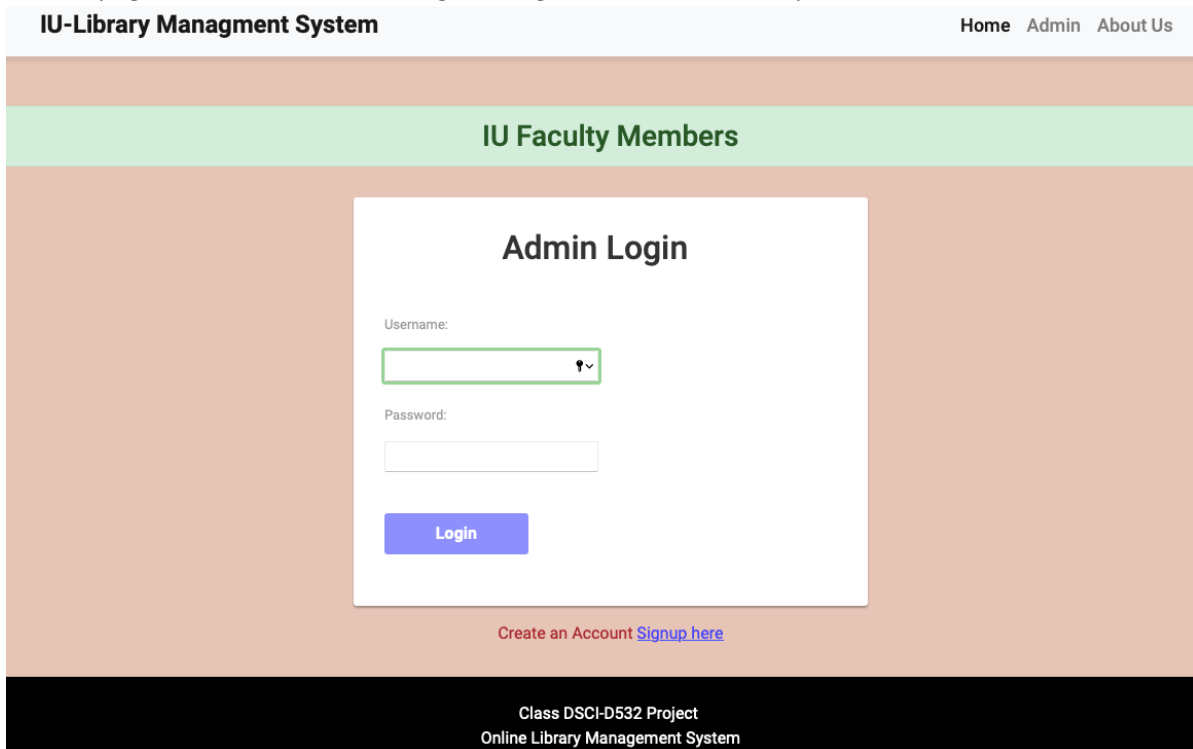
Home Page:

On this page, users can access the control panel.



Login Page:

On this page, users will be able to log in using their username and password.



Control Panel:

On this page, users will be able to add and view books in the library, as well as delete books.

The screenshot shows the LMS Admin Control Panel. The top navigation bar includes a home icon, 'LMS Admin Control Panel', a 'Book' dropdown menu, 'LOGOUT', and 'About Us'. The 'Book' dropdown menu is open, showing 'Add Book' and 'View Book' options. The main content area has a light gray background and features a large greeting 'Hello Mustafa !' followed by the text 'You have the ability to add and view books in the library, as well as delete books.' Below this, under the heading 'Available Features', there are three blue buttons: 'Add Book To Library', 'View Available Book', and 'Delete Book'.

Add Books:

On this page, users can add a book and select four categories related to the book.

The screenshot shows the 'ADD BOOK IN LIBRARY' form. The top navigation bar is the same as the previous page. The main content area has a light blue background. The form is centered and has a white background. It contains the following fields: 'Name:' with a text input, 'Isbn:' with a text input and a dropdown arrow, 'Author:' with a text input, and 'Category:' with a dropdown menu showing 'Education' and a green plus icon. Below these fields is a blue 'ADD' button. At the bottom of the form, there is a link 'View Available Books'. The footer of the page says 'Class DSCI-D532 Project'.

View Books:

Here, users can explore available books.

 LMS Admin Control Panel


Book ▾

LOGOUTAbout Us

Book Name	ISBN	Author	Category
Artificial Intelligence: A Modern Approach	5543	Stuart Russell and Peter Norvig	education
Deep Learning	2325	Ian Goodfellow, Yoshua Bengio, and Aaron Courville	education
Human Compatible: Artificial Intelligence and the Problem of Control	56775	Stuart Russell	education
Fairy Tale	25367	Stephen King	education
Rebooting AI: Building Artificial Intelligence We Can Trust	74456	Gary Marcus and Ernest Davis	education
Termination Shock	38748	Neal Stephenson	history
To Kill a Mockingbird	9780446310789	Harper Lee	Fiction
The Great Gatsby	9780743273565	F. Scott Fitzgerald	Fiction
1984	9780451524935	George Orwell	Science Fiction
Pride and Prejudice	9780141439518	Jane Austen	Romance
The Catcher in the Rye	9780316769488	J.D. Salinger	Fiction
Animal Farm	9780451526342	George Orwell	Satire
The Hobbit	9780547928227	J.R.R. Tolkien	Fantasy
The Lord of the Rings	9780544003415	J.R.R. Tolkien	Fantasy
The Hitchhiker's Guide to the Galaxy	9780345391803	Douglas Adams	Science Fiction
The Da Vinci Code	9780307474278	Dan Brown	Thriller
The Girl with the Dragon Tattoo	9780307269751	Stieg Larsson	Mystery
The Hunger Games	9780439023481	Suzanne Collins	Science Fiction
The Road	9780307387899	Cormac McCarthy	Fiction
The Alchemist	9780062315007	Paulo Coelho	Fiction
The Kite Runner	9781594631931	Khaled Hosseini	Fiction
The Girl on the Train	9781594634024	Paula Hawkins	Mystery
Gone Girl	9780307588371	Gillian Flynn	Mystery
A Game of Thrones	9780553593716	George R.R. Martin	Fantasy

Delete Books:

Here, users can delete books.

 LMS Admin Control Panel

Book ▾

LOGOUTAbout Us

DELETE BOOK

Name:

Isbn:

Author:

Category:

Education

DELETE

[View Available Books](#)

Part 3. Individual and Team Work Assessment

References:

<https://docs.djangoproject.com/en/3.0/ref/settings/#databases>

<https://docs.djangoproject.com/en/3.0/howto/static-files/>

<https://docs.djangoproject.com/en/3.0/topics/i18n/>

<https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/>