WEB LIBRARY DOCUMENTATION

SOFTWARE REQUIREMENTS SPECIFICATION

1.0 INTRODUCTION

1.1 Purpose

Allow people from all over the world to sell and buy books for free.

1.2 Scope

1) Creating WEB application.
2) Creating database for storing data and convenient administration.

1.3 Definitions, Acronyms and Abbreviations

1) User: Actor(not logged in) which visits the website and does not have access to all functionality of the application.

2) Customer: The logged in user that has full access to all functionalities of the application.

3) Administrator: Special employee with the role "admin" who has possibility to ban offers on the website

4) ByPol: The name of the application.

1.4 References

1)IEEE Standard 830-1998: IEEE Recommended Practice for Software Requirements Specifications, IEEE Computer Society, 1998

2.0 OVERALL DESCRIPTION

Overview

The ByPol application is an open online platform designed for book enthusiasts to buy and sell books easily. It provides a user-friendly environment where both sellers and buyers can interact and transact efficiently. The key feature of ByPol is its openness and accessibility, allowing any registered user to participate in the marketplace by listing their books for sale or finding the books they need.

Features

User Registration and Profile Management

Registration: Users can register on the platform by providing their details, creating a profile that includes their contact information.

Profile Management: Users can manage their profiles, including updating contact information and other personal details.

Listing Books for Sale

Add Book: Registered users can list books for sale by providing necessary details such as book photos, description, price, and contact information.

Edit and Delete Listings: Users can modify or remove their book listings at any time.

Searching and Viewing Books
Browse Books: Users can view the complete list of available books on the platform.
Search Functionality: Users can search for specific books using various filters and keywords.
Sorting Options: Users can sort book listings based on different criteria such as price, date added, etc.

Favorite Books
Add to Favorites: Users can add books to their favorites list for easy access and future reference.
Remove from Favorites: Users can remove books from their favorites list at any time.

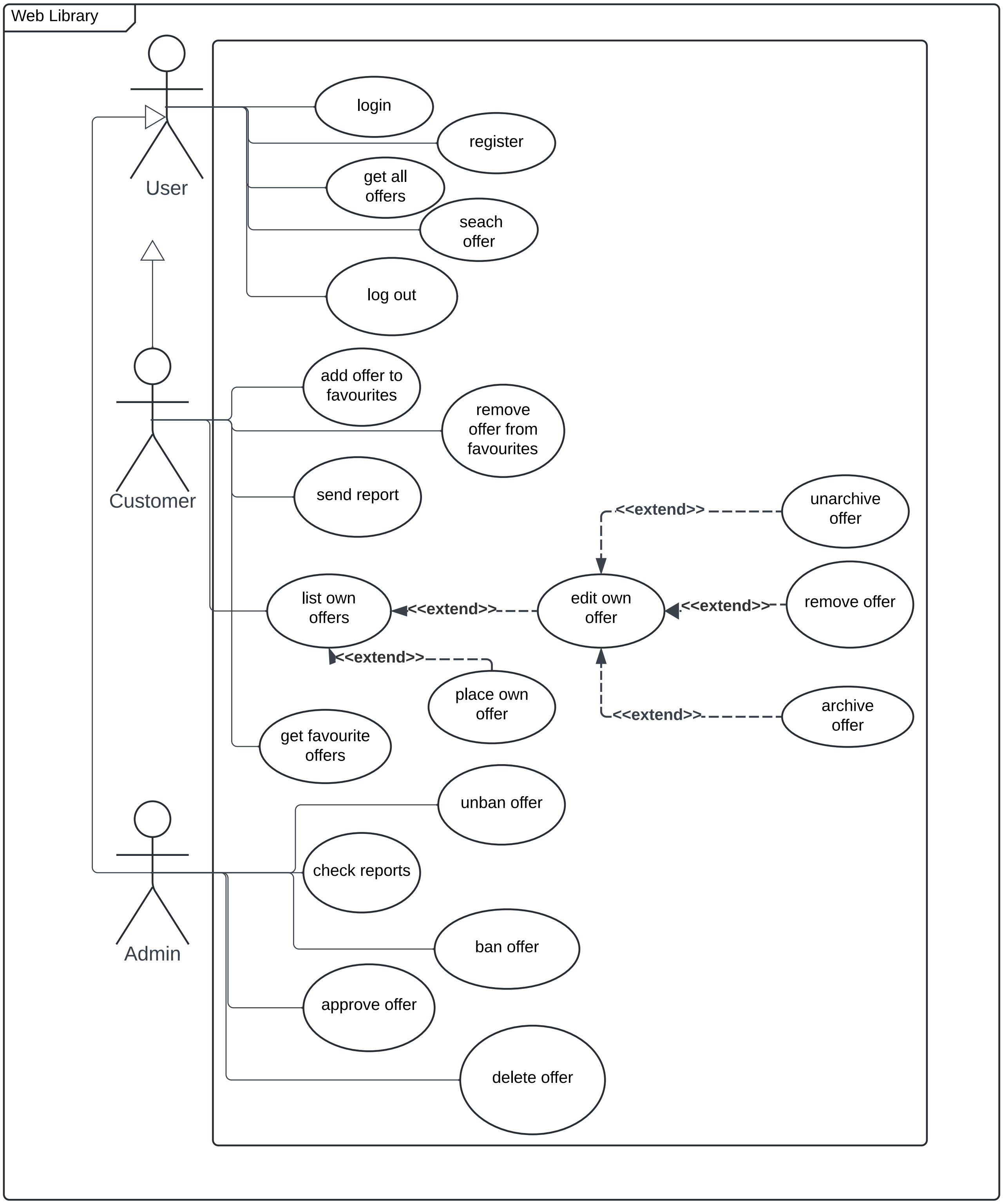
Complaint Function
Submit Complaint: Users can submit complaints regarding an offer if it involves prohibited literature or inappropriate content. Each offer has a "complain" button for this purpose.
Review and Action by Administrator: Administrators review the complaints and, if found valid, delete the offending offer from the platform.

Administrator Role
Review Complaints: Administrators are responsible for reviewing complaints submitted by users.
Delete Offers: If a complaint is validated, the administrator deletes the inappropriate offer from the platform.
Manage Platform: Administrators oversee the platform's overall functionality, ensuring a safe and user-friendly environment.

Conclusion

ByPol is a comprehensive platform that caters to both buyers and sellers, providing essential functionalities to facilitate smooth transactions and maintain a safe marketplace. Its user-friendly features and robust complaint handling system ensure a reliable and enjoyable experience for all user

3.0 USE CASES



3.0.1 *List own offers*

Use Case name	List own offers

Actor	Customer
Purpose	Get the list of own offers
Trigger	Click “My Offers” button
Assumption and pre-conditions	User is logged in
Basic path	1)System displays page with all customer's offers 2)Customer clicks the offer of interest 3)Go to 'edit own offer use case'
Alternative flow of events	Customer clicks "Create Offer" 2a Go to place offer use case
Post condition	User gets the list of his offers

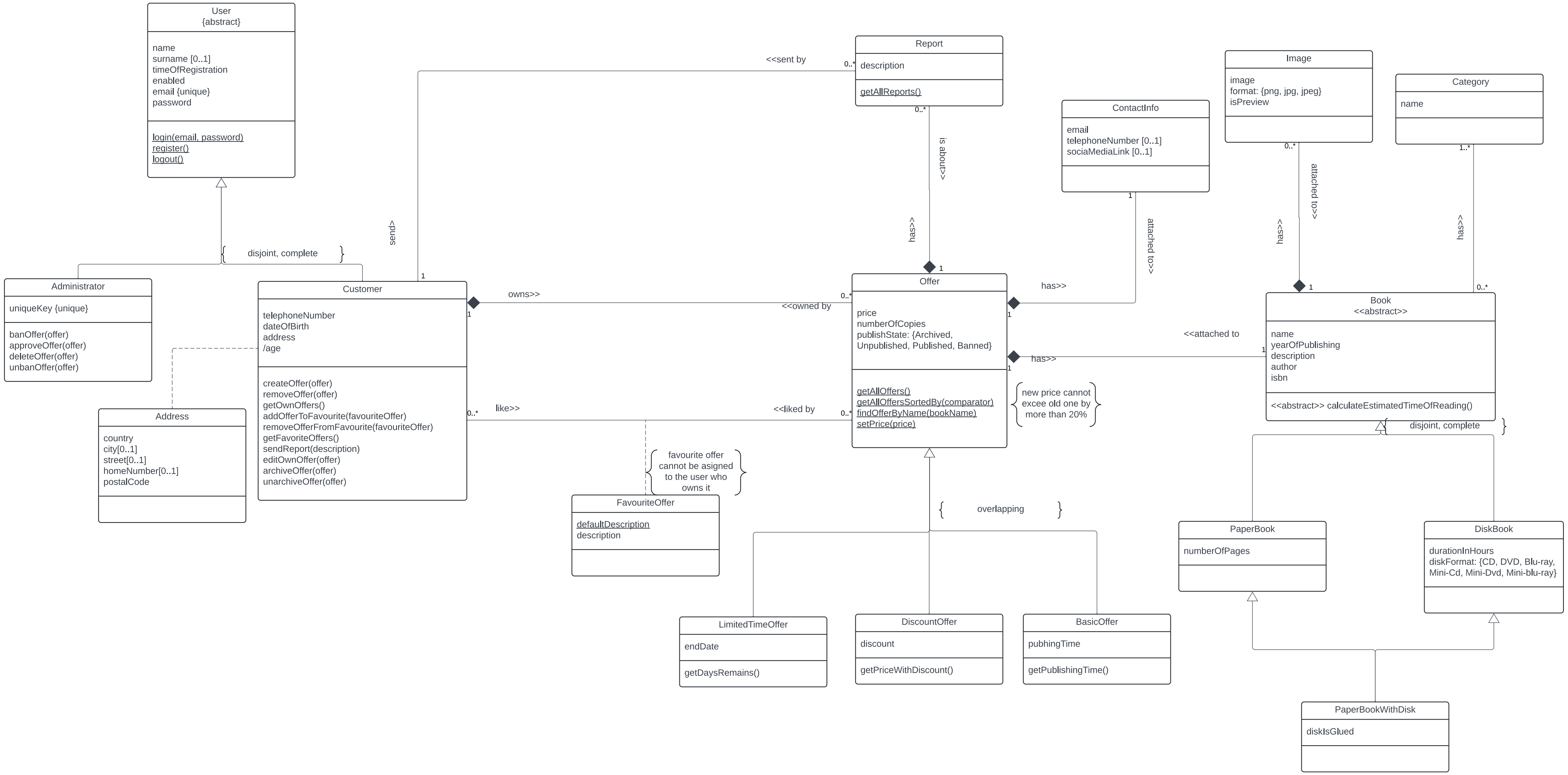
3.0.2 *Archive own offer*

Use Case name	Archive own offer
Actor	Customer
Dependencies	Edit own offer
Purpose	Archive own offer
Trigger	Click 'archive' button
Assumption and pre-conditions	User is logged in
Basic path	1)System asks for confirmation 2)Customer confirms 3)System changes offer's state to archived 4)System returns user to 'edit own offer' page
Alternative flow of events	Customer doesn't confirm 2a1 Go to “Edit own offer” use case
Post condition	User gets the list of his offers

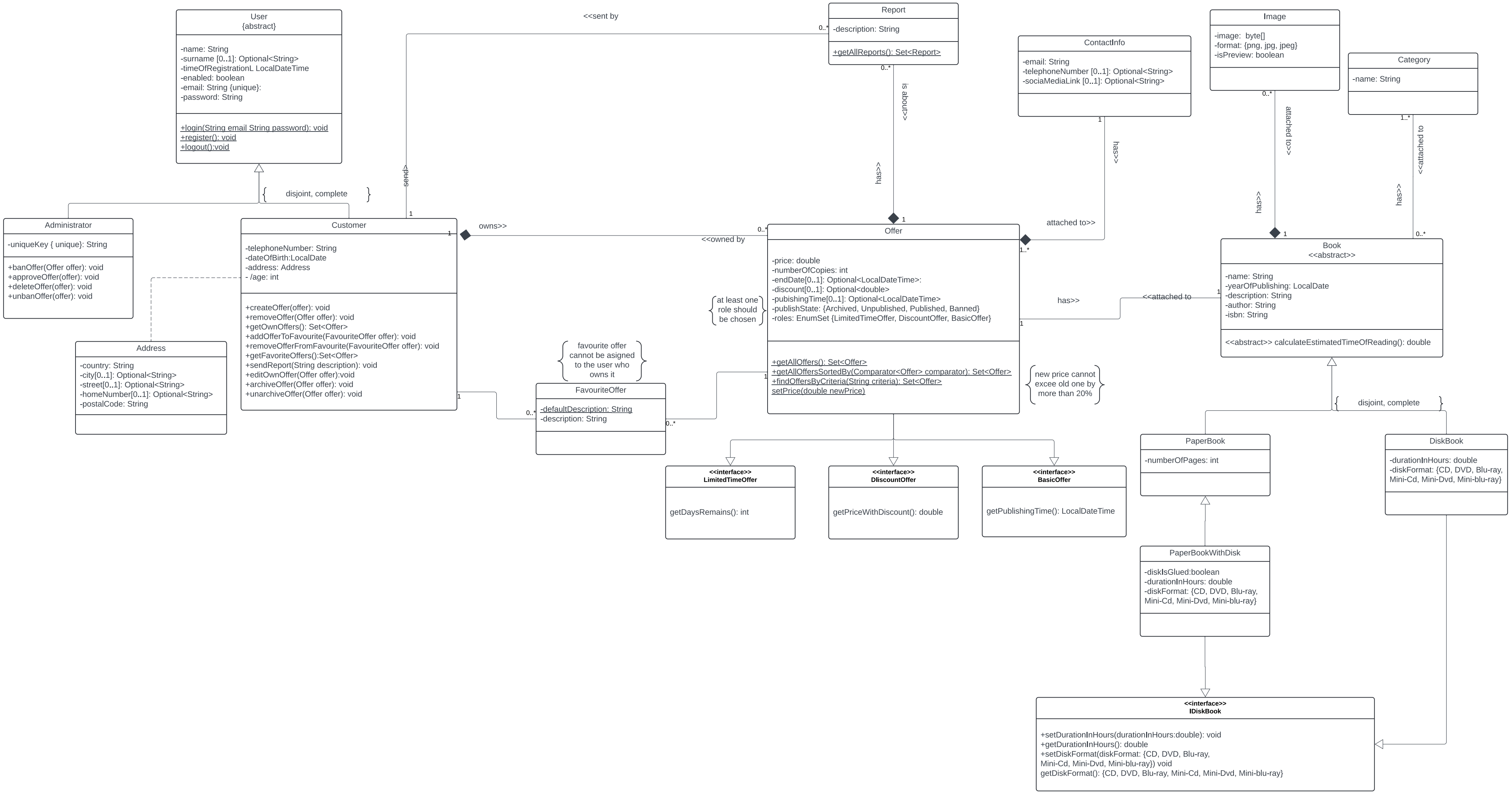
3.0.3 *Edit own offer:*

Use Case name	Edit own offer
Actor	Customer
Dependencies	List own offers
Purpose	Archive own offer
Trigger	Click 'archive' button
Assumption and pre-conditions	User is logged in
Basic path	1)System displays details of the offer and its corresponding book 2)Customer edits form which includes all the attributes of the offer and the book as well as list of categories and images related to this book 3)Customer saves changes 4)System validates data 5)System saves updated offer
Alternative flow of events	1)Customer chooses archive option 2a1 Go to archive use case 2a2 Go to point 1 2)Customer chooses delete option 2b1 Go to delete use case 3)Customer chooses unarchive option 2c1 Go to unarchive use case 2c2 Go to point 1 4)Customer leaves the page or cancelled the operation: 2d1 System discards the changes 5)Data validation fails: 5e1 System displays an error 5e2 go to point 1
Post condition	User updated own offer

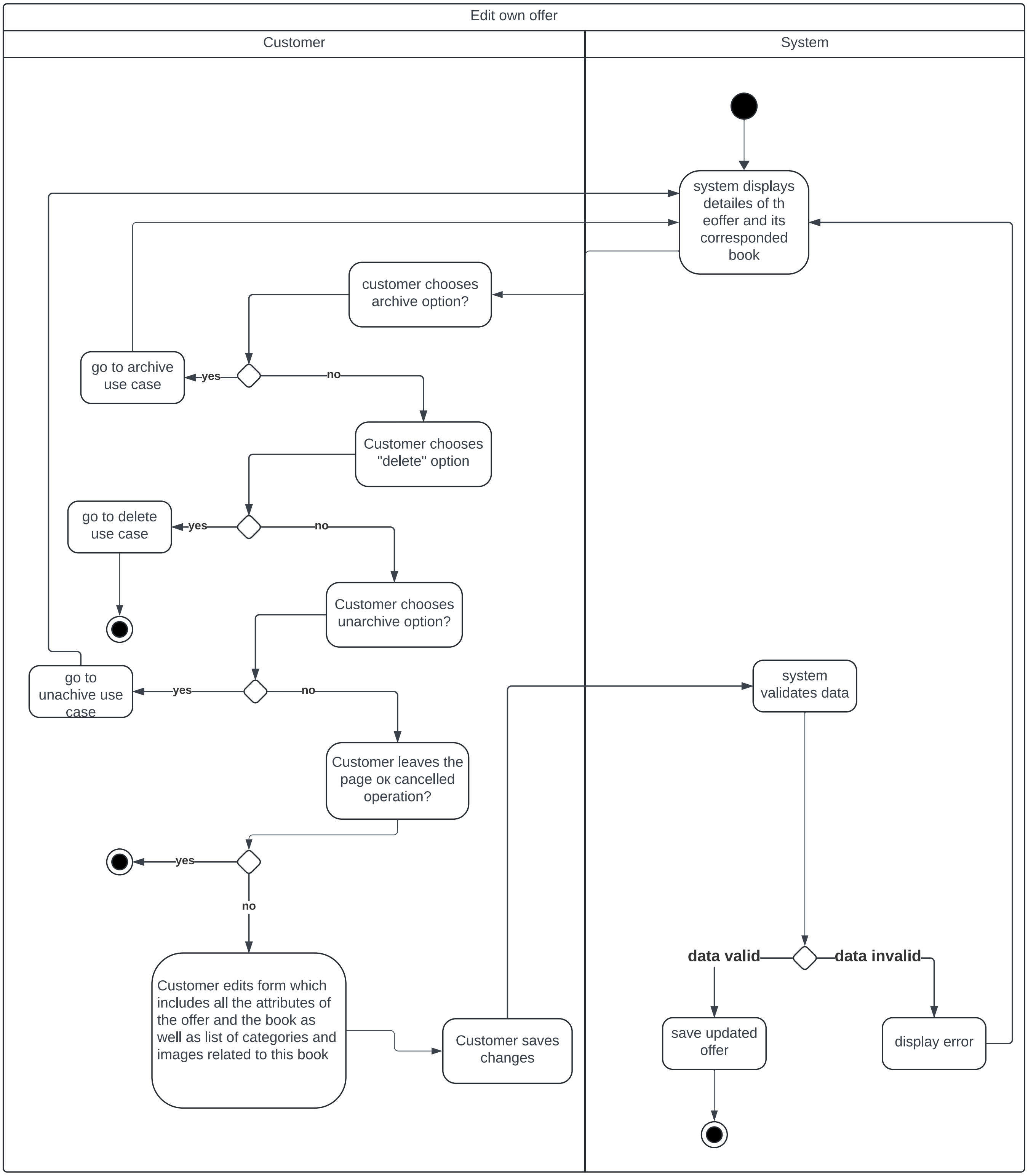
4 Analytical class diagram



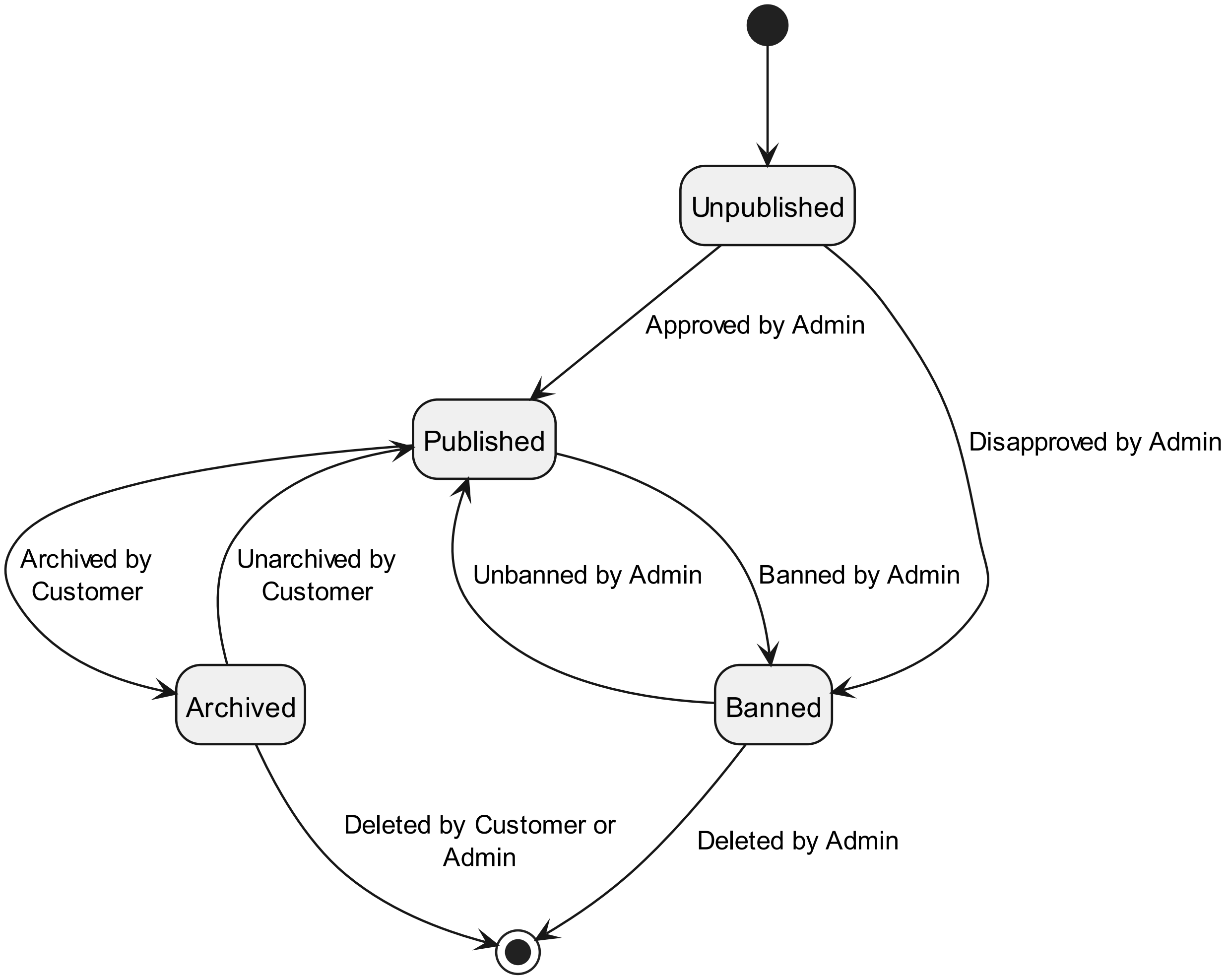
5 Design class diagram


User
{abstract}
-name: String
-surname [0..1]: Optional<String>
-timeOfRegistrationL LocalDateTime
-enabled: boolean
-email: String {unique}:
-password: String
+login(String email String password): void
+register(): void
+logout():void
{ disjoint, complete }
Administrator
-uniqueKey { unique}: String
+banOffer(Offer offer): void
+approveOffer(offer): void
+deleteOffer(offer): void
+unbanOffer(offer): void
Customer
-telephoneNumber: String
-dateOfBirth:LocalDate
-address: Address
- /age: int
+createOffer(offer): void
+removeOffer(Offer offer): void
+getOwnOffers(): Set<Offer>
+addOfferToFavourite(FavouriteOffer offer): void
+removeOfferFromFavourite(FavouriteOffer offer): void
+getFavoriteOffers():Set<Offer>
+sendReport(String description): void
+editOwnOffer(Offer offer):void
+archiveOffer(Offer offer): void
+unarchiveOffer(Offer offer): void
Address
-country: String
-city[0..1]: Optional<String>
-street[0..1]: Optional<String>
-homeNumber[0..1]: Optional<String>
-postalCode: String
<<sent by
send>
1
0..*
Report
-description: String
+getAllReports(): Set<Report>
0..*
is about>>
has>>
1
owns>>
1
<<owned by
0..*
Offer
-price: double
-numberOfCopies: int
-endDate[0..1]: Optional<LocalDateTime>:
-discount[0..1]: Optional<double>
-pubishingTime[0..1]: Optional<LocalDateTime>
-publishState: {Archived, Unpublished, Published, Banned}
-roles: EnumSet {LimitedTimeOffer, DiscountOffer, BasicOffer}
+getAllOffers(): Set<Offer>
+getAllOffersSortedBy(Comparator<Offer> comparator): Set<Offer>
+findOffersByCriteria(String criteria): Set<Offer>
setPrice(double newPrice)
{ at least one role should be chosen }
{ new price cannot excee old one by more than 20% }
{ favourite offer cannot be asigned to the user who owns it }
FavouriteOffer
-defaultDescription: String
-description: String
1
0..*
0..*
1
<<interface>>
LimitedTimeOffer
getDaysRemains(): int
<<interface>>
DIiscountOffer
getPriceWithDiscount(): double
<<interface>>
BasicOffer
getPublishingTime(): LocalDateTime
ContactInfo
-email: String
-telephoneNumber [0..1]: Optional<String>
-sociaMediaLink [0..1]: Optional<String>
1
has>>
attached to>>
1..*
Image
-image: byte[]
-format: {png, jpg, jpeg}
-isPreview: boolean
0..*
attached to>>
has>>
1
Category
-name: String
1..*
<<attached to
has>>
0..*
has>>
<<attached to
1
1
Book
<<abstract>>
-name: String
-yearOfPublishing: LocalDate
-description: String
-author: String
-isbn: String
<<abstract>> calculateEstimatedTimeOfReading(): double
{ disjoint, complete }
PaperBook
-numberOfPages: int
DiskBook
-durationInHours: double
-diskFormat: {CD, DVD, Blu-ray, Mini-Cd, Mini-Dvd, Mini-blu-ray}
PaperBookWithDisk
-diskIsGlued:boolean
-durationInHours: double
-diskFormat: {CD, DVD, Blu-ray, Mini-Cd, Mini-Dvd, Mini-blu-ray}
<<interface>>
IDiskBook
+setDurationInHours(durationInHours:double): void
+getDurationInHours(): double
+setDiskFormat(diskFormat: {CD, DVD, Blu-ray, Mini-Cd, Mini-Dvd, Mini-blu-ray}) void
getDiskFormat(): {CD, DVD, Blu-ray, Mini-Cd, Mini-Dvd, Mini-blu-ray}

6 Activity diagram

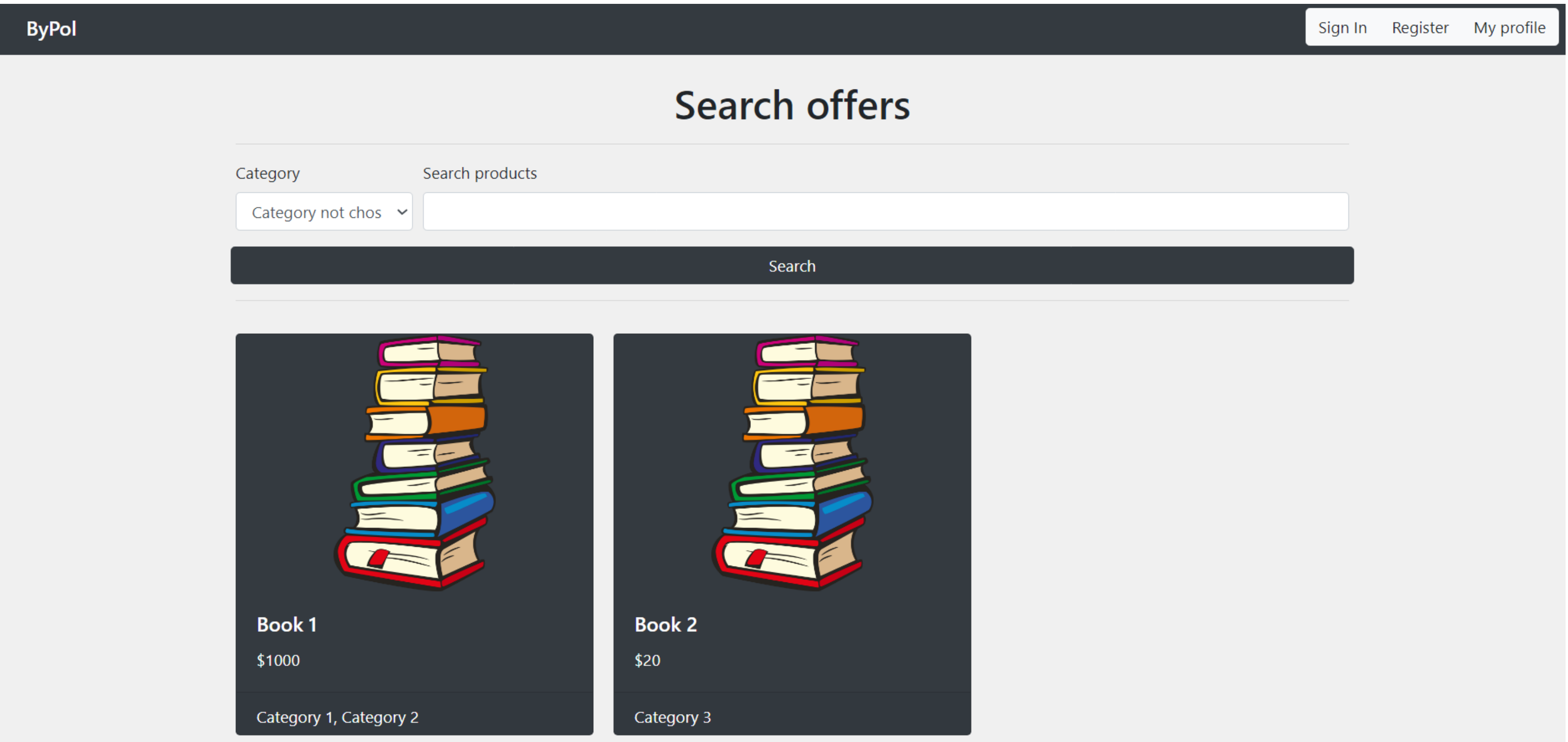

Edit own offer
Customer
System
system displays detailes of th eoffer and its corresponded book
customer chooses archive option?
go to archive use case
yes
no
Customer chooses "delete" option
go to delete use case
yes
no
Customer chooses unarchive option?
go to unachive use case
yes
no
Customer leaves the page ок cancelled operation?
yes
no
Customer edits form which includes all the attributes of the offer and the book as well as list of categories and images related to this book
Customer saves changes
system validates data
data valid
data invalid
save updated offer
display error

7 State diagram



8 GUI

Home page:



Search offers

Category: Category not chos

Search products:

Book 1
$1000
Category 1, Category 2

Book 2
$20
Category 3

Navigation Bar:

- The navigation bar at the top includes the brand name "ByPol" and links for "Sign In," "Register," and "My profile." This is a common layout that helps users quickly find essential navigation options.

Search Functionality:

- The search section prominently features a category dropdown and a search input field, allowing users to filter offers by category and search for specific products. The use of a large search button ensures that the search functionality is easy to find and use.

Product Display:

- Products are displayed as cards with a consistent design, including an image, title, price, and categories. This card-based layout is visually appealing and makes it easy for users to browse through multiple offers quickly.

Visual Hierarchy:

- The design employs a clear visual hierarchy with the page title "Search offersprominently displayed at the top. The search functionality is immediately below the title, followed by the product listings. This logical flow guides the user through the process of searching and viewing offers effectively.

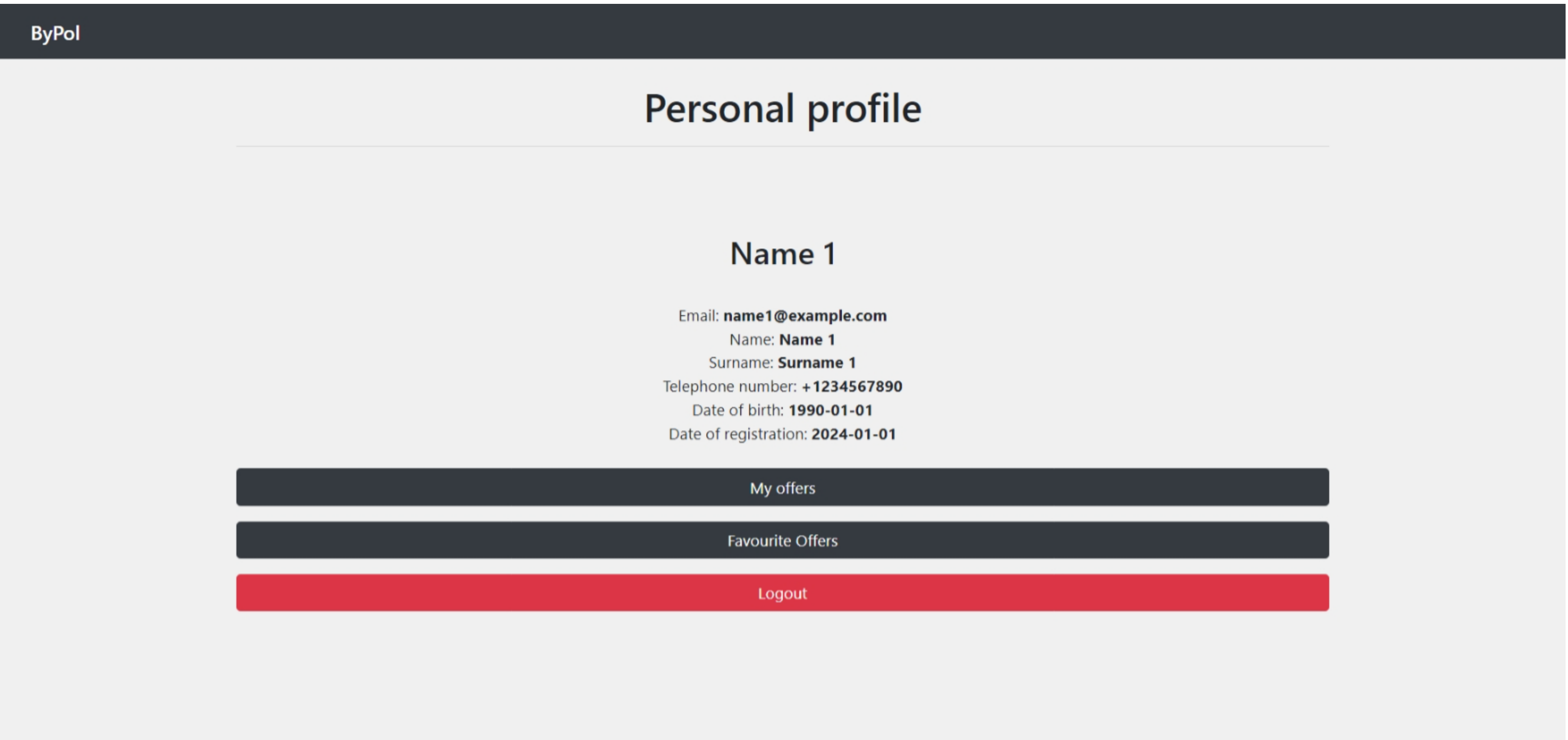
Use of Colors:

- The design uses a neutral color palette with dark backgrounds for the product cards, which contrasts well with the product images and text. This contrast makes the content easy to read and visually appealing.

Responsive Design:

- Although not explicitly shown, the layout appears to be designed with responsiveness in mind, ensuring that the page looks good on various devices, including desktops, tablets, and smartphones.

My profile



Personal profile

Name 1

Email: **name1@example.com**
Name: **Name 1**
Surname: **Surname 1**
Telephone number: **+1234567890**
Date of birth: **1990-01-01**
Date of registration: **2024-01-01**

Layout:

- *The page has a clean and centered layout with the title "Personal profile" prominently displayed at the top. This ensures that the user immediately knows they are viewing their profile information.*

Profile Information:

- *The user's profile information is displayed in a list format, which is easy to read. Each piece of information (email, name, surname, telephone number, date of birth, and date of registration) is clearly labeled, making it easy for the user to verify their details at a glance.*

Buttons for Navigation:

- *Below the profile information, there are three buttons: "My offers," "Favourite Offers," and "Logout." These buttons are distinctly colored, with the "Logout" button in red to signify its importance and potential impact. The use of dark buttons for navigation options helps differentiate actions the user can take within their profile.*

Design Aesthetics:

- *The design employs a minimalistic approach with a neutral color palette, which aligns with the overall aesthetic seen in the "Search offers" page. This consistency in design across different pages helps maintain a cohesive user experience.*

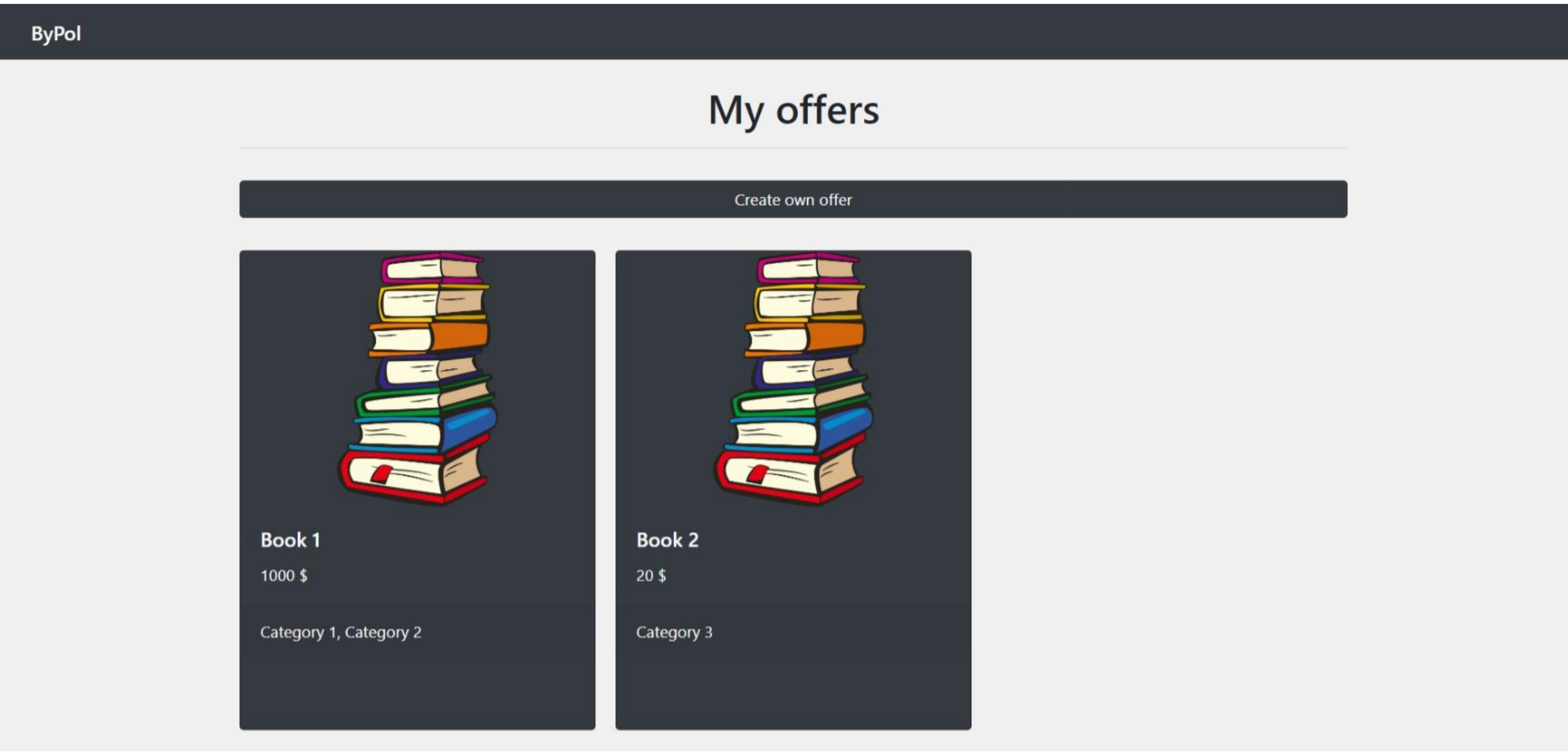
Typography:

- *The typography is clean and readable, with sufficient spacing between different sections of the profile information and buttons. The use of bold text for headings and important information helps to create a visual hierarchy.*

Responsiveness:

- *Although not explicitly shown, the centered and simple design suggests that the page is likely responsive and would display well on various devices, including desktops, tablets, and smartphones.*

My offers



Navigation Bar:

- The navigation bar at the top includes the brand name "ByPol," maintaining a consistent header across all pages. This helps users easily identify the brand and navigate through the site.

Page Title:

The title "My offers" is prominently displayed at the top, clearly indicating the purpose of the page. This ensures that users know they are viewing their offers.

- Create Offer Button: The "Create own offer" button is placed immediately below the page title and spans the width of the content area. This makes it easy for users to find and click on it if they want to create a new offer. The button's prominent placement encourages user engagement.
-

Offer Cards:

- Offers are displayed as cards with a consistent design, including an image, title, price, and categories. This card-based layout is visually appealing and makes it easy for users to browse through their offers.
- Each card contains a book image, the book title, price, and categories it belongs to. This information is essential for users to quickly identify and manage their offers.

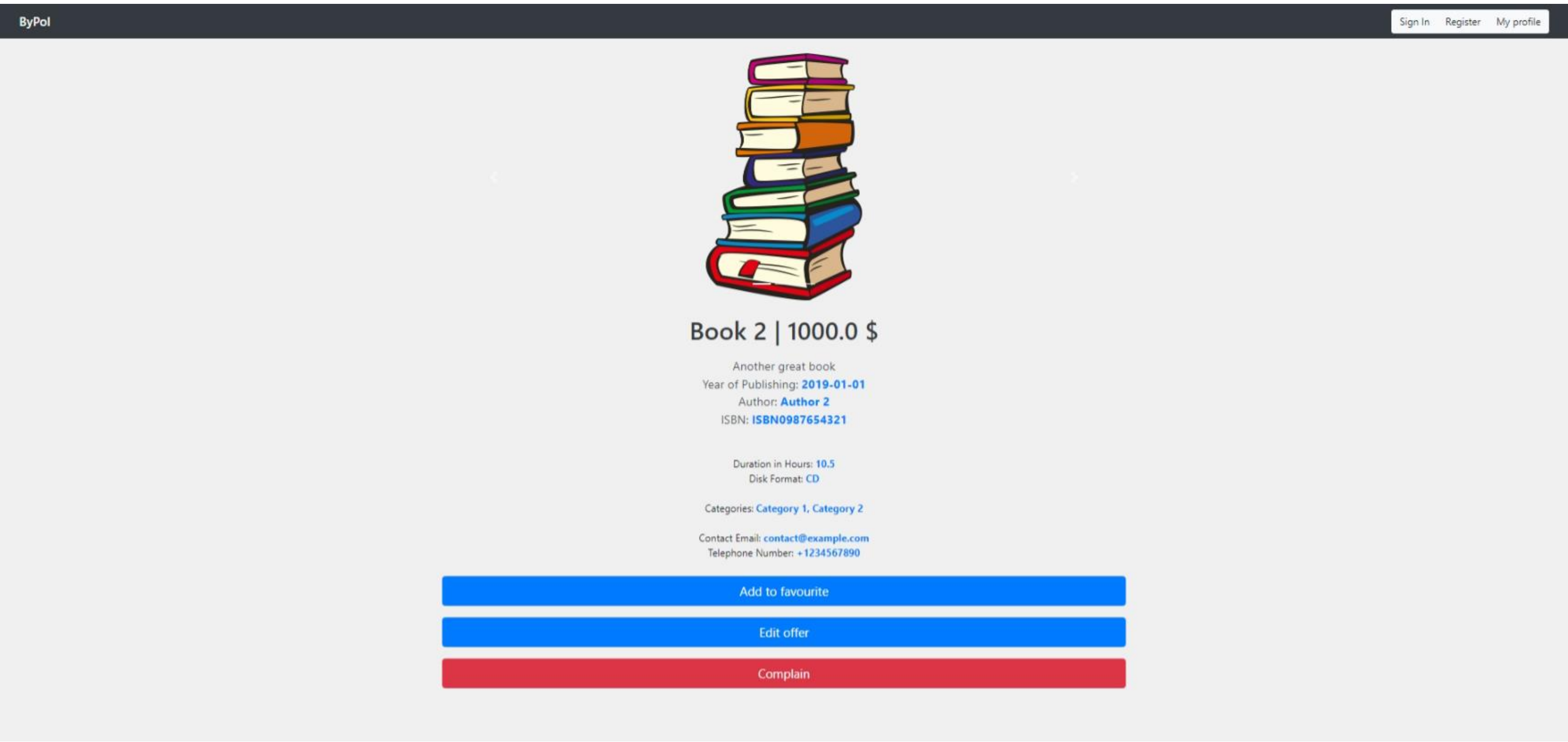
Visual Hierarchy and Clarity:

- The design employs a clear visual hierarchy with the "Create own offer" button and offer cards being the primary focus. The use of white space and consistent styling helps in maintaining clarity and readability.
- The background color and the card color contrast well, making the content stand out without being overwhelming.

Responsiveness:

- While the image does not explicitly show responsiveness, the layout suggests that it is designed to be responsive and adapt well to various screen sizes. This ensures a good user experience on both desktop and mobile devices.

Detailed info



Book 2 | 1000.0 $

Another great book
Year of Publishing: 2019-01-01
Author: Author 2
ISBN: ISBN0987654321

Duration in Hours: 10.5
Disk Format: CD

Categories: Category 1, Category 2

Contact Email: contact@example.com
Telephone Number: +1234567890

Navigation Bar:

- The navigation bar at the top is consistent with the other pages, including the brand name "ByPol" and navigation links for "Sign In," "Register," and "My profile." This maintains a uniform look and feel across the site.

Product Image and Title:

- The product image is prominently displayed at the top center of the page, ensuring it catches the user's attention immediately.
- The title of the book, along with the price, is displayed directly below the image. The use of bold and larger font sizes for these elements ensures they stand out.

Product Details:

- Detailed information about the book is presented in a structured format, including the description, year of publishing, author, ISBN, duration, disk format, categories, and contact information. The use of different font styles and colors (e.g., blue for links) helps differentiate various pieces of information and makes the text more readable.
-

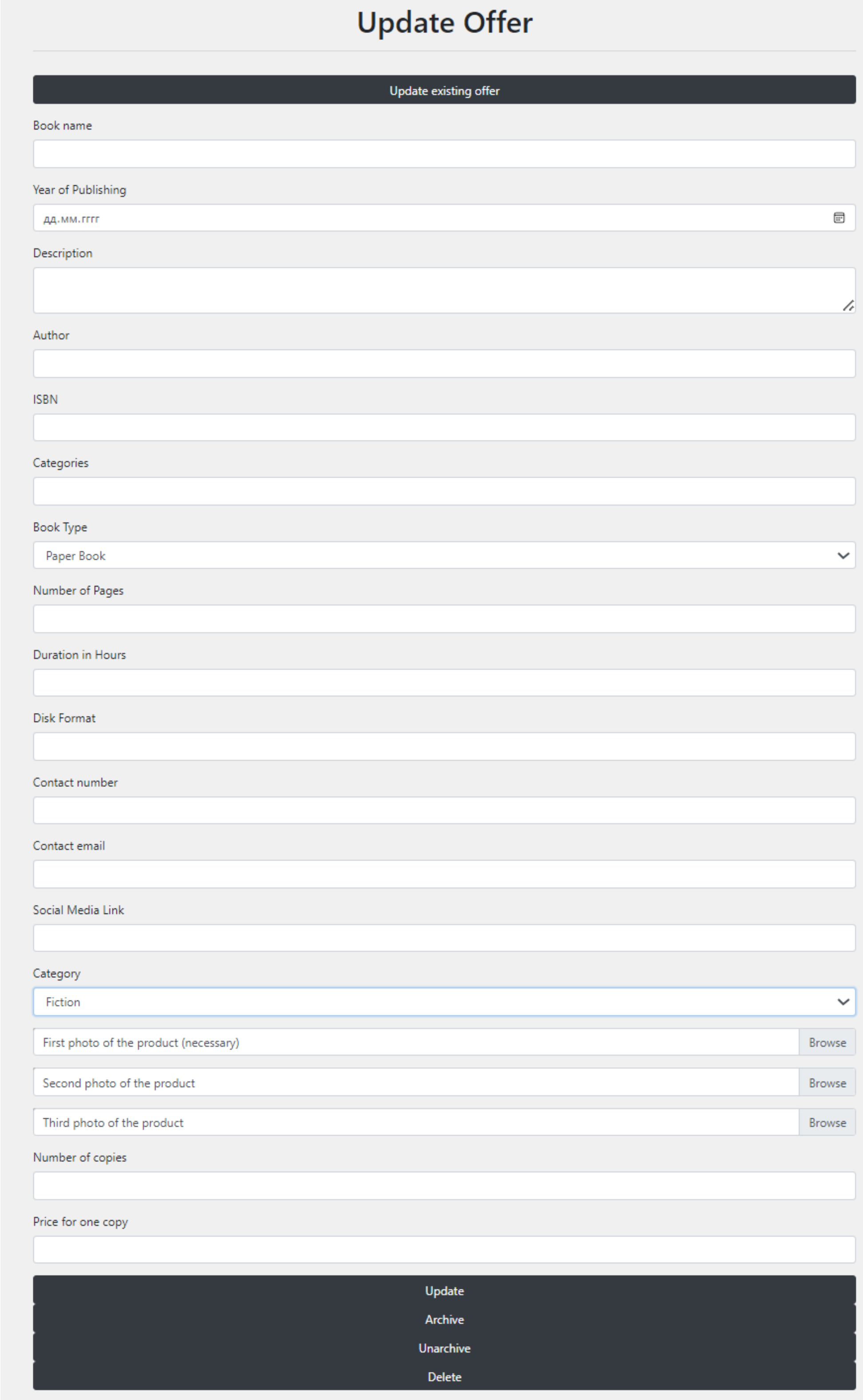
Call to Action Buttons:

- At the bottom of the page, there are three call-to-action buttons: "Add to favourite," "Edit offer," and "Complain." These buttons are distinctly colored (blue for primary actions and red for the secondary action), making them easily identifiable and encouraging user interaction.
- The placement of these buttons below the product details ensures that users have all the necessary information before taking any action.
-

Visual Hierarchy and Spacing:

- The page layout uses a clear visual hierarchy, with the product image and title at the top, followed by detailed information, and action buttons at the bottom. This logical flow guides the user through the page content naturally.
- Ample white space between different sections helps prevent the page from feeling cluttered and improves readability.

Edit offer info:



Update Offer

Book name

Year of Publishing: дд.мм.гггг

Description

Author

ISBN

Categories

Book Type: Paper Book

Number of Pages

Duration in Hours

Disk Format

Contact number

Contact email

Social Media Link

Category: Fiction

First photo of the product (necessary)

Second photo of the product

Third photo of the product

Number of copies

Price for one copy

Page Title:

- *The title "Update Offer" is prominently displayed at the top, clearly indicating the purpose of the page. This helps users quickly understand that they are on the page to edit an existing offer.*

Form Layout:

- *The form fields are neatly organized in a single column, making it easy to follow and fill out. The fields include Book Name, Year of Publishing, Description, Author, ISBN, Categories, Book Type, Number of Pages, Duration in Hours, Disk Format, Contact Number, Contact Email, Social Media Link, Category, and photo uploads. This comprehensive form ensures that all necessary details can be updated.*

Form Fields:

- *Each field is clearly labeled, providing users with an understanding of what information is required. The input fields for text, numbers, dates, and file uploads are appropriately used.*
- *The "Book Type" and "Category" fields are dropdown lists, which enhance usability by providing predefined options, reducing the chances of incorrect input.*

Action Buttons:

- *At the bottom of the form, there are four distinct action buttons: "Update," "Archive," "Unarchive," and "Delete." These buttons are clearly labeled and have different functionalities to manage the offer.*
- *The "Update" button is used for saving changes, while "Archive" and "Unarchive" are used for changing the offer's status. The "Delete" button is distinctly colored in red, signifying its importance and potential impact.*
-

Consistency and Clarity:

- *The design maintains consistency with other pages in terms of typography, color scheme, and overall layout. This creates a seamless user experience across the platform.*
- *The use of ample white space between form fields improves readability and prevents the form from looking cluttered.*

User Guidance:

- *The page includes a clear call to action at the top ("Update existing offer") and a structured form layout, guiding users through the process of updating their offer step by step.*

9 Discussion of design decisions and the effect of dynamic analysis

Language and Framework Choices

- Java: Chosen for its robustness, object-oriented capabilities, and widespread use in enterprise applications. Java provides a strong type system and a large ecosystem of libraries and frameworks.

- Spring Boot: Selected for its capability to simplify the development WEB applications. Spring Boot provides easy setup, auto-configuration, and a range of out-of-the-box functionalities that accelerate development.

- Hibernate: Used for ORM (Object-Relational Mapping) to handle the complex relationships and mappings between Java objects and the database. Hibernate's lazy loading and caching capabilities are leveraged to improve performance and manage associations efficiently.

- H2 Database: Selected for its in-memory database capabilities, making it suitable for development and testing purposes. H2 is lightweight and easy to configure, which speeds up the development process.

- HTML, CSS, JavaScript: Employed for the frontend to create a web-based UI, ensuring flexibility and simplicity in implementation. This stack provides a responsive and interactive user interface.

Class Hierarchy and Inheritance

- Flattened Class Hierarchy: The hierarchy for Book and its subclasses (PaperBook, DiskBook, PaperBookWithDisk) is designed using a flattened structure with discriminators to distinguish between different book types. This approach simplifies the design and avoids the complexity of deep inheritance trees.

- Composition over Inheritance: Where applicable, composition is used to combine functionalities. For example, PaperBookWithDisk combines features of both PaperBook and DiskBook.

- Type System and Constraints:

    Optional Types: In Java, the Optional type is used for attributes that may be null, ensuring clarity and reducing the risk of NullPointerException. For example, attributes like telephoneNumber in ContactInfo and endDate in Offer are optional and handled appropriately.

    - Enum Types: Enums are used for attributes like diskFormat and roles to enforce type safety and valid values. This approach ensures that only predefined values can be assigned to these attributes, reducing the risk of invalid data.

Analysis and Modifications

During the analysis, the following design elements were incorporated:

- *State Management:*

A publishState enum attribute could be added to the Book class to represent different states of a book (e.g., draft, published, archived). Although not explicitly mentioned in the diagram, it can be inferred for managing the book lifecycle.
Consistency Methods:

Methods to ensure consistency of state during changes, such as setting the duration and format for DiskBook, are added. These methods ensure that any changes to a book's composition or release formats are handled correctly.

- *User Actions:*

Methods for user interactions such managing favorite offers (addOfferToFavourite, removeOfferFromFavourite) were added to enhance user interaction and functionality. These methods allow users to interact with the system meaningfully.

ORM and Class Extent:

- ORM Facilities: Hibernate is used to manage class extents and relationships. Annotations like @OneToMany, @ManyToOne,

@ManyToMany, and @OneToOne are used to define the associations between entities. This setup ensures that relationships are managed efficiently and that database queries are optimized.

- Lazy Loading: Hibernate's lazy loading capabilities are leveraged to defer the loading of related entities until they are explicitly accessed. This approach improves performance by reducing the initial load time and fetching data only when necessary.

Frontend Implementation

- Web-Based UI: The UI is implemented using HTML, CSS, and JavaScript to ensure flexibility and ease of use. Frameworks like React or Angular can be used for a more dynamic and responsive interface. The frontend communicates with the backend through RESTful APIs, allowing for a decoupled and scalable architecture.

- Form Validation and User Feedback: JavaScript is used to provide form validation and user feedback, ensuring that users receive immediate responses to their actions. For example, when adding a book to the favorite list, the system can provide instant confirmation or error messages.

- Responsive Design: CSS is used to create a responsive design that adapts to different screen sizes and devices. This approach ensures that the application is accessible and user-friendly across various platforms.