

BOOKSWAP APPLICATION RESEARCH DOCUMENT

Author

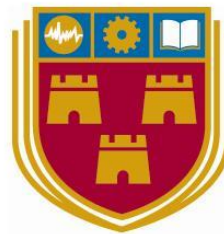
Ivan Yaremko
C00239239

Supervisor

Dr Chris Staff

Submission date

5/11/2021



INSTITUTE *of*
TECHNOLOGY

CARLOW

Institiúid Teicneolaíochta Cheatharlach

ABSTRACT

The purpose of this report researches a feasible way to implement an application for swapping second-hand books. The following topics were researched.

- The current state of the book industry.
- Similar implementations of second-hand books swapping.
- Technologies required to implement the application.
- Architecture designs to understand the whole application.
- Algorithms for book recommendations.

The results of the research indicate a strong need for a solution that allows trading second-hand books, considers technologies and architecture designs needed to implement such a solution.

Table of Contents

ABSTRACT	1
Table of Contents	2
TABLE OF FIGURES	4
INTRODUCTION	5
DEMAND FOR BOOKS	6
Do people read books?	6
E-Books the final nail in the coffin for printed books?	6
Turbulent times for printing books	8
Utility of research topic	8
RESEARCH OF SIMILAR BUSINESSES	9
Bookswap UK	9
Utility of research topic	10
TECHNOLOGIES	11
Architecture designs	11
Clean Architecture	11
Architecture patterns	13
CQRS	13
Mediator	14
.NET Framework with C#	16
ASP.NET Core	16
Web API	16
Entity Framework	17
Core Identity	17
React	17
Axios	18
MobX	18
Routing	19
TypeScript	19
ISBN API	20

DETAILED ALGORITHM & SYSTEM DESCRIPTIONS.....	21
Recommendation systems	21
SUMMARY AND CONCLUSION.....	22
REFERENCES	23
Plagiarism Declaration.....	26

TABLE OF FIGURES

Figure 1 Estimate of books/eBooks purchases.....	7
Figure 2 U.S data on total trade sales	7
Figure 3 The Clean Architecture.....	12
Figure 4 Modified clean architecture	13
Figure 5 CQRS design	14
Figure 6 flow of control	14
Figure 7 mediator pattern.....	15
Figure 8 MobX design	19
Figure 9 TypeScript relationship with JavaScript.....	20

INTRODUCTION

Could a global book shortage open an untapped market for second-hand books? A recent surge of book sales [1] has increased the popularity of pleasure reading. With popularity comes the demand for more printed books however the future supply of physically making books is getting impeded [7] .

While efforts are being made to ease the difficulties of book printing, this research report will investigate the current need for a second-hand book trading solution and best practices of implementing such an application.

This objective of this report is to research the following topics:

- The current dynamics in the book industry, investigating if printed books are more prevalent than e-books and the future of the industry.
- Investigating existing implementations of trading second-hand books.
- Researching suitable technologies and architecture designs required to implement the application.
- Recommendation algorithms to enhance the applications utilities.

This research report will first discuss the current demand for books, then investigate current examples of solutions that trade second-hand books, then explain how architecture designs and patterns will be used in the project, and finally discuss how a recommendation algorithm can be implemented.

DEMAND FOR BOOKS

The aim of researching this topic is to understand the current demand for books.

Investigating this topic will clarify questions such as:

- Is reading books still popular?
- What are the current dynamics in the book market?
- Are printed books still dominant in the industry or do eBooks edge ahead?
- What does the future hold for printed books?

Answers to these questions will shape the project and justify the need for a solution such as BookSwap.

Do people read books?

The popularity of books can be interpreted through current sales.

- For the first quarter of 2021, book sales have increased by 29% when compared to the same period in 2020 [1].
- In the Republic of Ireland, 13.1 million books were sold in 2020 which is an increase of 7.8% compared to 2019 [2].
- The revenue increase of €14 million brought the total revenue to €161.5 million in the Irish market. Such a figure has not been achieved since 2008.

With the quarantine lockdowns, people had more free time to enjoy hobbies such as reading. This has caused a resurgence of popularity for pleasure reading and it is evident that people have kept the commitment post-lockdown.

E-Books the final nail in the coffin for printed books?

According to the data from Statista's Advertising & Media Outlook [3], printed books are still the dominant format. From personal experience, I am not surprised. E-books are a convenient tool for reading when on the move, but nothing beats the feel of a real book when reading. Over half of the population in Germany and France bought a printed book in 2020. It is evident from across the world, with different countries and languages, which printed books are still significantly more popular than E-books. Most recent data from the Association of American Publishers [4] further support this claim, hardback and paperback books made most sales for the month of February 2021 in the United States. With more people owning printed books, more second-hand books become available.



Figure 1 Estimate of books/eBooks purchases

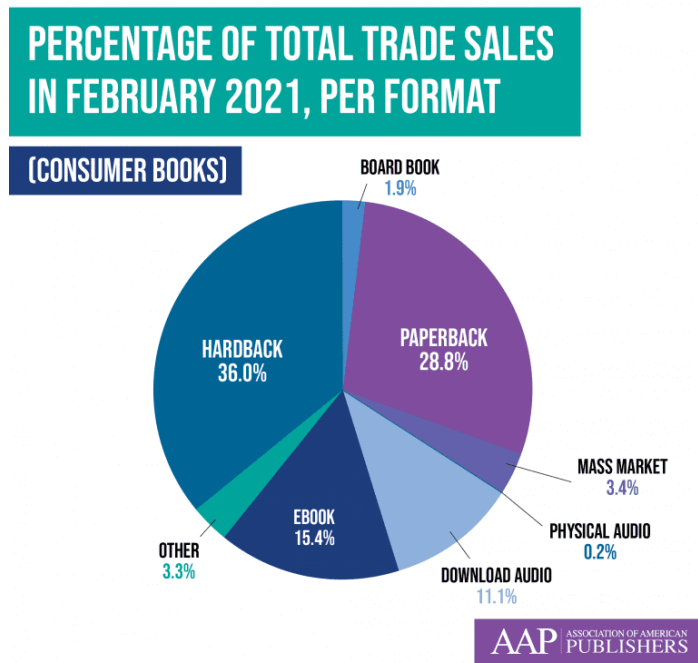


Figure 2 U.S data on total trade sales

Turbulent times for printing books

With the current Covid-19 pandemic affecting supply chains worldwide, it is no surprise that the book industry is also heavily affected. A perfect storm of conditions has experts predicting a book shortage [5]. With the recent increasing popularity of books, demand for printed material has also increased. Unfortunately, it is getting more complicated and expensive to physically make books.

“A shortage of raw materials ... is expected to negatively affect the printing inks market”
A quote from a report by the Business Research Company [6] states the dire situation of the ink market currently. China closed several raw material plants, due to environmental concerns, which has decreased the number of raw materials available to produce ink. Also, China is dominant in producing raw materials such as resins and refining them for ink. With the current shortage and competition for ink, the global printing ink market is expected to increase by 4% [6].

Paper and paper mills are faring no better in this current market climate. A report from the printing company Sheridan [7] states the price of wood pulp has risen from \$700/750 per metric tonne in 2020 to almost \$1200 in 2021. The report indicates that this spike was caused by China closing 279 pulp and paper mills [7], also capping the pulp capacity. The good news is that new pulp mills are being built across the world, but it will take some time for them to get up and running.

Utility of research topic

The shortage of raw materials and the cost increase of printing books has compounded problems in the book industry. It is likely that there will be a shortage of popular books printed and on the bookshelves of retailers. The price increase for books is also highly likely. The current and expected future difficulties in acquiring printed books opens a market for second-hand books. The research on this topic shows that there is an opportunity for a solution that specialises in exchanging second-hand books.

RESEARCH OF SIMILAR BUSINESSES

The aim of researching this topic is to do an analysis of similar businesses or projects.

Analysing this topic will give insight on use cases such as:

- How do users interact with the business?
- How do users search for books?
- How do users negotiate a swap for books?
- How does the business generate revenue?

Bookswap UK

The amply named BookSwap [8] is a business located in the United Kingdom that specialises in second handbooks. The website allows users to register to become members, search the catalogue, place a book on the market, order a book from the market and a wish list system. The business uses a third party to validate and deliver books.

The business model orientates itself around an ordering point system. To order a book from the marketplace a member must spend:

- ordering point or its equivalent of £3.
- A mandatory swapping fee of £1.19.
- A delivery fee of £2.66.

The total cost of ordering a book is 1 ordering point + £3.85 or £6.85 in total without the ordering point. A member can purchase ordering points through the website.

For a member to place a book in the marketplace they must:

- Use the search function to find the book to offer.
- Select the book.
- Select the appropriate quality rating.
- Submit the book offer to the marketplace.

A member is rewarded one ordering point when sending a book that is ordered by another member. A book must be packaged in padded envelopes or recycled materials. To send the book a member must drop it off at a Hermes Locker or parcel shop with the posting label or QR code. Hermes parcel courier is used as a third party to verify a book has been sent and delivered.

Utility of research topic

This research topic explored use cases related to swapping books. How users browse, order, and send books.

This topic has also shown how BookSwap UK has monetised its business.

BookSwap UK generates revenue from a mandatory swapping fee, the fee cannot be omitted as it needs to be paid first to generate the shipping label.

A third-party system to execute and verify deliveries eliminates the need for people to meet face to face to swap books.

The research on this topic has inspired ideas on ways to generate revenue without the need for a courier and mandatory swapping fees. The project can implement systems such as novel recommendations and user trust ratings, which can be offered to premium paying members.

This topic suggests further research on a feasibility study of integrating a third-party courier system for the project.

TECHNOLOGIES

The purpose of this research is to investigate suitable methodologies, technologies, frameworks, and libraries to develop the BookSwap project. The goal of this topic is to research in-depth the following:

- Software architecture design.
- Software architecture patterns.
- Frameworks.
- Libraries.
- Programming Languages.
- Continuous Integrations / Continuous Delivery Pipeline.

Architecture designs

Good software architecture is vital in developing a modular, maintainable, scalable application. There are different software architecture designs that vary in their details. However, they all try to achieve the same goal. The goal is the separation of concerns, which is a design principle for separating an application into distinct sections.

Clean Architecture

Clean Architecture was created by Robert C. Martin and is promoted through his book *Clean Architecture* [9]. This architecture is intended to keep the codebase under control. It achieves this by ensuring that the application code and logic are written in a way that they do not have any direct dependencies. The core of the system should not be changed if the framework or UI changes. This protects the core of the system and makes the external dependencies completely replaceable.

Figure 3 illustrates the standard clean architecture design. The circles represent different areas of software. The inner circles are the policies whereas the outer circles are the mechanisms. The more inwards you go the higher level the software becomes.

The golden rule that makes this architecture work is the dependency rule:

“Source code dependencies must point only inward, toward higher-level policies.” [9]

To put it simply, the inner circle must not know anything about the outer circle. Anything declared in the outer circle must not be mentioned by the code in the inner circle. That can include functions, classes, variables, or other entities. The architecture can be adjusted as needed if the dependency rule is adhered to.

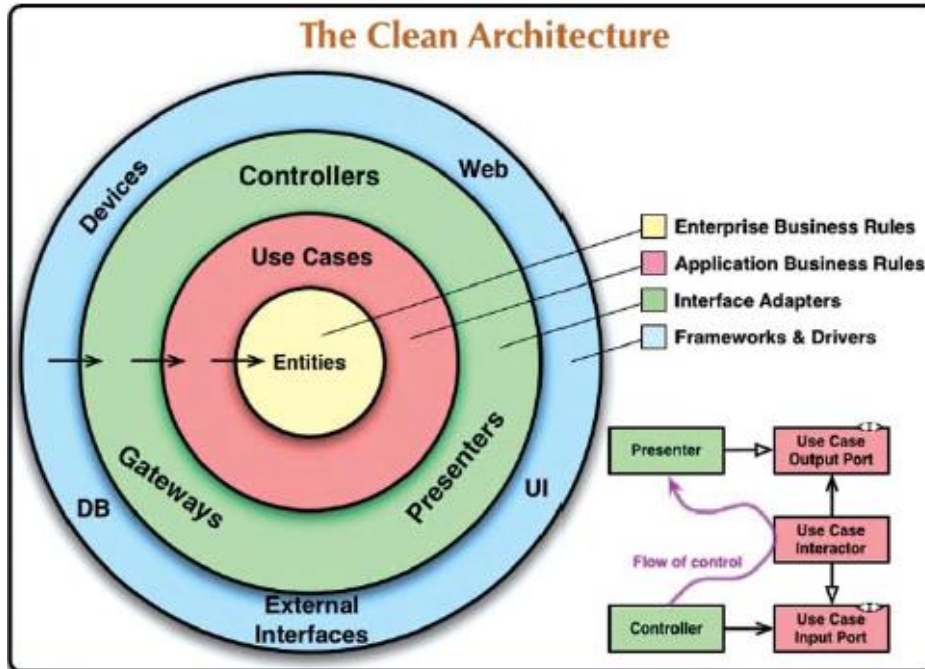


Figure 3 The Clean Architecture [9]

Figure 4 demonstrates a modified version of the clean architecture that will be used in the development of the backend application for the BookSwap project. The design still follows the dependency rule of clean architecture. The project structure is as follows:

- The domain layer contains the enterprise logic.
- The application layer contains the business logic.
- The application layer has a dependency on the domain. Any of the domain entities will be accessible in the application layer.
- The persistence layer has a dependency on the domain layer which provides the connection to the database and translates written code into SQL queries.
- The applications layer job is to use the domain entities, query the database through the dependency on the persistence layer and return the logic to the API.
- The API layer is responsible for the HTTP requests and responds to them.
- The API layer has a dependency on the application layer where the business logic is processed.
- The API has access to both the domain and persistence layer through its transitive dependency via the application layer.

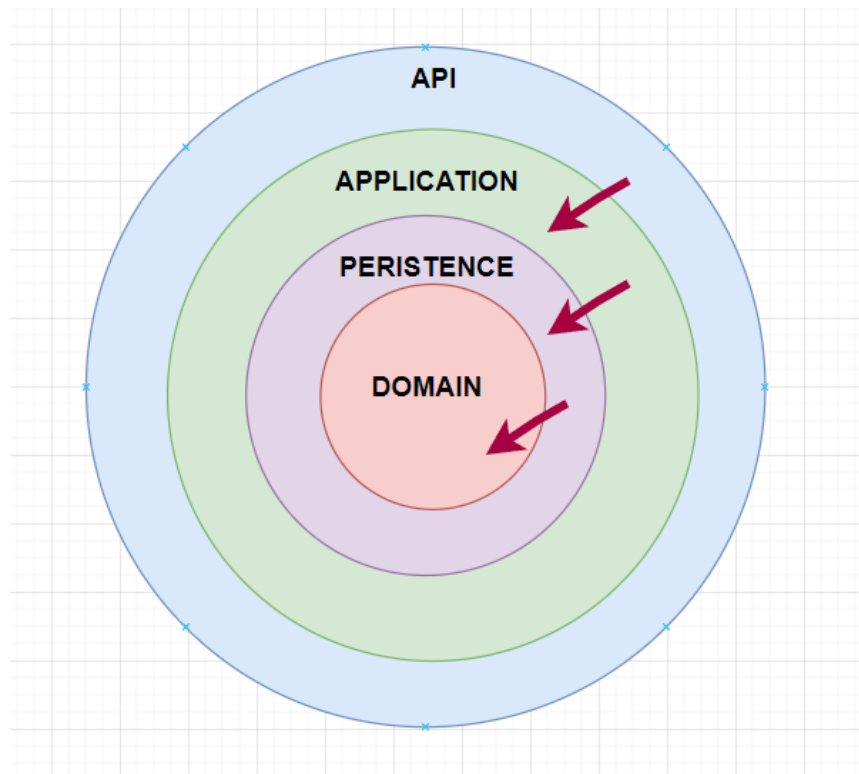


Figure 4 Modified clean architecture

Architecture patterns

When defining the term pattern Christopher Alexander's quote is often used [10],

“Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice”

Christopher Alexander is an architect, in his quote he is talking about buildings however, the definition works for software patterns as well. Patterns focus on a particular solution that is both common and effective in dealing with recurring problems.

CQRS

CQRS stands for Command and Query Responsibility Segregation [11]. This pattern is used to separate the create/update and read operations for a data store. This pattern offers a clean scalable solution for an application.

Create/update operations are known as commands in this pattern and reads are queries. CQRS separates the commands and queries, commands do something with the database and queries read data from the database. The mediator pattern is used to mediate between the different layers in the clean architecture design of this application.

The commands generally:

- Modifies state of the database.
- Should not return a value.

Whereas a query generally:

- Answer a question.
- Does not modify state.
- Should return a value.

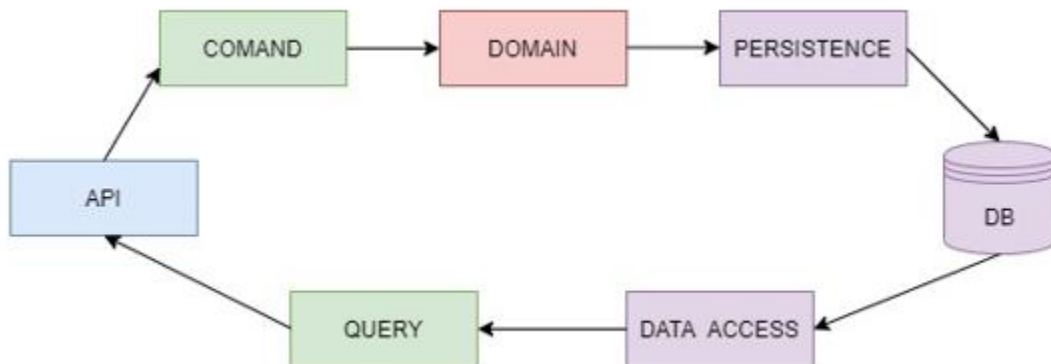


Figure 5 CQRS design

Figure 5 illustrates the CQRS cycle. When a client makes a request to the backend application the API is either going to:

- Make a command request to do something with the domain entities and use the persistence layer to update the database.
- Make a query to retrieve data using the data access to query the database.

Mediator

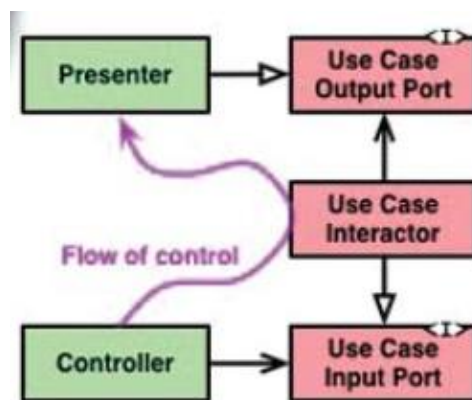


Figure 6 flow of control

The clean architecture specifies the flow of control as illustrated by Figure 6 [9]. With referring to the modified design, in Figure 6 the green represents the API layer and red represents the application layer. The application layer is responsible for the business logic and the flow of control travels from the API controller to the application layer and

back out through. The API layer has two jobs, it is going to receive an HTTP request and it is going to respond with an HTTP response.

When the API controller receives an HTTP request from a client the controller is going to:

- Send the request to the use case imports, which is the application layer.
- The use case interactor is going to process the business logic.
- The use case output will return an object via the presenter back to the client

This flow of control use case will be implemented in the application with a tool called MediatR [12]. MediatR is a library that describes itself as “*Simple mediator implementation in .NET*” [12]. It is a tool that allows process managing and supports request/response, command, and queries.

Figure 7 demonstrates the design on the mediator pattern. The API controller resides in the API layer and the handler in the application layer.

When a request comes to the API controller:

- The API controller will send this request via the mediator send method.
- The request could be either a query or a command.
- The mediator will send this request to the mediator handler.
- The handler will handle the use case.
- The handler will also return an object to the API controller.
- The API controller will then return the object with an HTTP response to the client.

For an example use case, a client wants to retrieve a book from the database. This query would be sent via the mediator send method to the mediator handler where it is going to process the business logic and return the object back via the API controller.

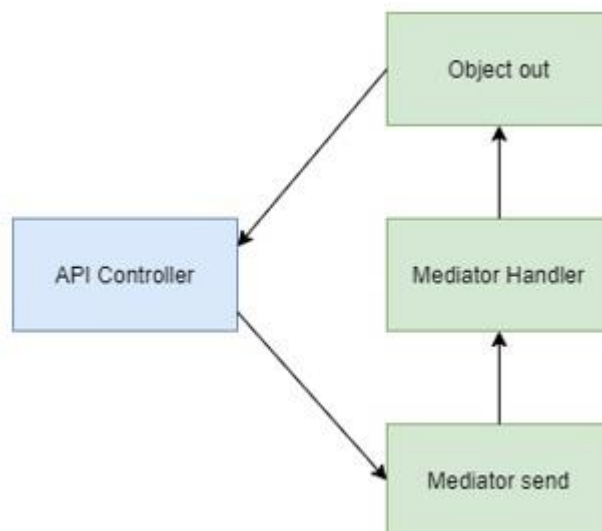


Figure 7 mediator pattern

.NET Framework with C#

The .NET Framework is a software framework developed by Microsoft [13]. .NET is a free, cross-platform, open-source developer platform for building applications. All .NET Core is open source including the class libraries, runtime, compilers, ASP.NET Core web framework, Entity Framework, and other frameworks. The framework is maintained by various GitHub repositories which typically use the MIT or Apache 2 licenses.

The framework is designed to fulfil the following objectives:

- Provide an object-oriented programming environment.
- Eliminate performance problems of scripted or interpreted environments.
- Promote safe execution of code.
- To be extensible.
- To integrate with Web standards and best practices.
- Make development simpler.

.NET applications can be written in C#, F# or Visual Basic. The backend for the BookSwap application will be written using the .NET Framework with C# because:

- It follows the object-oriented paradigm, which includes encapsulation, inheritance, and polymorphism.
- It is a type-safe language that enforces type safety at run time and supports static typing which enforces type safety at compile time.

ASP.NET Core

ASP.NET Core is an open-source web-development framework. Microsoft rewrote the framework to combine the previously separated ASP.NET MVC and ASP.NET Web API into a single model. The big advantages of using ASP.NET Core is [14]:

- Supports multiple platforms, applications can run on Windows, Linux, and Mac.
- It includes a built-in Inversion of Control (IoC) container for automatic dependency injection.
- It can integrate with modern front-end UI Frameworks such as React.

Web API.

ASP.NET Web API is a framework for building HTTP services that can be accessed from any client such as browsers or smartphones. It is a strong platform for building RESTful applications. The framework supports:

- Convention based CRUD actions such as HTTP verbs GET, POST, PUT and Delete.
- Multiple text formats such as JSON and XML.
- ASP.NET MVC features such as controllers and routing.

Entity Framework

Entity Framework (EF) is an open-source Object-Relational Mapping framework for a .NET application and is supported by Microsoft. EF helps developers to work with data using objects of domain-specific classes without the need of focusing on the database tables and columns where the data is stored. EF allows developers to work at a higher level of abstraction when dealing with data. The official definition of EF [15],

“Entity Framework (EF) is an object-relational mapper that enables .NET developers to work with relational data using domain-specific objects. It eliminates the need for most of the data-access code that developers usually need to write.”

The EF framework supports several features:

- It is cross-platform and can run on Windows, Linux, or Mac.
- It creates an Entity Data Model based on the Plain Old CLR Object entities with get/set properties. This is used when querying or saving entity data to the database.
- It allows the use of C# LINQ queries to retrieve data from the database.
- It provides a set of migration commands that can create or manage database Schema.

Core Identity

ASP.NET Core Identity [16] is an Application Programming Interface (API) that supports user interface login functionality. The framework manages users, passwords, emails, roles, tokens and much more. With the Identity Framework, users can create an account and login with a username and password, or they can use external login providers such as Google, Facebook, and Twitter.

React

React is an open-source library that is used for building user interfaces (UI) for single-page applications (SPA). It is the view layer for web and mobile applications. React was created in 2011 by Jordan Walke [17], who is a software engineer working for Facebook. At JSConf 2013, React was made open source.

With React, developers can create large SPA applications that can manage data without reloading the page. The purpose of React is to be fast, scalable, and simple to learn and use. A React application is built using components. The definition for a component,

“Components let you split the UI into independent, reusable pieces, and think about each piece in isolation” [18]

A component allows a mixture of HTML and JavaScript to capture all the logic necessary to display a small section of a UI. These components are built upon each other to create a SPA.

Since React is a library and not a framework the developer is free to decide which additional libraries to integrate with React. React is only concerned with rendering the UI, other responsibilities are delegated to the numerous open-source libraries.

Axios

Axios [19] is an HTTP client library that allows clients to make requests to a given endpoint. Axios is promise-based, which gives the ability to use JavaScript's `async` and `await` for more readable asynchronous code. Axios can also intercept and cancel requests.

The advantages of using Axios for client-side HTTP requests are:

- Works default with JSON. Unlike the Fetch API library, Axios does not need to set headers or convert request body to JSON string.
- Axios uses function names to match HTTP methods. For example, to submit a GET request the client can use the `.get()` method.
- Axios has more “syntax sugar” that allows doing more with less code.
- Axios does better error handling. It can throw a range of 400 and 500 errors.

MobX

MobX [20] is an open-source state management tool. It is a standalone library that works with popular front-end frameworks such as React, Vue and Angular. The core value of MobX states that [20] *“MobX makes state management simple again by addressing the root issue: it makes it impossible to produce an inconsistent state.”*

A ‘state’ is a plain JavaScript object used by React to represent information about the component’s current situation. MobX makes state management simple by focusing on the root issue. MobX makes it impossible to produce an inconsistent state by having a

strategy that [20] “Makes sure that everything that can be derived from the application state, will be derived. Automatically.”

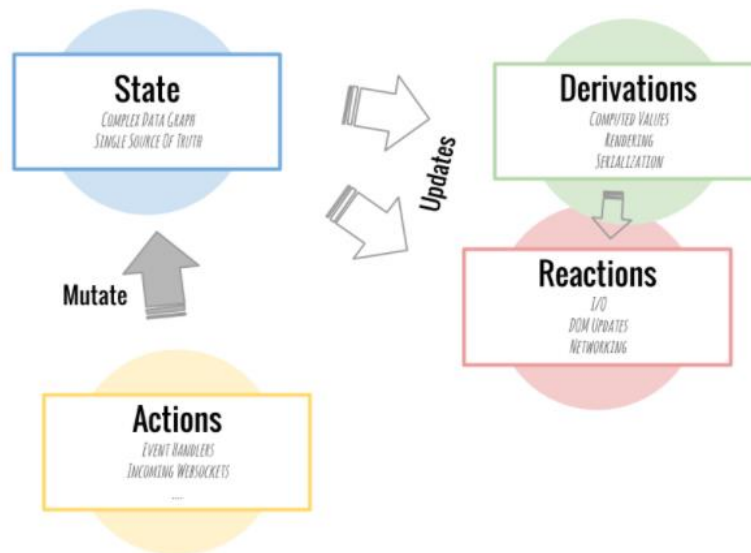


Figure 8 MobX design [20]

Figure 8 shows how MobX concepts and how it works.

- **State.** Application state is a graph of objects, arrays, primitives, and references that form the model of an application.
- **Derivations.** These are values that can be automatically computed from the state of an application.
- **Reactions.** Reactions run automatically to perform some tasks that is usually Input/Output related. Reactions make sure that Document Object Mapper is updated or that network requests are run automatically.
- **Actions.** Actions are things that alter the state. MobX will apply all changes to the application state caused by actions. These changes are automatically processed by all the *Derivations* and *Reactions*.

Routing

React Router [21] is a standard library for routing in React. React Router enables navigation among components in a React Application, it also allows changing the browser URL, and keeps the UI coordinated with the URL. React Router enables the display of multiple views in a Single Page Application.

TypeScript

TypeScript (TS) [22] was designed by Anders Hejlsber, who also designed C#, at Microsoft. TS is a strongly typed, object oriented, compiled language. TS builds on top

of JavaScript (JS). When TS code is compiled, it turns into JS code using the TS compiler.

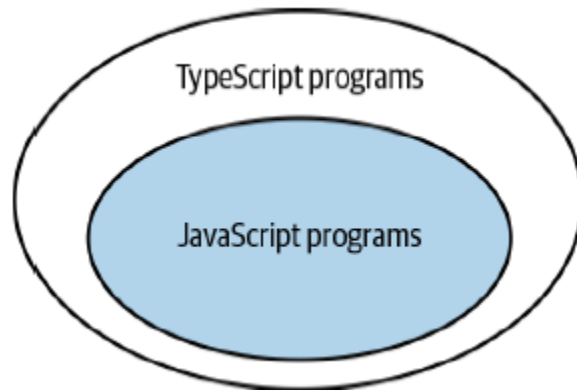


Figure 9 TypeScript relationship with JavaScript[23]

Figure 9 illustrates how all JavaScript is TypeScript but not all TypeScript is JavaScript. TS uses JS syntaxes and adds additional syntaxes for supporting types. If a JS program does not have any syntax errors, it is also a TypeScript Program. This means that all JavaScript programs are TypeScript programs, and it is simple to migrate JS codebase to TS. In conclusion, the benefits of using TypeScript are:

- **Compilation.** TS transpiler provides an error-checking feature. TS will compile the code and generate compilation errors if there are syntax errors. This enables the highlight of errors before the script is even run.
- **Strong Static Typing.** TS comes with a static typing and type interface system through the TypeScript Language Service.
- **OOP.** TS supports object-oriented programming concepts like classes, interfaces, and inheritance.

ISBN API

All books have an International Standard Book Number (ISBN). The BookSwap application will require functionality to interact with existing books. The application will need to outsource a provider for an ISBN API that will contain a catalogue of existing books.

GoodReads is the most popular website for book lovers. They have provided an API for developers in the past to interact with their catalogue. However, As of December 8th, 2020, Goodreads no longer issues new developer keys for their public developer API.

Fortunately, there are alternatives to the GoodReads API. ISBNdb [24] gathers data from hundreds of libraries, publishers, and merchants to compile a collection of unique

book data searchable by ISBN. ISBNdb is a paid service but they have an academic / non-profit discount for €5 a month.

DETAILED ALGORITHM & SYSTEM DESCRIPTIONS

Merriam-Webster defines algorithms as *“a set of steps that are followed in order to solve a mathematical problem or to complete a computer process”* [25]. Algorithms sort problems and aid computer automation. This research topic will investigate ways of implementing recommendation algorithms. This type of algorithm can be used as a feature within the BookSwap application. This feature would help users in recommending books to trade based on their interests.

Recommendation systems

A recommendation algorithm is a type of machine learning that can be trained to make future recommendations based on a list of users, a list of items, and a list of ratings given to items by users.

A recommendation system can be build using three algorithms [25]:

- 1) Content-based based filtering.
- 2) Collaborative filtering.
- 3) Hybrid filtering method.

For these algorithms to be implemented a dataset requires to contain:

- **Books.** Which contain relevant information to a book like an author, title, and publication year.
- **Users.** Which contain information about the users such as user id.
- **Ratings.** That contain information about users' ratings of books.

Content-based filtering is a technique of recommendation systems [26]. Content is attributes of things a user likes. This system uses what the user likes to recommend other similar content the user might like. The objective of content-based filtering is to classify content with specific keywords, learn what the user likes, and recommend similar things.

Collaborative filtering is a system used for building personalised recommendations [27]. In this system algorithms are used to make automatic predictions about user's interests. This is done by compiling preferences from other users. This type of algorithm assumes that users who have agreed in the past are likely to agree in the future.

Hybrid filtering is a combination of both content-based and collaborative filtering. When comparing hybrid filtering with content based and collaborative filtering, the

recommendation accuracy is generally higher in hybrid systems [28]. This is due to the lack of information about content in collaborative filtering, and about user's preferences in content filtering. This combination of information in hybrid systems leads to a better recommendation system.

SUMMARY AND CONCLUSION

Currently the book industry is experiencing a period of increased commercial activity. Printed books still dominate the industry when compared to eBooks. The printing industry is expecting difficulty to meet the supply of demand. This combination of events broadens the opportunities in the second-hand book market. An application catered for trading second-hand books with like-minded people can thrive in this environment.

This report investigated in-depth the following topics:

- The current state of the book industry.
- Similar implementations of trading second-hand books.
- Technologies and architecture required to implement the BookSwap project.
- Algorithms and systems to enhance features of the project.

This BookSwap project can create local communities to trade second-hand books with. This project could alleviate the pressure for people not being able to attain new books. Not only would users save money in reading new books, but the carbon footprint of books would reduce as more users are using second-hand instead of buying new.

REFERENCES

- [1]. Milliot, J., 2021. The Surge of Print Books Sales Continues. [online] PublishersWeekly.com. Available at: <<https://www.publishersweekly.com/pw/by-topic/industry-news/bookselling/article/86061-the-print-books-sales-surge-continues.html>> [Accessed 20 October 2021].
- [2]. McGreevy, R., 2021. Irish booksellers have record year as lockdown helps boost sales. [online] The Irish Times. Available at: <<https://www.irishtimes.com/culture/books/irish-booksellers-have-record-year-as-lockdown-helps-boost-sales-1.4468237>> [Accessed 20 October 2021].
- [3]. Statista. 2021. Books - Worldwide | Statista Market Forecast. [online] Available at: <<https://www.statista.com/outlook/amo/media/books/worldwide>> [Accessed 21 October 2021].
- [4]. AAP. 2021. AAP FEBRUARY 2021 STATSHOT REPORT: PUBLISHING INDUSTRY UP 24.7% FOR SECOND MONTH OF 2021, AND 16.4% YEAR TO DATE - AAP. [online] Available at: <<https://publishers.org/news/aap-february-2021-statshot-report-publishing-industry-up-24-7-for-second-month-of-2021-and-16-4-year-to-date/>> [Accessed 21 October 2021].
- [5]. Sherwin, A., 2021. Buy books today to beat paper shortage publishers warn. [online] inews.co.uk. Available at: <<https://inews.co.uk/culture/books/christmas-books-2021-buy-presents-beat-paper-shortage-publishers-1235531>> [Accessed 21 October 2021].
- [6]. Reportlinker.com. 2021. Printing Inks Global Market Report 2021: COVID 19 Impact and Recovery to 2030. [online] Available at: <https://www.reportlinker.com/p06018799/Printing-Inks-Global-Market-Report-COVID-19-Impact-and-Recovery-to.html?utm_source=GNW> [Accessed 21 October 2021].
- [7]. Parente, S., 2021. The Scarcity of Paper – Pulp (non)Fiction - Sheridan. [online] Sheridan. Available at: <<https://www.sheridan.com/journals-blog/the-scarcity-of-paper-pulp-nonfiction>> [Accessed 21 October 2021].
- [8]. Bookswap.co.uk. 2021. *Bookswap*. [online] Available at: <<https://www.bookswap.co.uk/>> [Accessed 26 October 2021].

- [9]. Martin, R. (2017). *Clean Architecture: A Craftsman's Guide to Software Structure and Design (Robert C. Martin Series)* (1st ed.). Pearson.
- [10]. Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., & Angel, S. (1977). *A Pattern Language: Towns, Buildings, Construction (Center for Environmental Structure Series)*. Oxford University Press.
- [11]. Narumoto, M. N. (2021, June 14). *What is the CQRS pattern? - Azure Architecture Center*. Microsoft Docs. <https://docs.microsoft.com/en-us/azure/architecture/patterns/cqrs>
- [12]. Bogard, J., 2021. *GitHub - jbogard/MediatR: Simple, unambitious mediator implementation in .NET*. [online] GitHub. Available at: <<https://github.com/jbogard/MediatR>> [Accessed 28 October 2021].
- [13]. G. (2021, September 15). *Overview of .NET Framework - .NET Framework*. Microsoft Docs. <https://docs.microsoft.com/en-us/dotnet/framework/get-started/overview>
- [14]. Microsoft. (2016, May 15). *What is ASP.NET Core? | .NET*. <https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet-core>
- [15]. R. (2020, February 19). *Entity Framework*. Microsoft Docs. <https://docs.microsoft.com/en-us/aspnet/entity-framework>
- [16]. R. (2021c, September 14). *Introduction to Identity on ASP.NET Core*. Microsoft Docs. <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-5.0&tabs=visual-studio>
- [17]. Team –. (2020, July 5). React. <https://reactjs.org/community/team.html>
- [18]. *Components and Props* –. (2021, June 2). React. <https://reactjs.org/docs/components-and-props.html>
- [19]. *npm: axios*. (2021, October 25). Npm. <https://www.npmjs.com/package/axios>
- [20]. *MobX: Ten-minute introduction to MobX and React*. (2021, October 5). Mobx.Js.Org. Retrieved October 31, 2021, from <https://mobx.js.org/getting-started>

- [21]. *React Router: Declarative Routing for React*. (2021, July 21). ReactRouterWebsite. Retrieved October 31, 2021, from <https://reactrouter.com/>
- [22]. *JavaScript With Syntax For Types*. (2021, September 21). <https://www.typescriptlang.org/>. Retrieved October 31, 2021, from <https://www.typescriptlang.org/>
- [23]. Vanderkam, D. (2019). *Effective TypeScript: 62 Specific Ways to Improve Your TypeScript* (1st ed.). O'Reilly Media.
- [24]. *ISBNdb API Documentation v2 | ISBNdb*. (2021, August 18). Isbndb.Com. Retrieved October 31, 2021, from <https://isbndb.com/apidocs/v2>
- [25]. Merriam-Webster, M. W. (2021). *algorithm*. The Merriam-Webster.Com Dictionary. <https://www.merriam-webster.com/dictionary/algorithm>
- [25]. Agrawal, R. (2021, June 29). Book Recommendation System | Build A Book Recommendation System. Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2021/06/build-book-recommendation-system-unsupervised-learning-project/>
- [26]. Mishra, U. (2021, May 21). What is a Content-based Recommendation System in Machine Learning? | Analytics Steps. [www.Analyticssteps.Com](https://www.analyticssteps.com/blogs/what-content-based-recommendation-system-machine-learning). <https://www.analyticssteps.com/blogs/what-content-based-recommendation-system-machine-learning>
- [27]. Techopedia. (2012, September 12). Collaborative Filtering (CF). Techopedia.Com. <https://www.techopedia.com/definition/1439/collaborative-filtering-cf>
- [28]. Geetha, G., Safa, M., Fancy, C., & Saranya, D. (2018). A Hybrid Approach using Collaborative filtering and Content based Filtering for Recommender System. *Journal of Physics: Conference Series*, 1000, 012101. <https://doi.org/10.1088/1742-6596/1000/1/012101>

Plagiarism Declaration



*I declare that all material in this submission e.g., thesis/essay/project/assignment is entirely my/our own work except where duly acknowledged.

*I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams, or other material; including software and other electronic media in which intellectual property rights may reside.

*I have provided a complete bibliography of all works and sources used in the preparation of this submission.

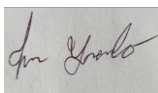
*I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offence.

Student Name (Printed) : Ivan Yaremko

Student Number : C00239239

Signature:

 Recoverable Signature

X 

Ivan Yaremko

Signed by: 6c456de4-a264-4906-bbdd-cd8e10592da5