



King Fahd University of Petroleum & Minerals  
College of Computer Sciences and Engineering

**SWE 363: Web Engineering and Development (Term 221)**

**Phase3: Implementation**

**Team#7:**

Project title: Online Library Website: Bookify

Prepared by:

Adi Sendi 201779270

Tareq Saleh Alqarawi 201826040

Mohammed Alwaleed Abushwarib 201935730

Fadel Hassan Abbas 201951290

# 1. Implementation Details

## ***I- Technologies and Tools:***

### **-Interface:**

- HTML
- CSS
- JavaScript
- Bootstrap
- J-Quarry


### **-Backend:**

- Node.js
- SQLite
- Express
- Express session
- Express validator
- Multer
- Bcrypt
- Node man
- Nunjucks
- Type Script

### **-Other:**

- Visual Studio
- Notepad++
- Microsoft-Teams


## II- GUI Screen Shots:



[HOME](#) [BOOKMARK](#) [REQUEST](#)


[LOG IN](#)

# Nothing Better Than To Read



Start reading at our library now!  
Search your book by name or ISBN

[Search Book](#) [Don't Find Your Book? Submit a request](#)

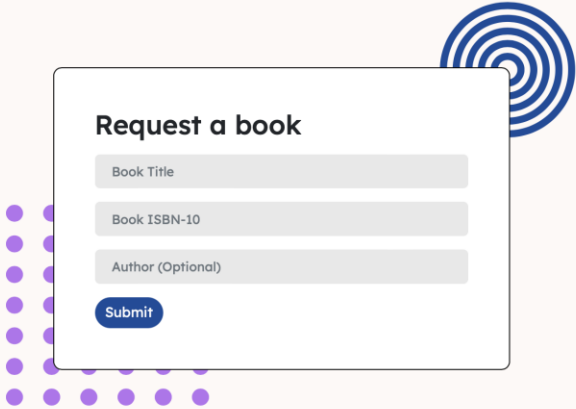



## YOU CAN'T FIND YOUR BOOK?

[Submit a request for a book](#)

### Request a book

[Submit](#)





[Quick Links](#)

[Follow us on Links](#)

Bookify

## Welcome back

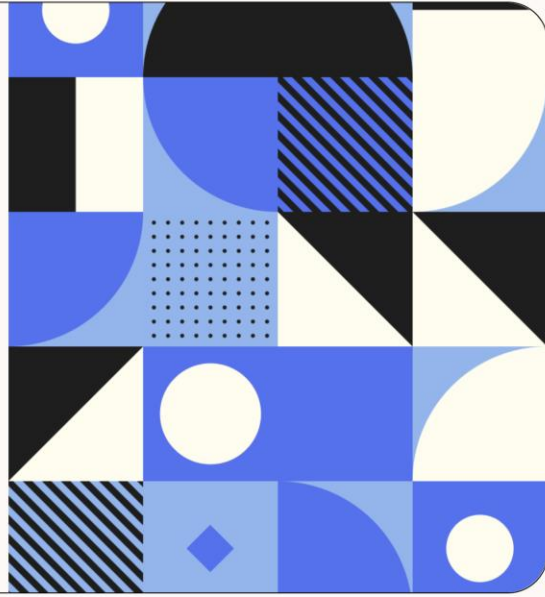
Welcome back! Please enter your details

Username

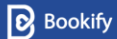
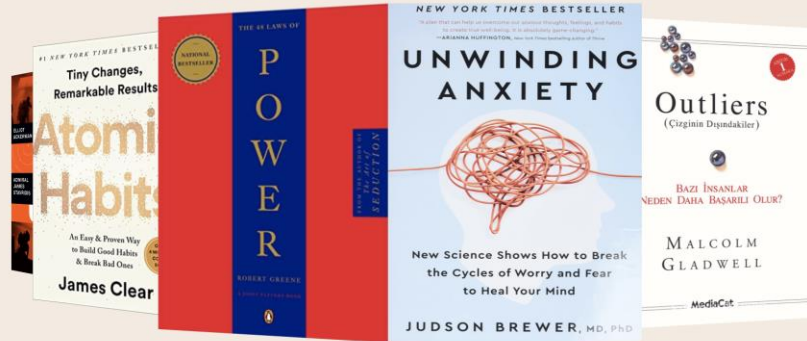
Password

Log in

Don't have an account? Sign up for free



# TRENDING BOOKS



Bookify is an online library that brings you books from all over the world and lets you enjoy them.

### Quick Links

[Home](#)  
[Books](#)  
[Request](#)  
[About us](#)

### Follow us on Links





## Create an account

Start your reading journey with us today!

Name \*

Username \*

Email \*

Password \*

Confirm your password \*

Log in

Already have an account? [Log in](#)



[HOME](#) [BOOKMARK](#) [REQUEST](#)

LOG IN

## Request a book

### Request a book

Submit

```

1  {% extends "base.html" %}
2  {% block title %}Login Bookify{% endblock %}
3  {% block body %}
4  <section class="p-5 p-lg-0 pt-lg-5 text-sm-star">
5      <div class="container login-container">
6          <div class="d-sm-flex">
7              <div class="login-form-side">
8                  <a href="">
9                      
10                 </a>
11             <div class="form-holder form-holder-login">
12                 <div class="form-content form-content-login">
13                     <div class="form-items form-items-login">
14                         <h1>Welcome back</h1>
15                         <p>Welcome back! Please enter your details</p>
16                         <form class="requires-validation" action="/login" method="post" novalidate>
17                             {% if error %}
18                             <div class="alert alert-danger" role="alert">
19                                 <ul>
20                                     <li>{{error}}</li>
21                                 </ul>
22                             </div>
23                             {% endif %}
24                             <div class="col-md-12">
25                                 <input class="form-control" type="text" name="username" placeholder="Username"
26                                     value="{{username}}" required />
27                             </div>
28                             <div class="col-md-12">
29                                 <input class="form-control" type="password" name="password" placeholder="Password"
30                                     required />
31                                 <div class="invalid-feedback">
32                                     Password field cannot be blank!
33                                 </div>
34                             </div>
35                             <div class="form-button mt-3">
36                                 <button id="submit" type="submit" class="btn btn-primary btn-primary-login">
37                                     Log in
38                                 </button>
39                                 <p>
40                                     Don't have an account?<strong><a href="/register"> Sign up for free</a></strong>
41                                 </p>
42                             </div>
43                         </form>
44                     </div>

```

```

1  {% extends "layout.html" %}
2  {% block title %}Search Bookify{% endblock %}
3  {% block content %}
4  <section class="search-section">
5      <h1>Search</h1>
6      <div class="search-book-search-container">
7          <form class="search-book-search-form" role="search">
8              <input class="form-control search-bar my-1 mb-1 p-lg-2 search-form-control" type="search" placeholder="Search"
9                  aria-label="Search" value="{{term}}"/>
10             <button class="btn form-control submit-button p-3" type="submit">
11                 Search Book
12             </button>
13         </form>
14     </div>
15     <div class="search-container">
16         {% for book in books %}
17         <div class="container search-book-container">
18             <div class="search-BookContainer">
19                 <div class="search-BookingContainer">
20                     
21                 </div>
22                 <div class="search-book-info-card">
23                     <h3 class="search-book-name"><strong>{{book.title}}</strong></h3>
24                     <h4 class="search-book-author"><strong>By {{book.authors}}</strong></h4>
25                     <p class="search-book-publish">First published: {{book.released_date}}</p>
26                     <!-- <p class="search-book-pages">{{book.pages}} Pages</p> -->
27                 </div>
28                 <div class="search-right-side-card">
29                     <a href=""></a>
30                     <button class="search-view-button" onclick="location.href='/book/{{book.book_isbn}}'">View</button>
31                     <button class="search-read-button" onclick="location.href='/book/{{book.book_isbn}}'">Read</button>
32                 </div>
33             </div>
34         </div>
35         {% endfor %}
36     </div>
37 </section>
38 {% endblock %}

```

```

/*****
*****
| | | | | Register css
*****
*****/

```

```

.login-container{
    margin-bottom: 3rem;

    max-width:1300px;
    max-height:700px;
}

.login-container {
    display: flex;
    flex-direction: row;
}

.login-form-side {
    text-align: left;
    border-top: 1px solid black;
    border-left: 1px solid black;
    border-bottom: 1px solid black;
    border-bottom-left-radius: 3rem;
    border-top-left-radius: 3rem;
}

.login-pattern-side {
    width: 50%;
    border-top: 1px solid black;
    border-right: 1px solid black;
}

```

```

265 | /*****
266 | /*****/
267 | /* Request */
268 | .request-sec{
269 | | padding-top: 2rem;
270 | }
271 | .wrapper{
272 | | position: relative;
273 | margin-right: 3rem;
274 | height: 35rem;
275 | top: -8rem;
276 | }
277
278
279 | .dashed-circle{
280 | | position: absolute;
281 | | z-index: 1;
282 | | width: 10rem;
283 | | right: -4.8rem;
284 | | top: 4.2rem;
285 | }
286
287 | .dotted-square{
288 | | position: absolute;
289 | | z-index: 2;
290 | | width: 15rem;
291 | | bottom: 1rem;
292 | | left: -3.5rem;
293 | }
294 | .form-body{
295 | | margin-top: 9rem;
296 | | position: relative;
297 | | z-index: 3;
298 | }
299
300
301 | .btn.form-control.submit-button{
302 | | width: 10rem;
303 | | border-radius: 50rem;
304 | | color: white;
305 | | margin-top: 1rem;
306 | | background-color: var(--primary-blue-color);
307 | }

```



```

/*****
****
| | | | | search css
****
*****/

.search-book-container {
  background-color: #fff;
  border: 1px solid black;
  padding: 1rem;
  box-shadow: 0 0 5px rgba(0, 0, 0, 0.25);
  border-radius: 1rem;
  margin-top: 1rem;
  margin-left: 1rem;
}

.search-book-container:active {
  background-color: rgb(80, 97, 175);
}

.search-BookContainer {
  display: flex;
  flex-direction: row;
  align-items: center;
  margin-right: 0.5em;
}

.search-BookingContainer {
  margin-right: 0.5em;
}
```

```

{% extends "layout.html" %}
{% block title %}Add a book{% endblock %}
{% block content %}
<main>
  <section class="add-book-section">
    <h1>Add a book</h1>
    <div class="form-holder mt-5">
      <div class="form-content">
        <div class="form-items">
          <h2></h2>
          <form class="requires-validation" action="/addbook" method="POST" enctype="multipart/form-data" novalidate>
            <div class="col-md-12">
              <input
                class="form-control"
                type="text"
                name="title"
                placeholder="Book Title"
                value="{{title}}"
                required
              />
            </div>
            <div class="col-md-12">
              <input
                class="form-control"
                type="text"
                name="authors"
                placeholder="Author Name"
                value="{{authors}}"
                required
              />
            <div class="valid-feedback">ISBN field is valid!</div>
            <div class="invalid-feedback">ISBN field cannot be blank!</div>
          </div>
          <div class="col-md-12">
            <input
              class="form-control"
              type="text"
              name="isbn"
              placeholder="ISBN-10"
              value="{{isbn}}"
            />
          </div>
          <div class="col-md-12">
            <input
              class="form-control"
              type="text"
              name="release_date"
              placeholder="Publish year"
              value="{{release date}}"
            />
          </div>
        </form>
      </div>
    </div>
  </section>
</main>

```

```

69 app.post('/register', validate([newUsernameSchema, newPasswordSchema, emailSchema]), async (req: Request, res: Response) => {
70   const { username, password, email } = req.body;
71   console.log(req.body);
72
73   // check if user already exists
74   const userExists = await db.getUser(username);
75   const emailExists = await db.getUserByEmail(email);
76   if (userExists) {
77     res.status(400).render('register.html', { errors: [{msg:'Username already in use'}], username: username, email: email });
78   } else if (emailExists){
79     res.status(400).render('register.html', { errors: [{msg:'Email already in use'}], username: username, email: email });
80   } else {
81     const hashedPassword = password //await bcrypt.hash(password, 10);
82     await db.addUser(username, hashedPassword, email);
83     // ? pop up message saying account created
84     res.redirect('/login');
85   }
86 });
87

```

```

// read a book
app.get('/book/:book_isbn/read', async (req: Request, res: Response) => {
  const book_isbn = Number(req.params.book_isbn);
  const book = await db.getBooks([book_isbn]);
  res.render('read-book.html', book[0]);
});

// get reviews about a book
app.get('/book/:book_isbn/reviews', async (req: Request, res: Response) => {
  const book_isbn = Number(req.params.book_isbn);
  const reviews = await db.getReviews(book_isbn);
  res.send(JSON.stringify(reviews));
});

// post search form
app.post('/search', async (req: Request, res: Response) => {
  const searchTerm = req.body.term;
  if (searchTerm) {
    console.log(searchTerm);
    const isbnns = await db.searchBookByTerm(searchTerm);
    let arr : number[] = [];
    for (let i = 0; i < isbnns.length; i++) {
      arr.push(isbnns[i].book_isbn);
    }
    const books = await db.getBooks(arr);
    res.render('search.html', {term:searchTerm, books: books, username: req.session?.username, admin:req.session?.admin });
  } else {
    res.status(400).send('Bad request');
  }
});

/** Requests */
// book request form
app.get('/request', async (req: Request, res: Response) => {
  if (req.session?.admin) {
    res.status(403).send('Admins cannot request books');
    return;
  }
  if (!req.session?.user_id) {
    res.redirect('/login');
    return;
  }
  res.render('request.html', {username: req.session.username, admin:req.session.admin});
});

```

```

// add a user to the database
async function addUser(username: string, password: string, email: string): Promise<sqlite.ISqlite.RunResult<sqlite3.Statement>> {
  const db = await connectToDB();
  return await db.run(sql`INSERT INTO user (username, password, email) VALUES (${username}, ${password}, ${email})`);
}

// get user by username
async function getUser(username: string): Promise<any> {
  const db = await connectToDB();
  const user = await db.get(sql`SELECT * FROM user WHERE username = ${username}`);
  return user;
}

// get user by id
async function getUserById(user_id: number): Promise<any> {
  const db = await connectToDB();
  const user = await db.get(sql`SELECT * FROM user WHERE user_id = ${user_id}`);
  return user;
}

💡 add to saved books
async function addSavedBook(user_id: number, book_isbn: number): Promise<sqlite.ISqlite.RunResult<sqlite3.Statement>> {
  const db = await connectToDB();
  return await db.run(sql`INSERT INTO saved_book (user_id, book_isbn) VALUES (${user_id}, ${book_isbn})`);
}

// get saved books by user_id
async function getSavedBooks(user_id: number): Promise<Array<any>> {
  const db = await connectToDB();
  const saved_books = await db.all(sql`SELECT book_isbn FROM saved_book WHERE user_id = ${user_id}`);
  return saved_books;
}

// remove saved book
async function removeSavedBook(user_id: number, book_isbn: number): Promise<sqlite.ISqlite.RunResult<sqlite3.Statement>> {
  const db = await connectToDB();
  return await db.run(sql`DELETE FROM saved_book WHERE user_id = ${user_id} AND book_isbn = ${book_isbn}`);
}

// add a book
async function addBook(isbn: number, title: string, admin_id: number, pdfPath: string, authors?: string, description?: string, publisher?: string): Promise<any> {
  const db = await connectToDB();

```

```

1  {% extends "base.html" %} {% block body %}
2  <nav class="navbar navbar-expand-lg py-4">
3      <div class="container-fluid">
4          <a href="/" class="navbar-brand">
5              
10         </a>
11         <button
12             class="navbar-toggler"
13             type="button"
14             data-bs-toggle="collapse"
15             data-bs-target="#navmenu"
16         >
17             <span class="navbar-toggler-icon"></span>
18         </button>
19         <div class="collapse navbar-collapse" id="navbarSupportedContent">
20             <ul class="navbar-nav me-auto mb-2 mb-lg-0">
21                 <li class="nav-item">
22                     <a href="/" class="nav-link">HOME</a>
23                 </li>
24                 <li class="nav-item">
25                     {% if admin %}
26                     <a href="/requests" class="nav-link">REQUESTS</a>
27                     {% else %}
28                     <a href="/bookmarks" class="nav-link">BOOKMARK</a>
29                     {% endif %}
30                 </li>
31                 <li class="nav-item">
32                     {% if admin %}
33                     <a href="/addbook" class="nav-link">ADD BOOK</a>
34                     {% else %}
35                     <a href="/request" class="nav-link">REQUEST</a>
36                     {% endif %}
37                 </li>
38             </ul>
39             <form
40                 class="d-flex me-4 ms-4"

```

### III- Source Code:

Snapshot:

## *Main Scenario*

Function	Scenario
<b>User create an account to read a book and add it as bookmark</b>	<ul style="list-style-type: none"><li>-User enter the website and is confronted with the main page</li><li>-he then click on the top right of the screen on login</li><li>-he click on create a new account and enter his detail and press submit</li><li>-he is transferred back to the main page with his account displayed on the top right of the page</li><li>-he press the search button and enter the name of the book he wants then click enter</li><li>-the website will redirect him to page with all book related to search he entered</li><li>- he then click on read book next to the cover</li><li>-the website will display a pdf file for him to read</li><li>-user exist from the pdf and is returned to website</li><li>-he press the bookmark symbol</li><li>-he press book marked to view all the book he marked</li></ul>
<b>User request a book to add</b>	<ul style="list-style-type: none"><li>-User enter the website and is confronted with the main page</li><li>-he then click on the top right of the screen on login</li><li>-he enter his detail and press submit</li><li>-he is transferred back to the main page with his account displayed on the top right of the page</li><li>-he press request book button</li><li>-he is redirected to request book page</li><li>- He clicks create a new request</li><li>- He is confronted with a page to enter his book details</li></ul>

- -he enter the details and click submit
- -his request is added to request board

#### **Admin add book to website**

- Admin enter the website and is confronted with the main page
- he then click on the top right of the screen on login
  - he enter his detail and press submit
- he is transferred back to the main page with his account displayed on the top right of the page
  - he press request book button
- he is redirected to request book page
  - he click add book button
  - he is confronted with a page to enter his book details and the pdf of the book
    - he enter the details and pdf then click submit
- The book is added to the website and can be viewed by user

#### ***IV- Open source code:***

- <https://codepen.io/onion2k/pen/xxZYBVj>

## 2. Conclusion:

### ***I- Learnings from the project:***

This project mainly focuses on developing an improved online library website that is simple to operate and manages all the Work by digitizing the process as well as introducing additional features that aid the website's users as well as giving quality work a priority in either the frontend or backend.

A relational database is linked to our Web Base application (SQLite).

The front end was coded using JavaScript and related tools, including HTML and CSS.

The backend is supported and connected to the database using express, as well as Node.js, Multer, and Nunjacks.

The library will become more optimized, enabling the development of new features that will enhance the utility of the program in the future.

### ***II- Limitations of your work***

The release version lacks a number of capabilities, including advanced search, reviews, sharing, and view histories. Additionally, there is no automation for adding books to the library, no support for book requests, a one-book-at-a-time restriction, and no genuine user-developer interaction.

### ***III- What would you do differently if you had more time***

Whether it was the addition of books by users or the modification and communication, we would have preferred to create a more focused community-based website. We would have also loved to make it viable for conventional libraries with a borrow and retrieve mechanism as we would like them to participate in the transfer to the digital space.

Additionally, we'd like to make use of several technologies, such as Redis for session storage, MySQL for the database, and PHP for the back



### 3.Worksheet:

Phase	Tasks
<b>Phase one</b> (Oct 05)	Work was done as group by every members
<b>Phase Two</b> (Nov 06-Nov 07)	Work was done as group by every members
<b>Phase Three</b> (Dec 10- Dec 14)	HTML/CSS: Fadel/Tareq JavaScript: Fadel/ Mohammad Backend: Mohammad Report: Adi

### 4.Demo

Link:

[https://drive.google.com/drive/folders/1WZOmnGgRZn3KZHAPahL5KrfMWqaRumsm?usp=share\\_link](https://drive.google.com/drive/folders/1WZOmnGgRZn3KZHAPahL5KrfMWqaRumsm?usp=share_link)