

***Software Engineering Department***

***Braude College***

***Recommendation-Based Digital Library System***

***Capstone Project Phase B***

*Yuval Zohar*

*Amit Perets*

***Supervisor:***

*Dr Naomi Unkelos Shpigel*

Table of Contents

[Abstract 4](#_Toc177724079)

[1. Introduction 4](#_Toc177724080)

[2. BACKGROUND AND RELATED WORK 5](#_Toc177724081)

[2.1 Digital Libraries 5](#_Toc177724082)

[2.1.1 Current Digital Libraries examples 5](#_Toc177724083)

[2.2 Recommendation systems 7](#_Toc177724084)

[2.3 Database Systems 8](#_Toc177724085)

[2.3.1 NoSQL Databases: Characteristics and Use Cases 8](#_Toc177724086)

[2.3.2 Firebase 8](#_Toc177724087)

[2.4 Web technologies 9](#_Toc177724088)

[2.4.1 Front-End Technologies 9](#_Toc177724089)

[2.4.2 Back-End Technologies 9](#_Toc177724090)

[2.4.3 Security Considerations 10](#_Toc177724091)

[2.4.4 Accessibility and Performance Optimization 10](#_Toc177724092)

[3. Engineering Process 10](#_Toc177724093)

[3.1 Process 10](#_Toc177724094)

[3.1.1 Understanding the Issue and Reviewing Existing Solutions 10](#_Toc177724095)

[3.1.2 Requirements Gathering 10](#_Toc177724096)

[3.2 Product 12](#_Toc177724097)

[3.2.1 Diagrams & Architectures 12](#_Toc177724098)

[3.2.2 Algorithms 19](#_Toc177724099)

[3.2.3 DB Structure 19](#_Toc177724100)

[4. User Feedback Questionnaire 22](#_Toc177724101)

[4.1 Questionnaire 22](#_Toc177724102)

[4.2 Questionnaire Results 23](#_Toc177724103)

[5. User guide 23](#_Toc177724104)

[6. Maintenance guide 38](#_Toc177724105)

[6.1 Local installation guide 38](#_Toc177724106)

[6.2 Database 38](#_Toc177724107)

[6.3 System deployment 40](#_Toc177724108)

[6.4 Code architecture 40](#_Toc177724109)

[6.4.1 Client-Side Structure 40](#_Toc177724110)

[6.4.2 Server-Side Structure 41](#_Toc177724111)

[7. Tests 42](#_Toc177724112)

[8. Conclusions 44](#_Toc177724113)

[9. References 44](#_Toc177724114)

[10. Books websites links 45](#_Toc177724115)

[11. GIT 45](#_Toc177724116)

[12. AI Tools 45](#_Toc177724117)

Figures

[Figure 1 - System architecture diagram 12](#_Toc177753265)

[Figure 2 - Use Case Diagram 13](#_Toc177753266)

[Figure 3 - Login/Registration activity diagram 14](#_Toc177753267)

[Figure 4 - Logged in user flow activity diagram 15](#_Toc177753268)

[Figure 5 - Admin activity diagram 16](#_Toc177753269)

[Figure 6 - Recommendation system activity diagram 17](#_Toc177753270)

[Figure 7 - Class Diagram 18](#_Toc177753271)

[Figure 8 - Questionnaire age group distribution 22](#_Toc177753272)

[Figure 9 - Our books screen 23](#_Toc177753273)

[Figure 10 - Logged in users navigation bar 24](#_Toc177753274)

[Figure 11 - Managers panel screen 24](#_Toc177753275)

[Figure 12 - Dropdown menu 24](#_Toc177753276)

[Figure 13 - Unsigned users navigation bar 25](#_Toc177753277)

[Figure 14 - Login screen 25](#_Toc177753278)

[Figure 15 - Reset password screen 26](#_Toc177753279)

[Figure 16 - Registration screen 26](#_Toc177753280)

[Figure 17 - Book details screen 27](#_Toc177753281)

[Figure 18 - Update user details screen 27](#_Toc177753282)

[Figure 19 - Create books request screen 28](#_Toc177753283)

[Figure 20 - Profile screen 28](#_Toc177753284)

[Figure 21 - Add book screen 29](#_Toc177753285)

[Figure 22 - Update / delete book screen 30](#_Toc177753286)

[Figure 23 - All borrow requests screen 30](#_Toc177753287)

[Figure 24 - Approve borrow request screen 31](#_Toc177753288)

[Figure 25 - Create borrow request for user screen 31](#_Toc177753289)

[Figure 26 - Return book / update return date screen 31](#_Toc177753290)

[Figure 27 - Book requests screen 32](#_Toc177753291)

[Figure 28 - Books report screen 32](#_Toc177753292)

[Figure 29 - Users report screen 33](#_Toc177753293)

[Figure 30 - Borrows report screen 33](#_Toc177753294)

[Figure 31 - Book's data screen 34](#_Toc177753295)

[Figure 32 - User's data screen 35](#_Toc177753296)

[Figure 33 - Permissions screen 36](#_Toc177753297)

[Figure 34 - Contact screen 37](#_Toc177753298)

[Figure 35 - DB structure 39](#_Toc177753299)

[Figure 36 - DB structure screenshot 39](#_Toc177753300)

[Figure 37 - Deployment diagram 40](#_Toc177753301)

Tables

[Table 1 - Digital Library Comparison 7](#_heading=h.3dy6vkm)

[Table 2 - Tests 40](#_heading=h.3im3ia3)

# Abstract

Our project introduces a digital library system for Moshav Kanaf, aiming to overcome the limitations caused by its current, outdated system, which is exclusively used by the library staff and not accessible to users. The project aims to develop an inclusive digital library accessible to both users and the administrative team, featuring a recommendation system based on web scraping data from book recommendation websites and employing modern, innovative technologies for an enhanced user experience. We will also implement waiting lists for books to efficiently manage book availability and address user demand. This addition will significantly enhance the user experience by providing a structured approach to accessing high-demand titles and ensuring that the system remains user-focused and efficient. This new system offers a user-friendly platform that simplifies access to digital resources and provides personalized content recommendations. By adopting advanced technology, the updated digital library system will significantly enhance user engagement and satisfaction, advancing Moshav Kanaf's library services into the modern digital age. This project signifies a significant shift towards delivering library services that prioritize accessibility, personalization, and technological innovation, addressing the community's growing needs.

# 1. Introduction

Nowadays, libraries are no longer just physical spaces where books are stored. they have evolved into dynamic, accessible platforms offering a wealth of knowledge and resources. However, this transformation brings its own set of challenges, notably in how users discover and access content that matches their interests and needs[3]. Our project aims to address these challenges by introducing an innovative book recommendation system for our digital library website. Digital libraries face several significant challenges that hold back their effectiveness and user satisfaction. Many digital libraries are still relying on outdated technologies, such as Microsoft Access, which can limit their performance and scalability[4]. Additionally, a notable deficiency in these platforms is their lack of personalized recommendations and adaptations for users through the use of modern algorithms. This gap means users are not receiving customized suggestions that could enhance their discovery and reading experience. Moreover, some of the smaller and local libraries utilize digital systems that are exceedingly difficult to navigate and use. This combination of outdated technology, a lack of personalization, and user-unfriendly systems presents a substantial barrier to accessing and enjoying the wealth of knowledge that digital libraries are meant to offer[4]. Existing solutions for modernizing digital libraries are diverse, with many libraries gradually updating their infrastructure, often falling behind current standards. Basic recommendation features are common but lack deep personalization. A few libraries use AI and machine learning to enhance search and recommendations, but financial limitations hinder widespread adoption, especially for smaller libraries[6]. Moshav Knaf Library is a small, local library that operates using outdated tools and lacks an internet connection. Its catalog and services are still managed manually, limiting access to modern digital resources and personalized features. Despite its valuable role in the community, the library faces challenges in keeping up with technological advancements that could enhance user experience and broaden its reach.

The stakeholders in our digital library enhancement project encompass a diverse range of libraries, from small local libraries with outdated or non-existent digital systems to large, well-resourced libraries that are somewhat modern but not fully leveraging the latest technological advancements, especially in the domain of artificial intelligence within their digital library systems. For small libraries that currently operate with older digital systems or lack a digital presence altogether, our solution offers a significant opportunity for transformation. By implementing our recommendations engine and updating their digital interfaces, these libraries can advance to the leading edge of digital library services. This upgrade will enable them to offer personalized book recommendations to their users, greatly enhancing the user experience and accessibility of their collections. In summary, our project targets the transformation of digital libraries into more dynamic, personalized, and user-friendly platforms.

Our proposed solution aims to enhance the functionality and user experience of digital libraries through the implementation of advanced technology. Given the absence of a large-scale database or API for Hebrew books, we recognized the necessity of alternative approaches to achieve AI-based recommendations system and power our recommendation engine effectively. Specifically, we aim to implement a web scraping system customized to gather data and provide personalized book suggestions through a recommendation engine. This system will analyze individual user interactions and reading preferences, utilizing this data to offer customized recommendations that align with each user's unique interests.

# 2. BACKGROUND AND RELATED WORK

## 2.1 Digital Libraries

In the digital age, digital libraries have become crucial in the world of technology, changing how we access, preserve, and share knowledge. These virtual repositories offer an expansive array of digital resources, including books, journals, images, and multimedia content, accessible from anywhere and at any time. The start of digital libraries is closely linked with the fast progress in computer and internet technologies. The widespread use of mobile devices has increased the possibility for digital libraries to offer universal access to information, overcoming the limits of traditional libraries[3]. This accessibility is driven by advancements in wireless communication technologies, like 4G and Wi-Fi, which allow smooth access to digital collections and make information access more widespread than ever before. At the heart of digital libraries is the integration of sophisticated technologies. These systems merge the latest in software engineering, database technology, and network infrastructure to offer robust solutions for information storage, retrieval, and management. The architecture of digital libraries, often based on the browser-server model, ensures users can access resources through web browsers on various devices, showcasing the system's platform independence and ease of maintenance[3]. Furthermore, the '5S' framework (Streams, Structures, Spaces, Scenarios, Societies) offers a way to understand the complexities of digital libraries, helping create systems that are both technologically advanced and user-focused [4]. Key challenges in digital library development include digital rights management, data preservation, and interface design. However, emerging technologies like cloud computing and AI provide promising solutions to enhance functionality and user experience, making libraries more intuitive and responsive.

### 2.1.1 Current Digital Libraries examples

In today's digital world, there are many different digital library websites, each showing different levels of commitment to modern technology and principles. Some platforms are at the forefront, using the latest technology to improve accessibility and the user experience, while others stick to older methods and don't fully use the newest technological advancements. This difference highlights the ongoing need for innovation in digital libraries, pointing out the importance of regular updates and the integration of new technologies and principles. These efforts are crucial for improving search functions, user interfaces, and access to digital resources, making sure they stay valuable and relevant in a quickly changing digital environment. There are various ways to tackle challenges in these resources, from simple software and hardware updates to adding basic recommendation features. Yet, many libraries still don't fully use artificial intelligence and machine learning to make user experiences more personal, often because of budget and technical limitations, especially in smaller libraries. While there have been advancements in making user interfaces more friendly and increasing digital content, there's still a lot of potential in using modern algorithms for dynamic, personalized interactions that hasn't been fully explored yet. We conducted a comparison between 6 digital library systems across a variety of features and elements within them, in order to highlight the standard structure of contemporary digital libraries:

1. **Israel Digital:** The Israel Digital library offers a rich collection of digital resources, including historical documents, photographs, and literature, focusing on the cultural and historical heritage of Israel.
2. **Global Digital Library:** The Global Digital Library aims to increase the availability of high-quality educational resources worldwide, primarily targeting countries with limited access to educational materials, with a strong focus on supporting children’s literacy in multiple languages.
3. **UNESCO Digital Library:** The UNESCO Digital Library provides access to UNESCO’s vast repository of publications and documents on topics related to education, science, culture, and heritage from around the world, supporting the organization's mission to promote knowledge and understanding.
4. **e-vrit:** e-vrit is a digital bookstore in Israel, offering a wide range of eBooks and digital content in Hebrew, catering to the Israeli reading audience with a variety of genres and titles.
5. **Steimatzky:** Steimatzky is Israel's oldest and largest bookstore chain, providing a vast selection of books, magazines, and multimedia products. It has a significant presence both offline in numerous locations across the country and online, serving the Israeli market with Hebrew and English literature.
6. **Audible:** Audible is a leading global provider of spoken word entertainment and audiobooks, offering a vast library of audiobooks, podcasts, and other audio products. It allows users to purchase and stream audio content from a wide range of genres on various devices.

*Table 1 - Digital Library Comparison*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Library  Features | Israel digital | Global Digital Library | UNESCO Digital Library | e-vrit | steimatzky | audible |
| User-Friendly Interface | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא |
| Advanced Search | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא |
| Mobile Compatibility | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא |
| Recommendations  system | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא |
| Multi-language Support | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם v עם מילוי מלא |
| Technologies | ASP.NET  jQuery | jQuery  WordPress | Cloudflare  Vanilla js | React  core-js | jQuery  core-js  moment js  React | jQuery  core-js  slick js |
| Mobile application | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא |
| Audio books | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם x עם מילוי מלא | תיבת סימון עם v עם מילוי מלא |
| Ebook | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא | תיבת סימון עם v עם מילוי מלא |

* For Audible, some of the books have a partnership with Amazon Kindle, so they provide audiobooks and EBOOKS together.

## 2.2 Recommendation systems

In the evolving landscape of digital libraries, the integration of recommendation systems has emerged as a crucial strategy to enhance user experience and engagement. These systems leverage various algorithms and data processing techniques to analyze user behavior, preferences, and interactions within the library's digital resources. By doing so, they can offer personalized suggestions that significantly reduce the time users spend searching for relevant information and materials. The fundamental concept of such systems is to create a more intuitive and responsive digital environment, where users can discover content that aligns with their interests and research needs more efficiently. The sophistication of recommendation systems in digital libraries has evolved from simple rule-based algorithms to advanced machine learning and deep learning models[6]. These models analyze extensive datasets, including borrowing histories and search queries, to provide more accurate and targeted recommendations[8]. The integration of multimodal data, such as textual, visual, and auditory information, further enhances the recommendation process, offering a comprehensive understanding of user preferences[9]. A critical aspect of these systems is their adaptability and scalability. As digital libraries grow and the diversity of their user base expands, recommendation systems must continuously learn and adjust to new information and changing user behaviors [6]. This dynamic nature ensures that the recommendations remain relevant and valuable over time, enhancing user satisfaction and engagement with the library's resources. Moreover, the implementation of recommendation systems in digital libraries also addresses the challenge of information overload. With the exponential growth of digital content, users can easily become overwhelmed by the expanded array of available resources. Recommendation systems serve as a filter, prioritizing content based on predicted relevance to the user's interests, thereby simplifying the discovery process [8]. This not only improves the efficiency of information retrieval but also increases the likelihood of users encountering unexpected discoveries that might not have been found through traditional search methods. The impact of these systems extends beyond individual user experiences, contributing to the overall improvement of library services. By analyzing the data on user interactions and preferences, libraries can gain insights into trends and gaps in their collections, guiding strategic decisions regarding purchases and resource allocation [5]. This data-driven approach enables libraries to adapt their offerings more closely to the needs and interests of their community, enhancing the library's role as an important resource for research, learning, and personal development. The adoption of recommendation systems in digital libraries enhances user experience by offering personalized navigation through vast content. These advanced algorithms simplify content discovery, making recommendation systems crucial for enriching digital libraries and providing customized content suggestions.

## 2.3 Database Systems

The database world is mainly divided into two specific categories: relational (SQL) and NoSQL databases. Relational databases, established in the 1970s, rely on a structured query language (SQL) for defining and manipulating data, which is organized in a tabular form and follows a strict schema. NoSQL databases, emerging in the late 2000s, are designed to accommodate a wide variety of data models, including key-value, document, wide-column, and graph formats. They offer flexibility in managing large volumes of structured, semi-structured, and unstructured data[7].

### 2.3.1 NoSQL Databases: Characteristics and Use Cases

Conversely, NoSQL databases are distinguished by their scalability and flexibility, designed to spread data across multiple servers or nodes, making them ideal for large-scale, distributed data environments. They are characterized by their lack of a fixed schema, allowing for the easy addition or modification of data structures without significant system interruptions. NoSQL databases are particularly fitted for big data applications, real-time web apps, social networks, e-commerce platforms, and IoT systems, where the data model is expected to evolve[7].

### 2.3.2 Firebase

Advantages: Fully managed, real-time database service with effortless integration with other Google services. Firebase excels in building quick prototypes, offering special solutions for user authentication, hosting, and cloud functions. The real-time synchronization is ideal for applications that require instant data updates.

Disadvantages: While offering ease of use, Firebase can become expensive, and the query capabilities are somewhat limited compared to MongoDB.

Firebase provides a more complete backend-as-a-service (BaaS) solution with features like user authentication, which makes it highly attractive for developers looking to minimize backend development time and focus on the client side.

The choice between MongoDB and Firebase depends on the specific requirements of the project, including the complexity of data handling, scalability needs, and the desire for a managed service with integrated backend capabilities.

## 2.4 Web technologies

Developing a digital library system with a recommendation engine involves a complex interplay of both front-end and back-end web technologies. This section will explore the essential technologies and frameworks that can be utilized to build such a system, distinguishing between front-end and back-end components.

### 2.4.1 Front-End Technologies

The front-end of a digital library involves the user interface and experience aspects—how users interact with the library, browse books, receive recommendations, and view their reading histories. The common technologies:

1. **HTML, CSS, and JavaScript**: The foundation of any web application, these technologies enable the creation of structured content, styled and interactive elements, respectively[1].
2. **Front-End Frameworks (React, Angular, Vue.js)**: These JavaScript frameworks and libraries offer platforms for developing dynamic and responsive user interfaces. They help the development of single-page applications that can update content dynamically without reloading the page, enhancing the user experience[1].

### 2.4.2 Back-End Technologies

The back-end of the digital library handles data management, server interactions, recommendation algorithms, and user authentication. It is crucial for storing user data, book information, and generating personalized recommendations.

1. **Server-Side Languages (PHP, Python, Node.js)**: These languages are used to develop the logic that runs on the server, handling requests from the front end and communicating with the database[1].
2. **Databases (MySQL, MongoDB, PostgreSQL)**: Essential for storing user data, book catalogs, and reading histories. The choice between SQL (e.g., MySQL, PostgreSQL) and NoSQL (e.g., MongoDB) databases depends on the application's data structure and scalability requirements.
3. **Recommendation Engine Algorithms**: Implementing machine learning algorithms (e.g., collaborative filtering, content-based filtering) on the server-side to analyze user behavior and provide personalized book recommendations.
4. **Web Scraping Tools (Beautiful Soup, Scrapy, Puppeter):** These tools are utilized to gather data from various online sources, such as bookstores or literary databases. They facilitate the extraction of information like book titles, authors, descriptions, and reviews, which can enrich the database and enhance the recommendation engine's effectiveness.

### 2.4.3 Security Considerations

Security is extremely important in protecting user data and ensuring trust in the digital library system. Implementing HTTPS, secure coding practices, input validation, and regular security assessments are crucial steps in mitigating potential security issues[1].

### 2.4.4 Accessibility and Performance Optimization

Ensuring the digital library is accessible to all users, including those with disabilities, is essential. This involves following to web accessibility standards and implementing performance optimization techniques such as paging, efficient database queries and more, to enhance the overall user experience[1]. Building a digital library system with a recommendation engine is a complex task requiring a blend of front end and back end technologies. By utilizing the right technologies and focusing on user experience, security, and performance, developers can create a dynamic, efficient, and secure digital library that meets the needs of its users.

# 3. Engineering Process

The process of creating the digital library system for Moshav Kanaf is being carefully planned and executed. We've combined technical expertise with user-friendly design principles while addressing existing technological gaps. This section explains how we’re developing the system, why we choose certain methods, and the challenges we are facing.

## 3.1 Process

### 3.1.1 Understanding the Issue and Reviewing Existing Solutions

In this step, we thoroughly reviewed existing digital library platforms to learn from their successes and identify any areas where Moshav Kanaf's new system could improve. This phase was crucial for understanding the library's challenges and the latest digital library technologies. It helped us discover new features and practices to meet the needs of Moshav Kanaf's users, ensuring the new system would be up-to-date and user-friendly.

### 3.1.2 Requirements Gathering

The first phase included talking to library staff and asking users to fill out surveys to learn about their needs, preferences, and the problems with the current system. This was important to make sure the new system would focus on users and solve the current issues.

#### Interview with the library staff

We visited the library at Moshav Kanaf and conducted an interview with the staff in the presence of our supervisor, Naomi.

#### Questionnaires for users

We conducted a survey using Google Docs and distributed it to residents of Moshav Kanaf, as well as to Facebook groups dedicated to books, friends, family, and general readers. The aim was to gather insights into users' preferences regarding a digital library website and refine the requirements gathering process.

\* The interviews conducted and the associated questionnaires, along with their analyzed results, are thoroughly presented in the capstone project, Phase A.

## 3.2 Product

### 3.2.1 Diagrams & Architectures

#### System architecture diagram

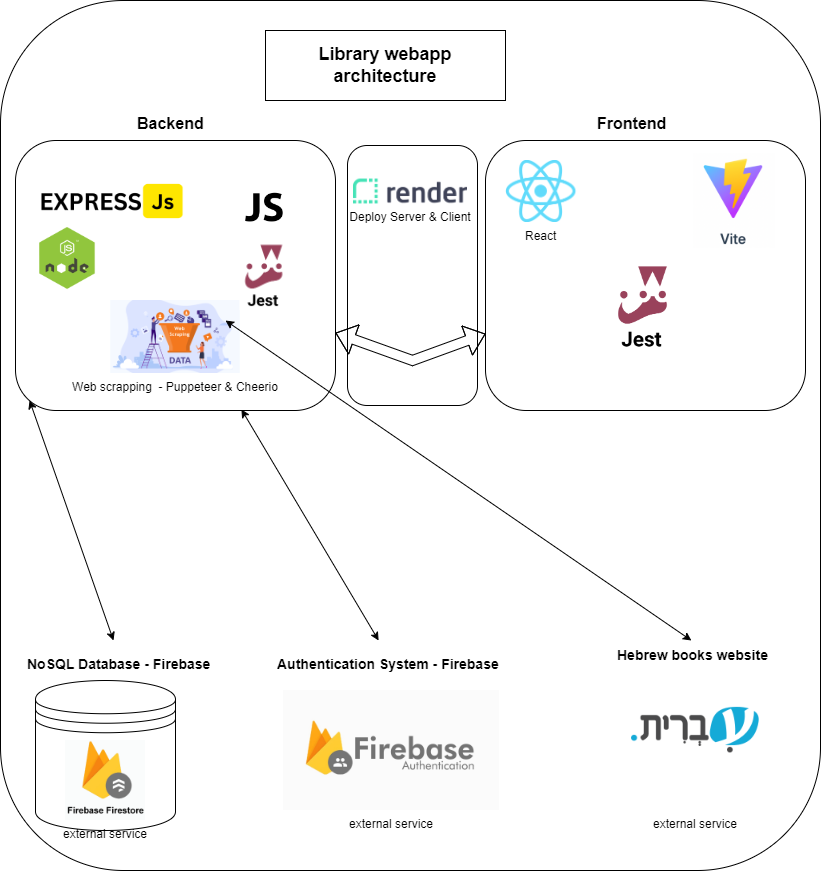


Figure 1 - System architecture diagram

#### Use Case Diagram

The Use Case Diagram offers a graphical overview of the interactions between users and the library system, illustrating the various operations users can perform within the system.



Figure 2 - Use Case Diagram

#### Activity diagram

The Activity Diagram visually represents the flow of activities and actions within the library system, specifically outlining how different users interact with the system, from searching and browsing digital resources to receiving personalized recommendations and managing their user accounts.

*Figure SEQ Figure \\* ARABIC 3 - Activity Diagram*

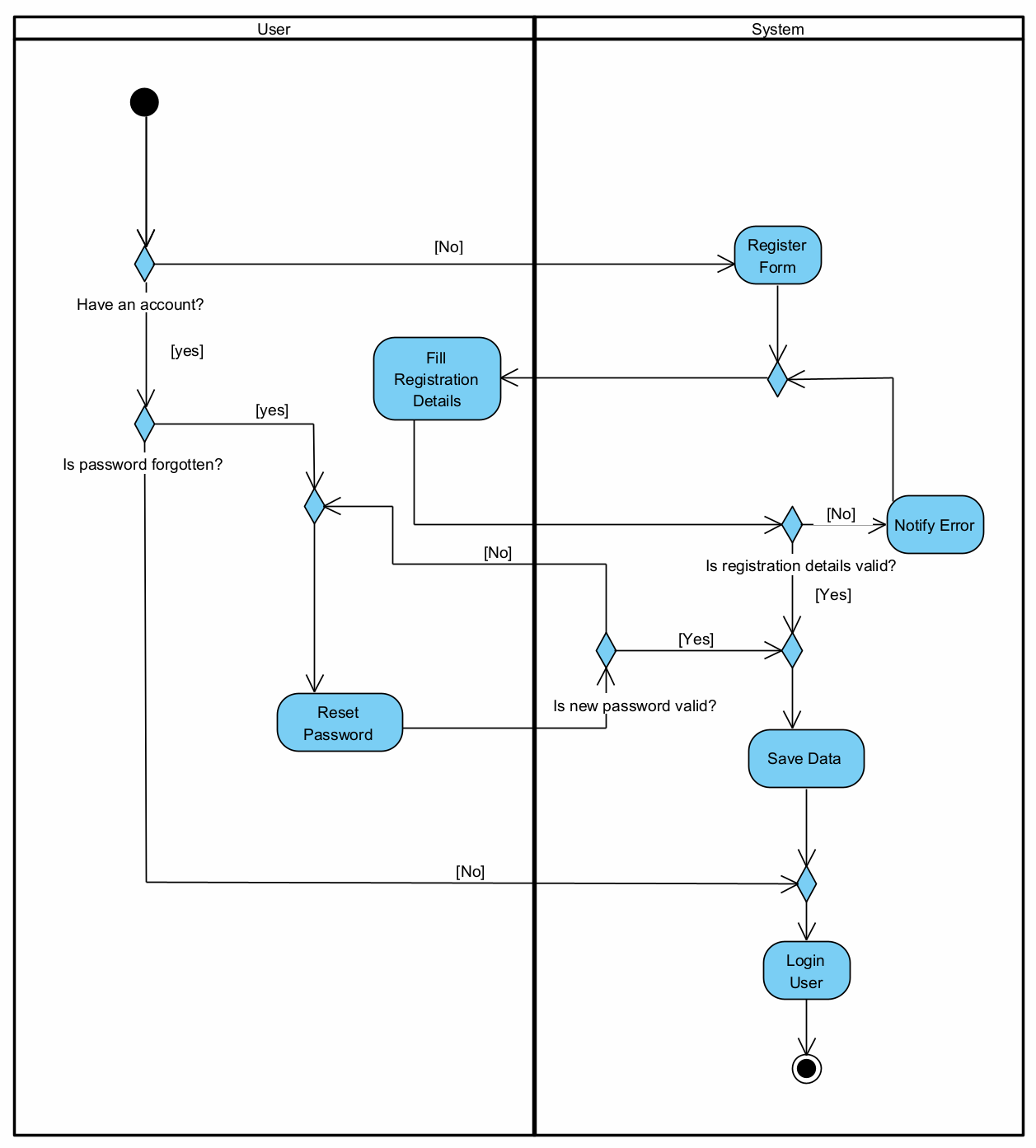
Login/Registration activity diagram:  


Figure 3 - Login/Registration activity diagram

Logged in user flow activity diagram:

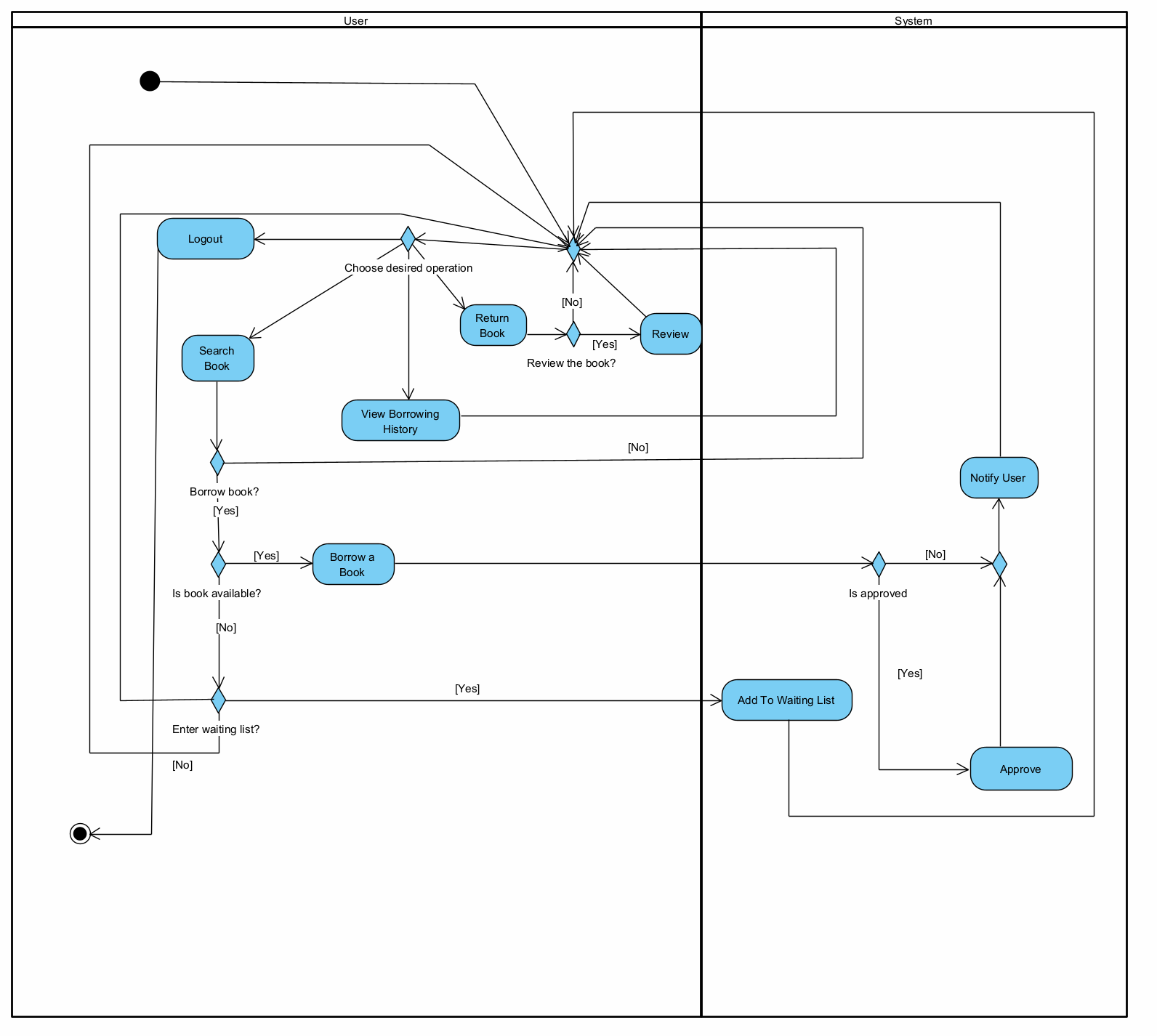


Figure 4 - Logged in user flow activity diagram

Admin activity diagram:

A diagram of a diagram

Description automatically generated

Figure 5 - Admin activity diagram

Recommendation system activity diagram:

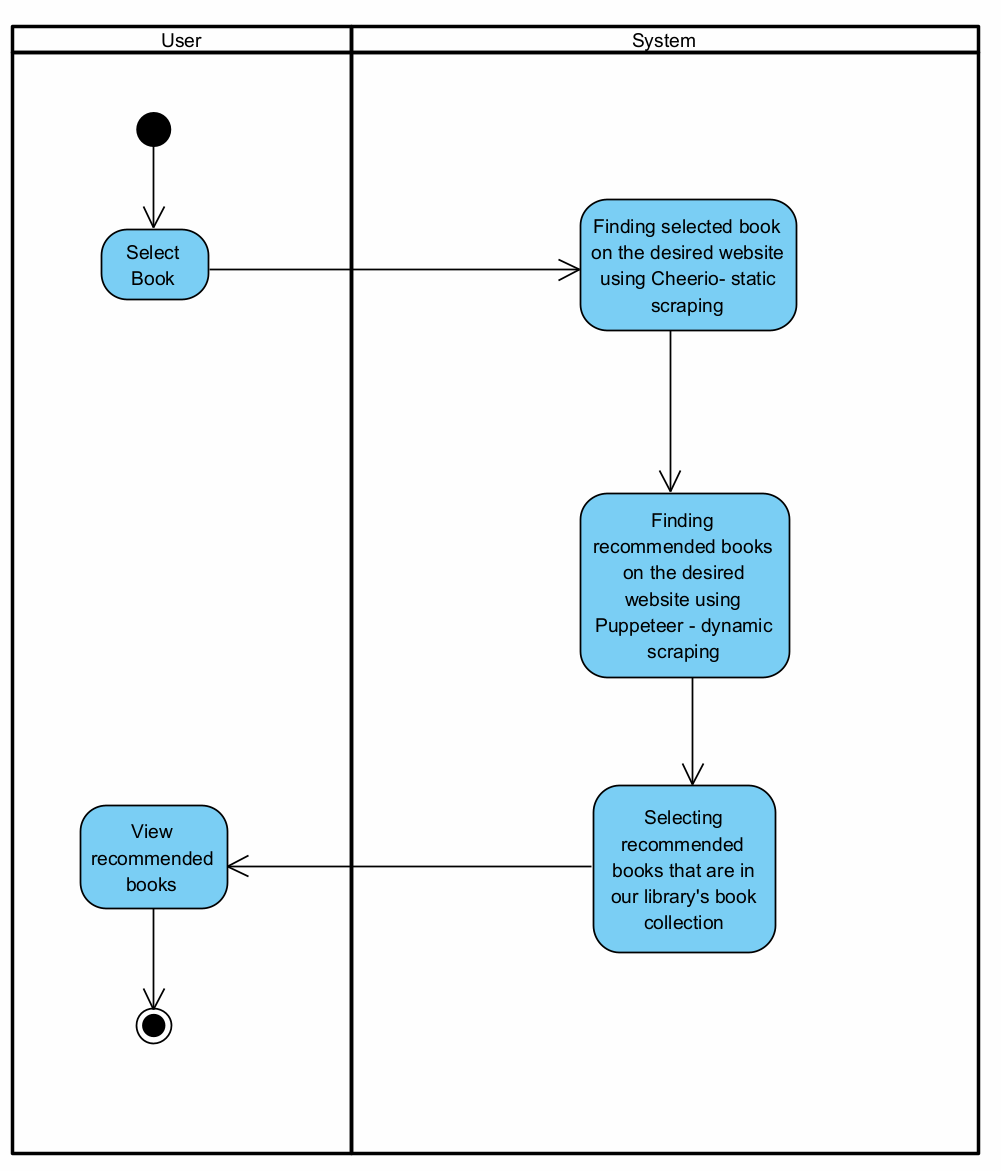


Figure 6 - Recommendation system activity diagram

#### Class Diagram

The Class Diagram offers a detailed structural view of the library system, depicting the system's classes, their attributes, operations, and the relationships among them, which collectively define how the system's functionality is organized and implemented.

*Figure SEQ Figure \\* ARABIC 5 - Class Diagram*

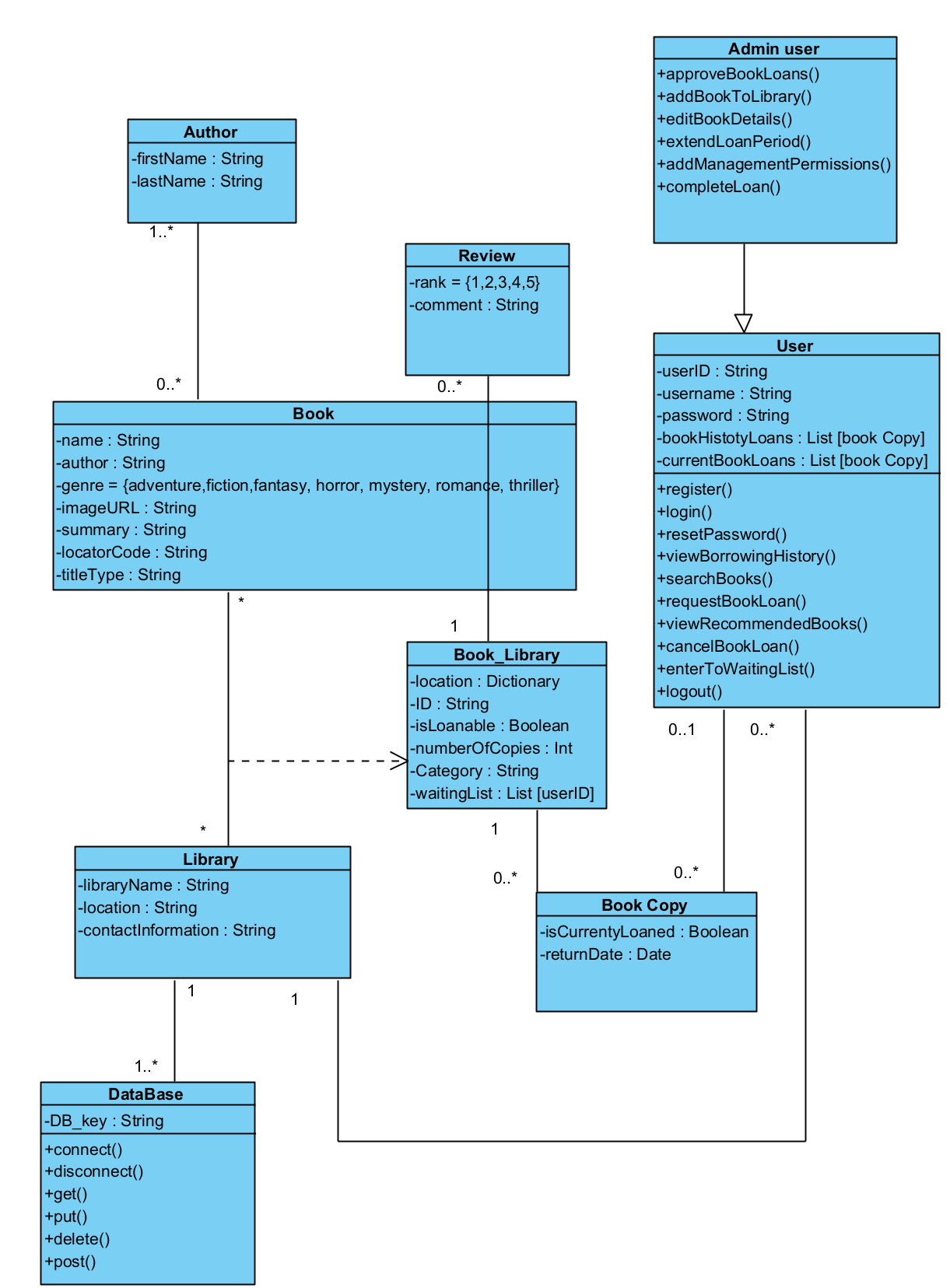


Figure 7 - Class Diagram

### 3.2.2 Algorithms

Web scraping - Web scraping is a technique we use to collect information from websites, and it involves different tools depending on the type of content we need to access. For static websites where the content doesn’t change often, we use a tool called Cheerio[11]. Cheerio is fast and straightforward, allowing us to browse website data efficiently without actually loading it in a web browser. It's perfect for quickly grabbing data that is easily accessible on the webpage. For more complex websites that show different information based on user interactions, like clicking buttons or filling out forms, we use a tool called Puppeteer[12]. Puppeteer acts like a person browsing a site, making it ideal for navigating and extracting data from dynamic websites. It can simulate real user actions to access content that appears in response to these actions. In our project for Moshav Kanaf’s digital library system, we utilize both Cheerio and Puppeteer to interact with Hebrew books websites such as “e-vrit” website, a platform that sells books and recommends similar books. We specifically use Cheerio to handle the static data from the site, which is faster and less resource-intensive. Cheerio let us “search” the book in the extracted data websites. Meanwhile, we deploy Puppeteer for the dynamic aspects of the websites, such as the book recommendation system[See Figure 7, 12]. By searching for a book in the extracted data websites using Puppeteer, we can gather detailed information not only about the searched book but also about other suggested books that the website recommends. This capability allows us to tap into the website's own recommendation system to enhance our book suggestions. Integrating this scraped data into our own library's system, we provide more personalized and diverse book recommendations to our users. This approach enriches our library service, making it more engaging and useful for people seeking new reading material. Using Cheerio for static data and Puppeteer for dynamic data ensures our library system remains cutting-edge, user-friendly, and responsive to the community's needs. By adopting these web scraping tools, our library offers broad access to various books and utilizes the latest technology to assist users in discovering books they might enjoy.

### 3.2.3 DB Structure

Books collection:

{

"author": "string",

"averageRating": "number",

"classification": "string",

"copies": "number",

"copiesID": [

"number"

],

"expenditure": "string",

"imageURL": "string",

"locatorCode": "string",

"ratings": [

{

"ratedAt": "timestamp",

"rating": "number",

"uid": "string"

}

],

"reviews": [

{

"firstName": "string",

"lastName": "string",

"review": "string",

"reviewedAt": "timestamp",

"uid": "string"

}

],

"summary": "string",

"title": "string",

"titleType": "string",

"waitingList": [

{

"Time": "timestamp",

"uid": "string"

}

]

}

Copies collection:

{

"borrowedTo": {

"firstName": "string",

"lastName": "string",

"phone": "string",

"uid": "string"

},

"copyID": "number",

"isBorrowed": "boolean",

"title": "string"

}

Users Collection:

{

"borrowBooksList": {

"title": {

"endDate": "null",

"requestDate": "timestamp",

"startDate": "null",

"status": "string"

}

},

"displayName": "string",

"email": "string",

"firstName": "string",

"historyBooks": [

{

"copyID": "number",

"requestDate": "timestamp",

"returnDate": "timestamp",

"startDate": "timestamp",

"title": "string"

}

],

"isManager": "boolean",

"lastName": "string",

"notifications": [

{

"date": "timestamp",

"isRead": "boolean",

"message": "string"

}

],

"phone": "string",

"random": "number",

"uid": "string"

}

Requests collection:

{

"requestText": "string",

"timestamp": "timestamp",

"uid": "string",

"username": "string"

}

# 4. User Feedback Questionnaire

After developing the system, we conducted a SUS questionnaire along with a few additional questions we created to evaluate the overall structure of the system and gather users' impressions of it.

## 4.1 Questionnaire

The questions included in the SUS questionnaire (the questions were rated on a scale of 1 to 5):

* I would like to use this system frequently.
* I found the system unnecessarily complex.
* I thought the system was easy to use.
* I think I would need assistance from a technical person to be able to use this system.
* I found the various functions in the system well integrated.
* I thought there was too much inconsistency in the system.
* I imagine most people would learn to use this system very quickly.
* I found the system very cumbersome to use.
* I felt confident using the system.
* I needed to learn a lot of things before I could get going with this system.

In addition to the SUS questionnaire, we added a few custom questions to gather more specific feedback and determine if there are any areas that need improvement or additional features. The questions we asked were:

* What elements do you like in the system?
* What elements would you suggest improving in the system?
* Any additional comments regarding the system?

We also collected age group data and contact information in case users wanted us to follow up with them:

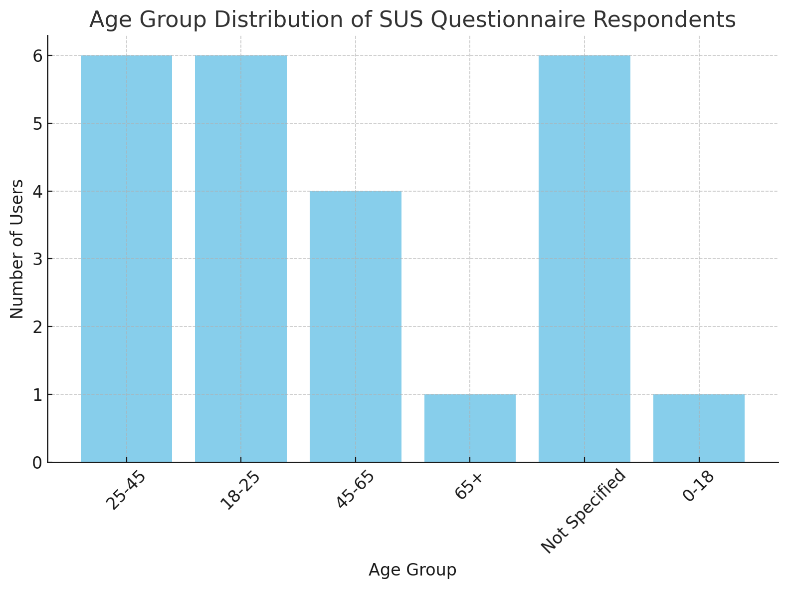


Figure 8 - Questionnaire age group distribution

## 4.2 Questionnaire Results

SUS questionnaire results:

The result we received from the SUS questionnaire was an impressive score of 97.5, indicating an exceptional level of user satisfaction and ease of use. This score suggests that our system is highly intuitive and well-received by users, reflecting the effectiveness of our design and functionality.

Additional questions summary:

* Users appreciated the user-friendly design and intuitive interface.
* The Google account integration and email notifications for available books were well-received.
* The book recommendations and ability to view reading history were also valued.
* Suggestions for improvement included faster loading times for recommendations, brighter website colors, and the option to exclude already-read books from search results.
* Overall, the system was described as professional, practical, and easy to use.

# 5. User guide

The users of the website are the library management team and the readers in Moshav Kanaf. All users are Hebrew speakers. therefore, the user guide has been written in Hebrew according to the requirements.

**מסך ראשי – הספרים שלנו**

מסך זה הוא המסך הראשי באתר.

במסך זה ניתן לראות את כלל הספרים באתר, ע"י מעבר בין הדפים בתחתית המסך.

במסך זה קיימת אופציה "סנן" בה ניתן לבצע סינון  לפי שם סופר או דירוג הספרים.

יתר על כן קיימת אופציית חיפוש, כאשר אפשר לבצע בחיפוש לפי שם הספר ולפי שם הסופר.

מסך זה זמין לצפייה ע"י משתמשים מחוברים ולא מחוברים.

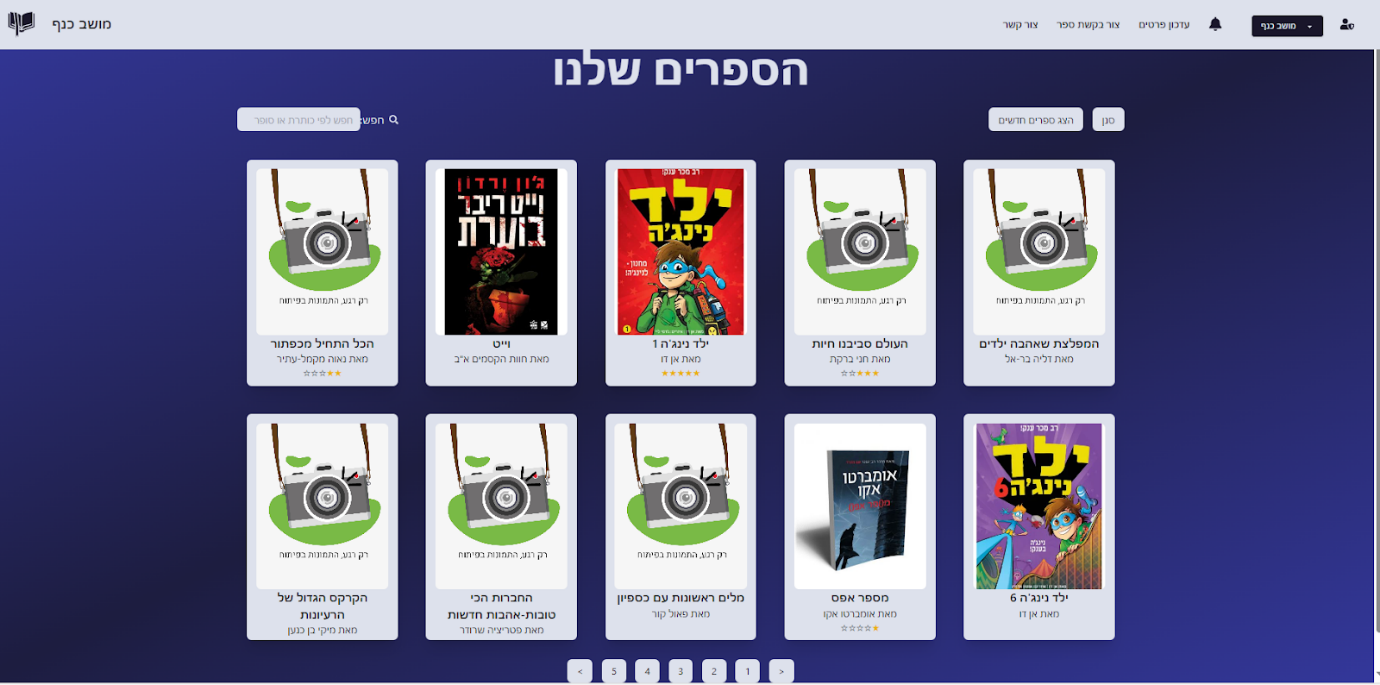


Figure 9 - Our books screen

**סרגל כלים –Navigation bar**

סרגל הכלים משתנה עבור משתמש שמחובר ועבור משתמש שאינו מחובר.

עבור משתמשים מחוברים:



Figure 10 - Logged in users navigation bar

סרגל כלים זה מופיע בכל העמודים בחלק העליון של המסך.

מצד ימין, מופיע סמל של מנהל , אשר מונגש רק למשתמשים שהם Admin (הסבר בעמוד נפרד). לחיצה על סרגל זה תראה כך:

A screenshot of a computer

Description automatically generated

Figure 11 - Managers panel screen

הסבר על כלל האפשרויות הזמינות למנהל יוסברו בהמשך.

ליד, קיים כפתור Dropdown אשר מציג את שם המשתמש ולחיצה עליו תראה כך:

A screenshot of a phone

Description automatically generated

Figure 12 - Dropdown menu

"הפרופיל שלי" ו"מידע בסיסי" – דפים אשר יוסברו בהמשך.

לחיצה על התנתקות תחזיר לעמוד הראשי כאשר המשתמש מנותק.

יתר על כן, קיים אייקון  אשר לחיצה עליו תביא לפתיחת כלל ההתראות של המשתמש.

הכפתורים "עדכון פרטים", "צור בקשת ספר" ו"צור קשר" יביאו לעמודים הרלוונטיים.

עבור משתמשים שאינם מחוברים:



Figure 13 - Unsigned users navigation bar

לחיצה על תפריט פותחת אפשרות של התחברות, בנוסף להתחברות שקיימת בסרגל הכלים עצמו.

לחיצה על "הרשמה" ו"מידע" יובילו לדפים הרלוונטיים, אשר יוצגו בהמשך.

**מסכים למשתמשים שאינם מחוברים:**

**מסך התחברות**

מסך זה יופיע לאחר לחיצה על "התחברות" בתפריט Dropdown בסרגל הכלים.

מסך זה מופיע רק למשתמשים שאינם מחוברים למערכת.

במסך זה ניתן להקיש כתובת מייל וסיסמה למשתמשים שנרשמו באמצעות מייל וסיסמה.

במסך זה ניתן להתחבר בנוסף דרך חשבון הGoogle של המשתמש באמצעות לחיצה על האייקון של גוגל.

יתר על כן, ניתן לעבור למסכים של יצירת חשבון / איפוס סיסמה בעת לחיצה על הכפתורים המתאימים לכך.

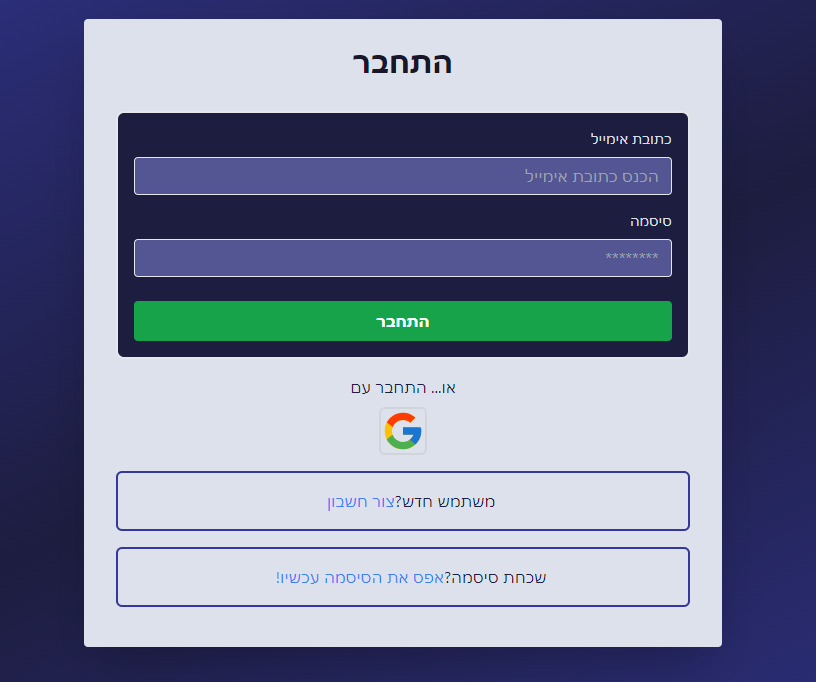


Figure 14 - Login screen

**מסך איפוס סיסמה**

למסך זה נגיע לאחר לחיצה על "אפס את הסיסמה עכשיו!” במסך ההתחברות.

במסך זה ניתן להקיש את כתובת המייל ולקבל הנחיות במייל לאיפוס הסיסמה.

A screenshot of a computer

Description automatically generated

Figure 15 - Reset password screen

**מסך הרשמה**

למסך זה נגיע לאחר לחיצה על "הרשמה" בתפריט Dropdown בסרגל הכלים / לאחר לחיצה על "צור חשבון" בעמוד ההתחברות.

במסך זה נידרש להזין מגוון שדות על מנת להירשם למערכת.

A screenshot of a computer

Description automatically generated

Figure 16 - Registration screen

**מסכים למשתמשים מחוברים:**

**מסך בחירת ספר**

כאשר אנו בעמוד הראשי ולחצנו על ספר מסוים, ספר העמוד ייפתח.

ראשית, התקציר על הספר מוצג, לצד תמונה, שם הספר ושם הסופר.

בתחתית התקציר, קיים כפתור "הזמן עכשיו" אשר יצור בקשת הזמנה לספר עבור המשתמש.

מצד שמאל, קיימים ביקורות משתמשים, אשר המשתמש יכול לכתוב ביקורת על הספר (ניתן גם לסמן אנונימי על מנת שלא יידעו מי כתב את הביקורת).

בתחתית המסך, קיים מקטע של המלצות אשר מגיעות מאתר "עברית" ורלוונטיות לקוראים אשר מתעניינים בספר זה.

A screenshot of a computer

Description automatically generated

Figure 17 - Book details screen

**מסך עדכון פרטים אישיים**

בעת לחיצה על "עדכון פרטים" בסרגל הכלים, מגיעים למסך זה אשר נועד לעדכן את הפרטים האישיים של המשתמש.

במסך זה ניתן לראות את כתובת המייל ואת קוד המשתמש.

יתר על כן, קיימים 3 שדות ("שם פרטי", "שם משפחה", "מס' פלאפון") אשר ניתן לעדכן במידת הצורך.

A screenshot of a login form

Description automatically generated

Figure 18 - Update user details screen

מסך בקשת ספר

**בעת לחיצה על "צור בקשת ספר" בסרגל הכלים נגיע לדף זה.**

**בדף זה ניתן לכתוב בטקסט חופשי בקשה לצוות הספרייה על הבאת ספר חדש לספרייה.**

**A screenshot of a computer

Description automatically generated**

Figure 19 - Create books request screen

**מסך הפרופיל שלי**

בעת לחיצה על "הפרופיל שלי" בתפריט בסרגל הכלים ("עבור משתמשים מחוברים בלבד"), ניתן לראות פרטים כללים כמו שם, אימייל, פלאפון וקוד המשתמש.

יתר על כן, ניתן לראות את כל הספרים המושאלים כרגע.

ספרים בסטטוס "ממתין", אשר ממתינים לאישור הספרנית – ספרים אלו ניתן לבטל ע"י לחיצה על ביטול. בנוסף ניתן לראות את תאריך בקשת הספר.

ספרים בסטטוס "מאושר", אשר אושרו ע"י הספרנית – ניתן לראות את תאריך בקשת הספר, תאריך אישור והתאריך שבו מצופה מהמשתמש להחזיר את הספר.

בנוסף, ניתן לראות את כלל הספרים שנלקחו ע"י המשתמש בעבר, כולל התאריכים שבהם נלקחו.

בדף זה ניתן גם לדרג ספרים שכבר קראת. ניתן לדרג ספר שקראת רק פעם אחת.

A screenshot of a computer

Description automatically generated

Figure 20 - Profile screen

**מסכי מנהל**

בפאנל המנהל אשר הוצג בסרגל הכלים למעלה, ניתן להגיע למגוון מסכים.

**מסך הוספת ספר חדש**

בעת לחיצה על "הוסף ספר חדש" נגיע למסך זה.

במסך זה נצטרך למלא מגוון פרטים על הספר על מנת להוסיף אותו למערכת. בעת לחיצה על "הוסף ספר", הספר יתווסף למאגר הספרים ויהיה זמין לצפייה ע"י הקוראים.

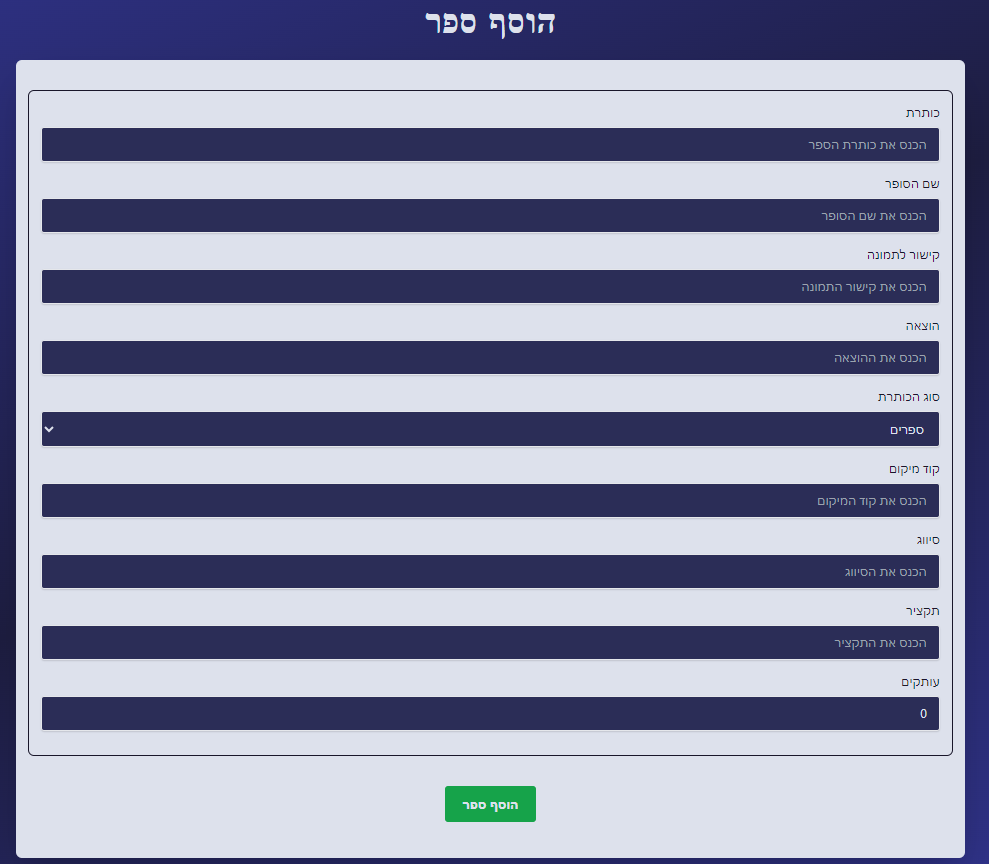


Figure 21 - Add book screen

**מסך עדכון/מחיקת ספר**

במסך זה ניתן לראות את כלל הספרים במערכת ולערוך / למחוק אותם.

בראשית המסך קיים אפשרות חיפוש של ספר לפי שם ספר או שם הסופר.

לחיצה על איקון  תוביל למחיקת הספר מהמאגר.

לחיצה על  תוביל למסך עדכון ספר.

במסך עדכון ספר, יופיעו כלל הפרטים על הספר (כפי שמוצג בהוספת ספר), כאשר נוכל לשנות כל אחד מהפרטים ולאחר מכן ללחוץ "עדכן ספר" על מנת לעדכן את הפרטים החדשים על הספר במאגר.

A screenshot of a computer

Description automatically generated

Figure 22 - Update / delete book screen

**אישור השאלות ספרים**

בעת לחיצה על "אישור השאלות ספרים" נגיע לדף זה.

בדף זה ניתן לראות את כלל הבקשות של המשתמשים אשר ביצעו הזמנת ספר באתר שממתינות לאישור המנהל. במסך זה יש אפשרות לחיפוש עפ"י שם הקורא, מייל ועוד.

A screenshot of a computer

Description automatically generated

Figure 23 - All borrow requests screen

בעת לחיצה על בקשה, נעבור לדף המציג את העותקים הפנויים והתפוסים עבור הספר שהתבקש, בה המנהל יוכל לאשר את ההשאלה במידה והעותק פנוי, ובנוסף לשלוח התראה למשתמש (באתר ובמייל) שהספר מוכן לאיסוף.

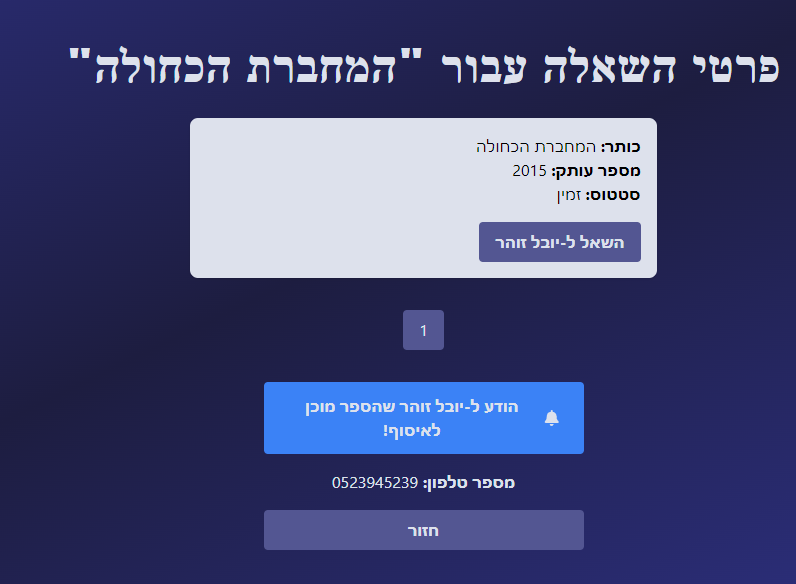


Figure 24 - Approve borrow request screen

**בקשות השאלה עבור משתמש**

בעת לחיצה על "בקשת השאלה עבור משתמש" נגיע לדף זה.

במסך זה נועד עבור כאלה שאינם רוצים או מסתבכים בביצוע פעולת ההזמנה דרך האתר בעצמם. מנהל יכול לבצע הזמנת ספר עבור משתמש, כאשר במסך זה בוחרים את המשתמש עבורו רוצים לבצע הזמנה ואת הספר שרוצים להזמין עבורו.A screenshot of a computer

Description automatically generated

Figure 25 - Create borrow request for user screen

**ניהול החזרת ספרים**

למסך זה נגיע לאחר לחיצה על "ניהול החזרת ספרים".

במסך זה ניתן לראות את כלל העותקים המושאלים ולסמן את החזרת העותק לספריה.

בנוסף, ניתן לעדכן את תאריך ההחזרה עבור משתמש שמעוניין להאריך את תאריך החזרת הספר, ואף לראות מי מהמשתמשים מאחר בהחזרת הספר.

**A screenshot of a computer

Description automatically generated**

Figure 26 - Return book / update return date screen

**מסך בקשות ספרים**

למסך זה נגיע לאחר לחיצה על "בקשות ספרים".

כאשר משתמשים מגישים בקשות ספרים (כפי שפורט לפני כן בעמודי המשתמש שאינו מנהל), בקשות אלו יופיעו בעמוד זה (כולל אפשרות חיפוש של הבקשות לפי שם הקורא). לכל בקשה ישנם 2 כפתורים למחיקת הבקשה והתראה למשתמש.

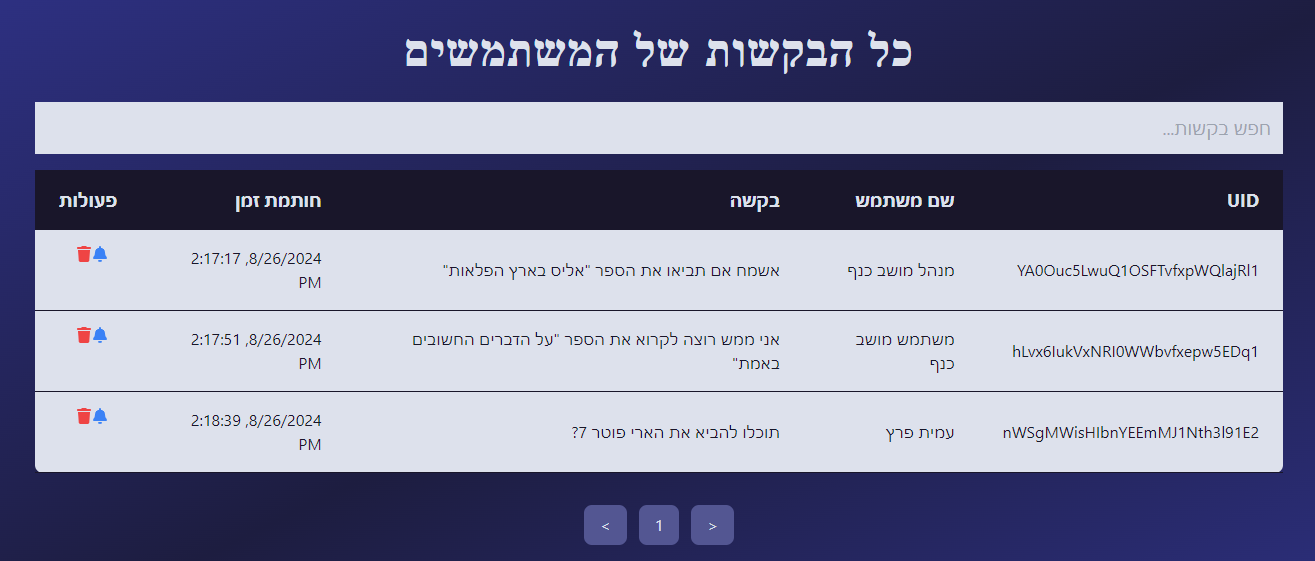


Figure 27 - Book requests screen

במידה ולוחצים על כפתור ההתראה, ניתן לנסח הודעה ולשלוח למשתמשים בין אם הבקשה טופלה או לא.

**מסך דוחות ספרים**

למסך זה נגיע לאחר לחיצה על "דוחות ספרים".

במסך זה נוכל לראות את כלל הספרים במערכת (כולל אפשרות לחיפוש ספר).

בתחתית המסך ישנו כפתור ייצוא לאקסל, אשר ייצא דו"ח זה לקובץ אקסל.

**A screenshot of a computer

Description automatically generated**

Figure 28 - Books report screen

**מסך דוחות משתמשים**

למסך זה נגיע לאחר לחיצה על "דוחות ספרים".

במסך זה נוכל לראות את כלל המשתמשים במערכת ופרטיהם (כולל אפשרות לחיפוש משתמש). בתחתית המסך ישנו כפתור ייצוא לאקסל, אשר ייצא דו"ח זה לקובץ אקסל.

A screenshot of a computer

Description automatically generated

Figure 29 - Users report screen

**מסך דוחות השאלות**

למסך זה נגיע לאחר לחיצה על "דוחות השאלות".

במסך זה נוכל לראות את כלל ההשאלות הפעילות במערכת ופרטיהם (כולל אפשרות לחיפוש משתמש / עותק).

בתחתית המסך ישנו כפתור ייצוא לאקסל, אשר ייצא דו"ח זה לקובץ אקסל.

A screenshot of a computer

Description automatically generated

Figure 30 - Borrows report screen

**מסך נתונים על ספר**

למסך זה נגיע לאחר לחיצה על "נתונים על ספר".

במסך זה נוכל לבחור ספר מסוים ממגוון הספרים הקיימים במאגר ולקבל עליו את המידע הבא: שם הספר, סופר, מס’ עותקים, עותקים זמינים, עותקים מושאלים ורשימת ההמתנה לספר.

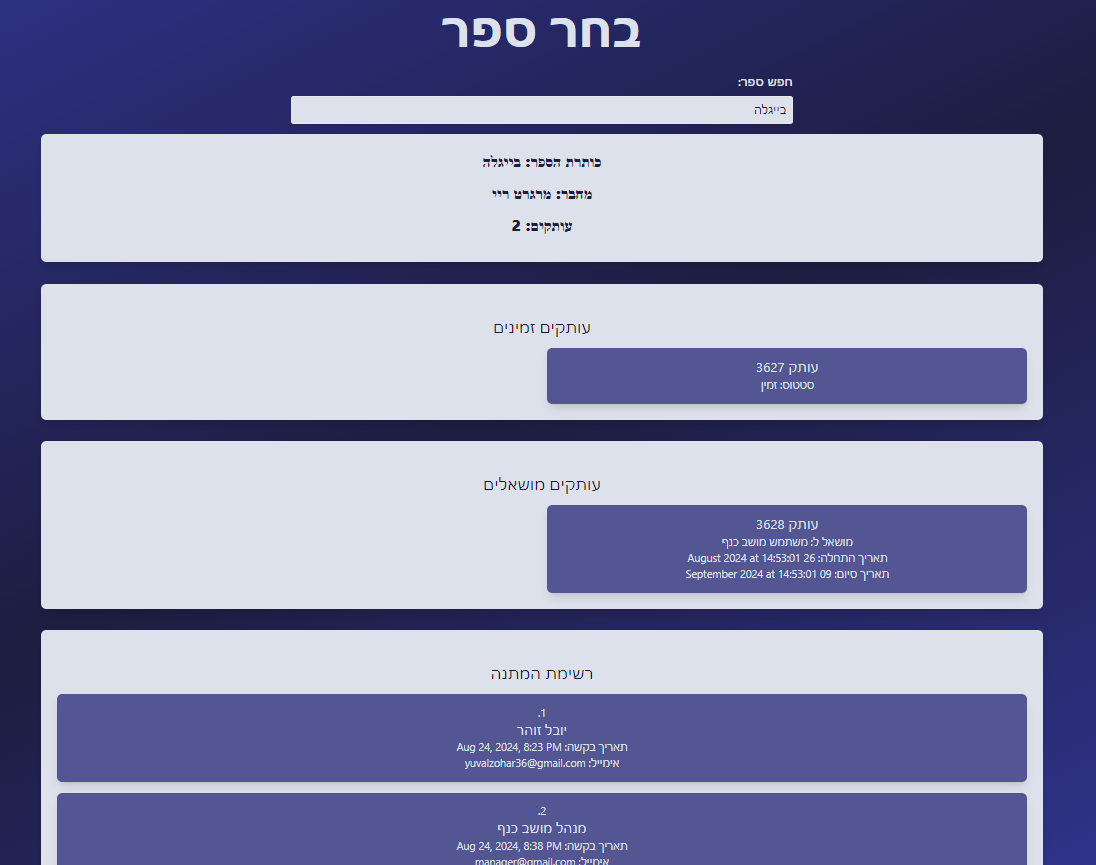


Figure 31 - Book's data screen

**מסך נתונים על משתמש**

למסך זה נגיע לאחר לחיצה על "נתונים על משתמש".

במסך זה נוכל לבחור משתמש מסוים ממגוון המשתמשים הקיימים במאגר ולקבל עליו את המידע הבא: שם מלא, מייל, מס’ פלאפון, קוד משתמש.

בנוסף ניתן לראות את הספרים שהוא ממתין להשאיל, ספרים שמושאלים כרגע על ידי המשתמש, וגם ספרים שהמשתמש קרא בעבר.

A screenshot of a computer

Description automatically generated

Figure 32 - User's data screen

**מסך הרשאות משתמשים**

למסך זה נגיע לאחר לחיצה על "הרשאות משתמשים".

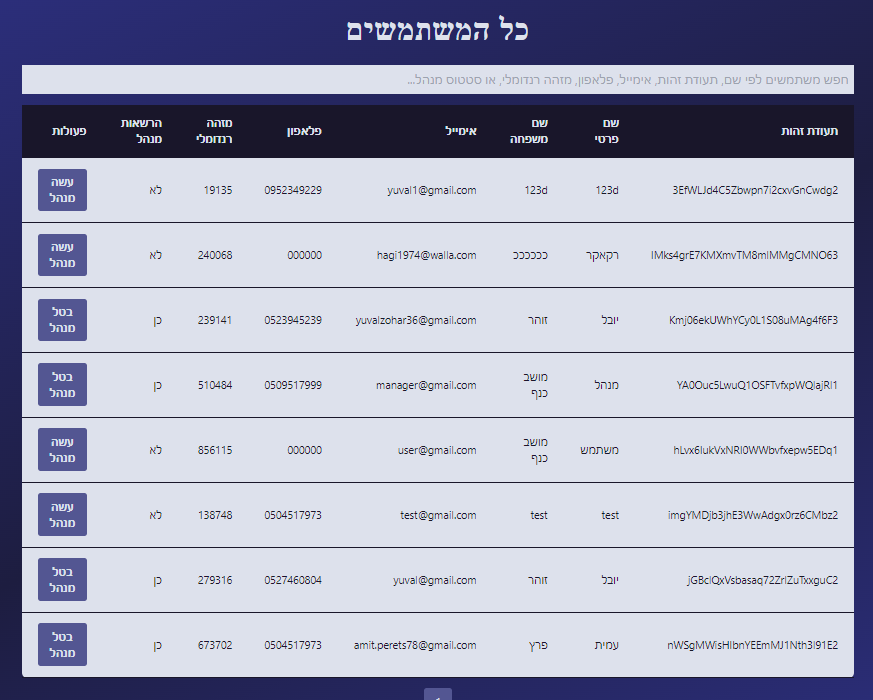
במסך זה נוכל לראות את כלל המשתמשים במערכת (כולל חיפוש משתמש לפי שם או שדות נוספים). במסך זה נוכל להעניק הרשאות מנהל למשתמש שאינו מנהל / להסיר הרשאות מנהל למשתמש בעל הרשאות מנהל.

Figure 33 - Permissions screen

*Figure SEQ Figure \\* ARABIC 32 - Permissions screen*

**מסך צור קשר**

במסך זה ניתן לראות פרטים כללים על הספרייה, כמו פרטים ליצירת קשר, שעות פתיחה ומיקום הספרייה.

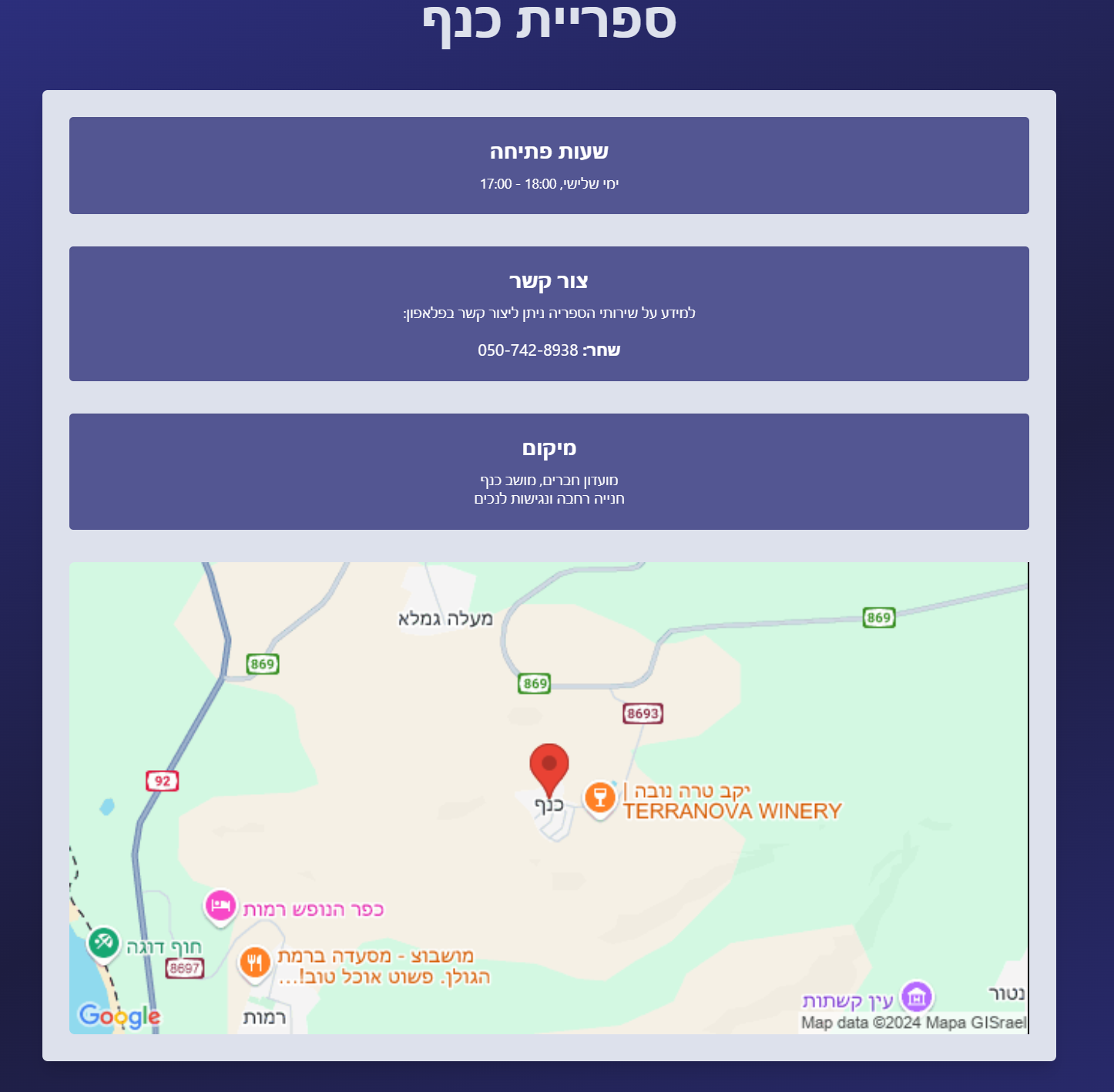


Figure 34 - Contact screen

# 6. Maintenance guide

## 6.1 Local installation guide

To run our website (both client and server) locally, please follow the steps outlined below:

1. **Install npm**:  
   Ensure npm is installed on your PC. We recommend version 10.5.2, although other versions may also be compatible.
2. **Clone the Project Repository**:  
   Clone our project from the GitHub repository using Git Bash or any other Git terminal. Run the following command:

`git clone [https://github.com/LibrarySystemFinalProject/LibMoshavKanaf.git`](https://github.com/LibrarySystemFinalProject/LibMoshavKanaf.git%60)

1. **Install Required npm Packages**:  
   After cloning the repository, install the npm packages required by our project. First, install the server packages by running:

` cd LibMoshavKanaf && cd Server && npm install`

Then, install the client packages by running:

`cd LibMoshavKanaf && cd Client && npm install`

1. **Run the Client and Server**:  
   The client and server must be run separately, both on localhost. Open two terminals:
   * **First Terminal**: Run the server:

` cd LibMoshavKanaf && cd Server && npm run server`

* **Second Terminal**: Run the client:

`cd LibMoshavKanaf && cd Client && npm run dev`

After running the commands in the second terminal, a URL will be displayed for the locally deployed website.

## 6.2 Database

The database outlines a book management system featuring several key collections: books, copies, requests, and users. Each collection holds specific data relevant to different aspects of the system:

* **books**: Contains information about the books themselves, including the author, average rating, summary, list of reviews, and more.
* **copies**: Stores details about the available copies of books, such as whether a copy is currently borrowed, who it is borrowed to, and additional information about each copy.
* **requests**: Tracks user requests for books, including the text of the request, the timestamp, and user information.
* **users**: Includes user data such as borrowing history, current borrow requests, personal details, notifications, and other related information.

This structure supports efficient management and retrieval of data within the system, enabling robust functionality and user interaction.

A screenshot of a computer

Description automatically generated

Figure 35 - DB structure

A screenshot of a computer

Description automatically generated

Figure 36 - DB structure screenshot

System architecture:

See diagram figure [1]

## 6.3 System deployment

Our website is deployed using a client-server architecture:

1. **Client**: The user interface is delivered to users' web browsers, where they interact with the website. The client sends requests to the server and receives data and updates in response. The uses can also access the website through mobile devices. Deployed on Render.
2. **Server**: The backend, also hosted on **Render**, handles client requests, processes data, and runs the web scraping module to gather book information from external sources ("evrit” website).
3. **Deployment on Render**: The server is deployed on Render, ensuring scalability, availability, and secure management of the server environment. This setup allows the server to effectively communicate with both the client and external services for web scraping.

This deployment strategy supports a robust and responsive application for managing book orders and loans.

A screenshot of a computer

Description automatically generated

Figure 37 - Deployment diagram

## 6.4 Code architecture

### 6.4.1 Client-Side Structure

* **dist/** - This directory contains the compiled files generated after the build process. These files are optimized and ready for deployment.
* **node\_modules/** - This folder stores all the dependencies of the project, installed via npm or yarn.
* **public/** - Contains static files like index.html that are served directly from the server without any further processing.
* **src/**
  + **api/** - Contains files related to API communication, such as fetching data from the server using tools like fetch or axios.
  + **assets/** - Stores static assets such as images, fonts, and SVGs used across the project.
  + **components/** - This directory holds reusable React components that are used throughout the application.A screenshot of a computer

    Description automatically generated

*Figure SEQ Figure \\* ARABIC 36 - Client code structure*

* + **contexts/** - Contains React Context definitions for managing the global state of the application.
  + **hooks/** - Custom hooks that encapsulate reusable logic are placed here, promoting code reuse and cleanliness.
  + **pages/** - This directory contains the main page components, with each page representing a unique view or screen in the application.
  + **App.jsx** - The main application file that sets up the overall structure, including routes and core components of the app.
  + **index.js** - The entry point of the React application, where the main render call is made, attaching the React app to the root element in the HTML.

### 6.4.2 Server-Side Structure

* **node\_modules/** - Contains all the dependencies required by the server, installed via npm or yarn.
* **.env** - A file that stores environment variables, such as API keys, database connection strings, and other configuration settings. This file is usually excluded from version control for security reasons.A screen shot of a computer

  Description automatically generated

*Figure SEQ Figure \\* ARABIC 37 - Server code structure*

* **db.js** - The file responsible for setting up and managing the database connection. It likely includes the configuration and connection logic for interacting with the database.
* **package.json** - The manifest file for the Node.js project, detailing scripts, dependencies, and metadata about the project.
* **server.js** - The main entry point for the server-side application. It likely sets up the Express server, configures middleware, and defines routes for handling incoming requests.
* **add\_books\_with\_image.mjs:**This script was created to make the website more accessible and user-friendly by ensuring that all books in the Firebase database have corresponding images and summaries. The script automates the process of updating the database by scraping data from the Steimatzky website. It reads book data from an Excel file, searches for matching book information on Steimatzky's site, retrieves images and summaries, and then updates or adds the book records in Firestore. If an image or summary isn't found, it uses a placeholder image.

#### Recommendations using web scrapping

We developed a script to fetch book recommendations for a specific book listed on the E-Vrit website. The code we used accomplishes this by navigating to the product page of the specified book, identifying the "More Recommendations" section, and extracting the titles of recommended books.

The fetchTitles function is an asynchronous script that uses Puppeteer, a headless browser automation tool, to visit the product page on E-Vrit. Once on the page, the function waits for the "More Recommendations" section to load fully. After ensuring the content is ready, it uses Cheerio, a lightweight HTML parsing library, to extract the titles of the recommended books.

The extracted titles are then logged and returned by the function. This code allows us to efficiently gather and display relevant book recommendations directly from E-Vrit, enhancing the user experience by offering additional book suggestions related to the one being viewed.

For further explanations, see figure [6]

# 7. Tests

Testing is crucial for ensuring the functionality and reliability of our library's website. At every stage, from coding to final implementation, thorough testing is required for all components, including user login, signup, admin access, book addition, ordering, and updates. In addition to testing functionalities, we will implement error checking, such as incorrect password scenarios and data integrity checks after a sudden restart. To enhance our testing framework, we will use Jest [10], a powerful JavaScript testing tool, allowing us to automate tests and simulate user interactions, ensuring the website operates smoothly and provides a seamless user experience.

*Table 2 - Tests*

|  |  |  |
| --- | --- | --- |
| **Expected result** | **Test description** | **Test** **name** |
| The user will get to homepage. | Email : “user@gmail.com”  Password:"123456"  Press: "Login". | User login successfully |
| The manager will get to manager’s homepage. | Email : “manager@gmail.com”  Password:"123456"  Press: "Login". | Manager login successfully |
| error message: “Invalid email or password” is displayed on the screen. | Email : “user@gmail.com”  Password:"0000"- Wrong Password  Press: "Login". | User fails to login |
| error message: "User is already logged in" is displayed on the screen. | Email : “user@gmail.com”  Password:" 123456"  Press: "Login".  Opening the app in different browser and trying to login again with the same user | Login user that already logged |
| Move to “more Information” page. | click the "signup" button and write:  Email : “user@gmail.com”  Password:" 123456"  Confirm password: "123456"  Press: ”Create account” | Sign Up new user |
| error message: "Password should be at least 6 characters" is displayed on the screen. | click the "signup" button and write:  Email : “user@gmail.com”  Password:" 12345"  Confirm password: "12345" – 5 characters  Press: ”Create account” | Sign Up short password |
| error message: "Password and confirm password do not match" is displayed on the screen. | click the "signup" button and write:  Email : “user@gmail.com”  Password:" 1234566"  Confirm password: "1234577"  Press: ”Create account” | Sign Up 2 different passwords |
| error message: "missing password” is displayed on the screen. | click the "signup" button and write:  Email : “user@gmail.com”  Password:""  Confirm password: ""  Press: ”Create account” | Sign up with missing fields |
| Success message: "The book was added successfully ” is displayed on the screen. | Sign in to Manager and click the "Add new book" button.  Title: “The Catcher in the Rye”  Author's Name:” J.D. Salinger  Copies: 2  Copy ID1: 123  Copy ID2: 124 | Add new book successfully |
| error message: “Title, author, and copy ID are required” is displayed on the screen. | Sign in to Manager account and click the "Add new book" button.  Title: “”  Author's Name:” J.D. Salinger  Copies: 1  Copy ID1: 123 | Add a book with missing fields |
| Success message: "The book was updated successfully ” is displayed on the screen. | Sign in to Manager account and click the "Update a book" button.  Title: “The Catcher in the Rye”  Author's Name:” J.D. Salinger  Copies: 2  Copy ID1: 123  Copy ID2: 222 | Update a book successfully |
| Success message: "The book was ordered successfully ” is displayed on the screen. | Sign in to user account, chose a book and click the "Order" button when the book is available | Order a book |

# 8. Conclusions

The modern library goes beyond being just places where books are kept. They have become dynamic centers of knowledge and resources. But this shift comes with its own challenges, especially in helping people find and access the content that suits their interests and needs. Our project tackles these issues by introducing a new book recommendation system for our digital library website. Digital libraries face big obstacles that stop them from working well. Many still use old technologies like Microsoft Access, which can hold back how well they work and how much they can grow. Also, they often don't give personalized recommendations, so users don't get suggestions that would make their reading experience better. On top of that, some libraries have systems that are hard to use, making it tough for people to get to the knowledge the libraries have. Our idea is to make digital libraries much better by using advanced technology. Since there is no huge database or special system for Hebrew books for training AI models, we have come up with a different plan to use AI for making book suggestions. We'll use a special system to collect data and make recommendations based on what each person likes to read. Our project helps all kinds of libraries, from small ones with old systems to bigger ones that need to catch up with the latest technology. By using our recommendation system and updating their digital systems, they can give users a better experience and make it easier for them to find books they like. In short, our project aims to make digital libraries more modern, personal, and easy to use, so they can stay important in the digital age.

# 9. References

[1] Abdurakhimovich, U. A. (2023). The Vital Role of Web Programming in the Digital Age. Journal of Science-Innovative Research in Uzbekistan, 1(6), 42-51.‏

[2] Díaz-Oreiro, I., López, G., Quesada, L., & Guerrero, L. (2019). Standardized questionnaires for user experience evaluation: A systematic literature review. *UCAml 2019*, 14.‏

[3] Fanqi, Wei & Yan, Zhang & Xiaoping, Feng. (2017). Design and Implementation of Digital Library. 10.2991/icemc-17.2017.249.

[4] Fox, E. A., & Ingram, W. A. (2020, August). Introduction to digital libraries. In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020 (pp. 567-568).‏

[5] Ko, H., Lee, S., Park, Y., & Choi, A. (2022). A survey of recommendation systems: recommendation models, techniques, and application fields. *Electronics*, *11*(1), 141.‏

[6] Liu, M. (2022). Personalized Recommendation System Design for Library Resources through Deep Belief Networks. *Mobile Information Systems*, *2022*

[7] Mihai, G. (2020). Comparison between relational and NoSQL databases. *Econ. Appl. Inform*, *3*, 38-42.‏

[8] Tao, H., Niu, X., Fu, L., Yuan, S., Wang, X., Zhang, J., & Hu, Y. (2022). DeepRS: A Library of Recommendation Algorithms Based on Deep Learning. *International Journal of Computational Intelligence Systems*, *15*(1), 45.

[9] Zhang, Q., Lu, J., & Jin, Y. (2021). Artificial intelligence in recommender systems. Complex & Intelligent Systems, 7, 439-457.‏

[10] <https://jestjs.io/>

[11] <https://cheerio.js.org/>

[12] <https://pptr.dev/>

# 10. Books websites links

[Israel Digital (libraryreserve.com)](https://telaviv.libraryreserve.com/10/50/en/Default.htm)

[Home - Global Digital Library - Global Digital Library](https://digitallibrary.io/)

[Home - UNESCO Digital Library](https://unesdoc.unesco.org/)

[e-vrit](https://www.e-vrit.co.il/)

[steimazky](https://www.steimatzky.co.il/)

[audible](https://www.audible.com/)

# 11. GIT

[LibrarySystemFinalProject/LibMoshavKanaf (github.com)](https://github.com/LibrarySystemFinalProject/LibMoshavKanaf)‏

# 12. AI Tools

The AI language models that we used are: [Claude](https://claude.ai/) , [ChatGPT](https://chatgpt.com/)