CS31A0820 Data-Intensive Systems

Project Report, Part 1 - Ecommerce Book Store

Jesse Mustonen
Jesse Pasanen
Vili Raunola
Joona Heinikari

1 Introduction

Company called NordicBooks wants to build a web-based bookstore for the Nordic people. The business needs a webpage and a database connected to it in order to operate. The web store is planned to operate in Finland, Norway, and Sweden. NordicBooks plans to distribute the service to each of these countries. Distributing the services for each of the countries is done due to the difference in each of the countries' selections and prices. NordicBooks wants to provide books to the customers mainly in their native language which is why they opted for distributing the selection.

The functionalities that NordicBooks expects from the bookstore are registering for the service, viewing the contents of the store, the option to change the language to English, adding items to their shopping cart, and paying for the selected items. They also require functionality for adding new items to each country's databases. Since the service is planned to operate only in the previously mentioned countries, users from outside of these countries are not planned to access the site and an appropriate page is shown for the users which gives the option to select which country's site they wish to use instead.

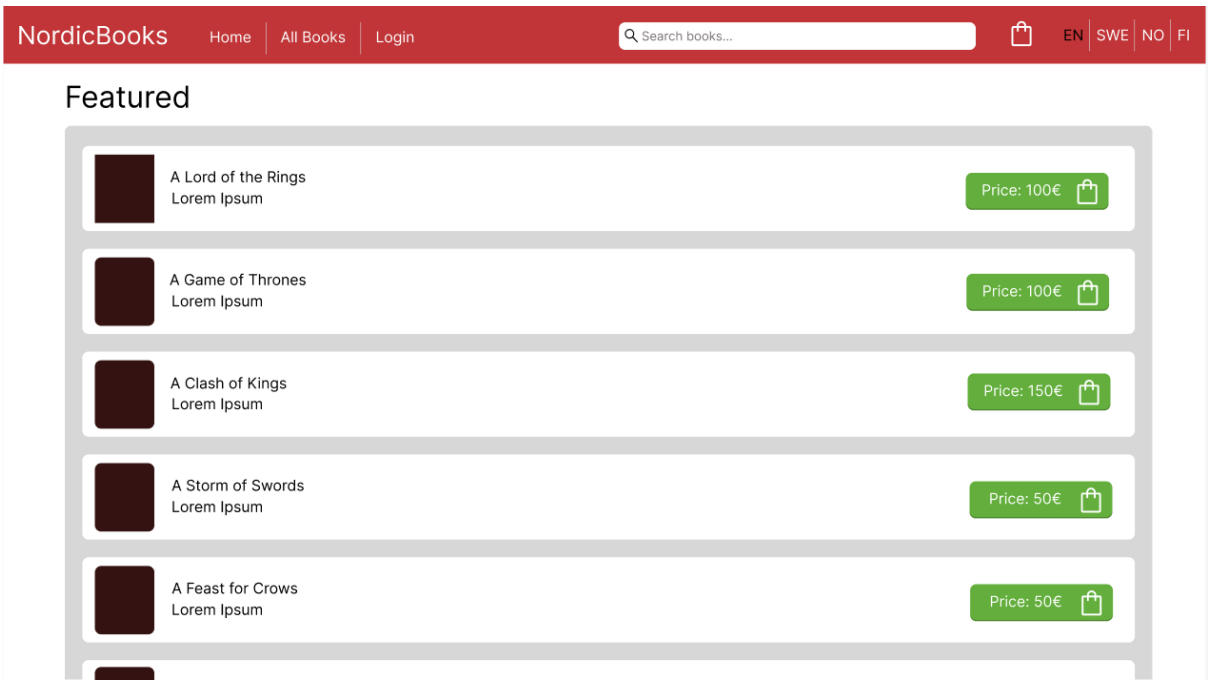
Distributing the databases will also help the company with the reliability, scalability, and performance of the website. Since the company desires to expand its business to other countries as well if the business is going to the plan can the distribution of the databases help with the scalability.

At the end of the project, there should be a new website created. The website should display e-books based on the selected country. For each country, a database is created which will store the e-book data items. This will result in a distributed database. Each of the databases will have a local collection for the country-specific e-books and a shared collection for the user accounts. The globally shared user account data is used so the users can use the same account even if they visit other instances of the website.

2 Description of the usage of the application and mock-ups of user interfaces

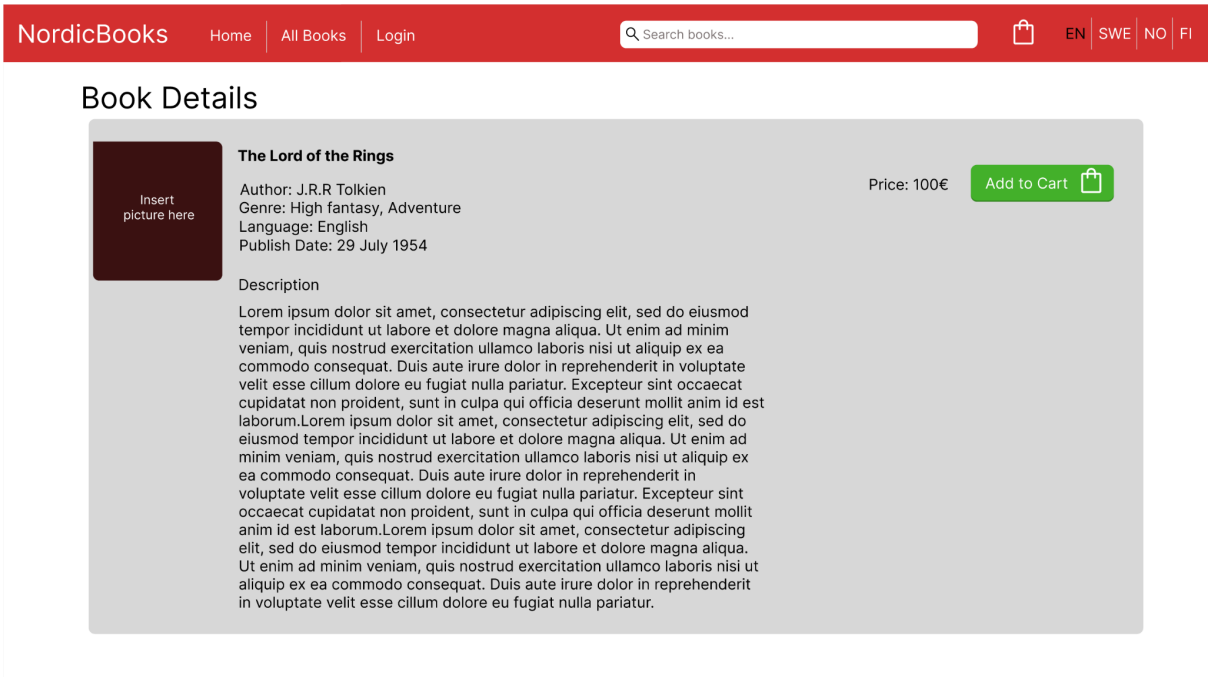
Web user interfaces, the functionality of the interfaces, and ways customers can navigate using the developed prototype are presented in this section.

The application is a basic eCommerce website where customers can buy books and download them in PDF format. The user interface of the application is straightforward. The toolbar is located at the top of the page and it consists of four main pages: the home page, all books page, the login/profile page, and the cart page. Search functionality is also located on the toolbar and by using it the customer can search books by their title. Customers can change the language of the application from the right side of the toolbar. Users can choose between English, Finnish, Swedish and Norwegian. The shopping bag icon on the right side of the search bar represents the customer's cart from where customers can see the contents of their cart and buy the books. On the login page, customers can log in to the application using their login information or they can register for a new account on a different basic registration page (not shown below).



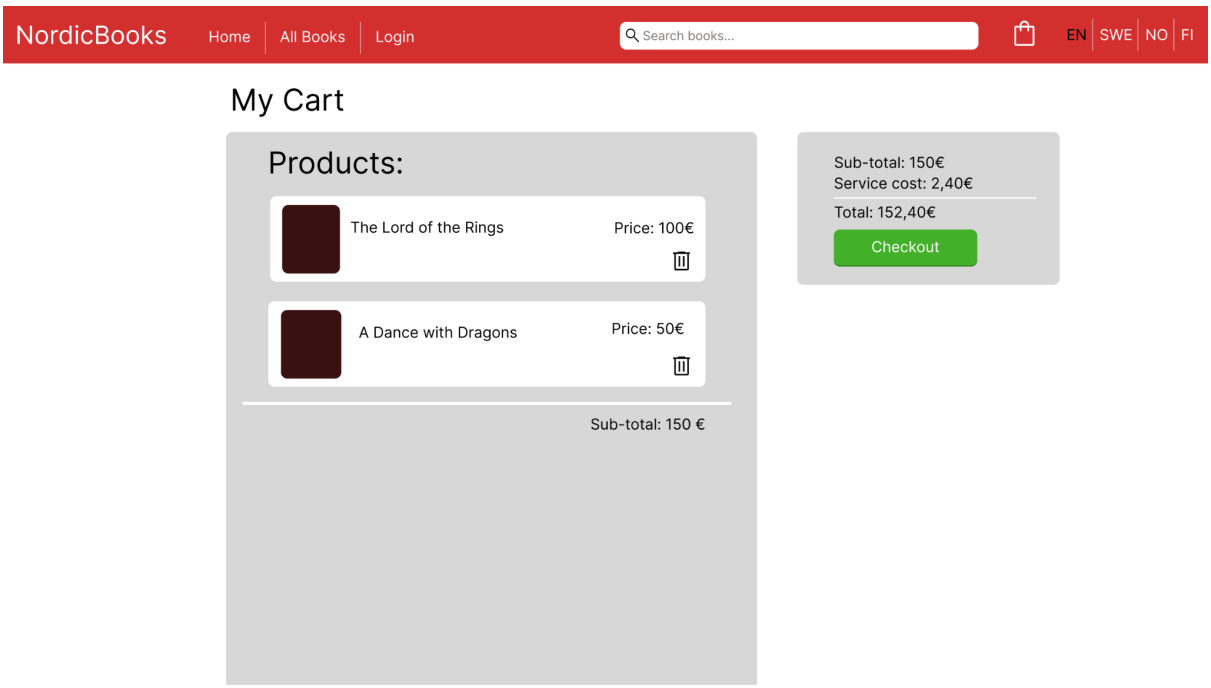
Picture 1

Picture 1 shows the home page of the application where all featured books can be seen. Featured books are in a list and customers can select a book from the list to see the details of the book. Featured books are based on the date that the books were added to the website. Customers can also add a book to their cart by pressing the green button related to that particular book.



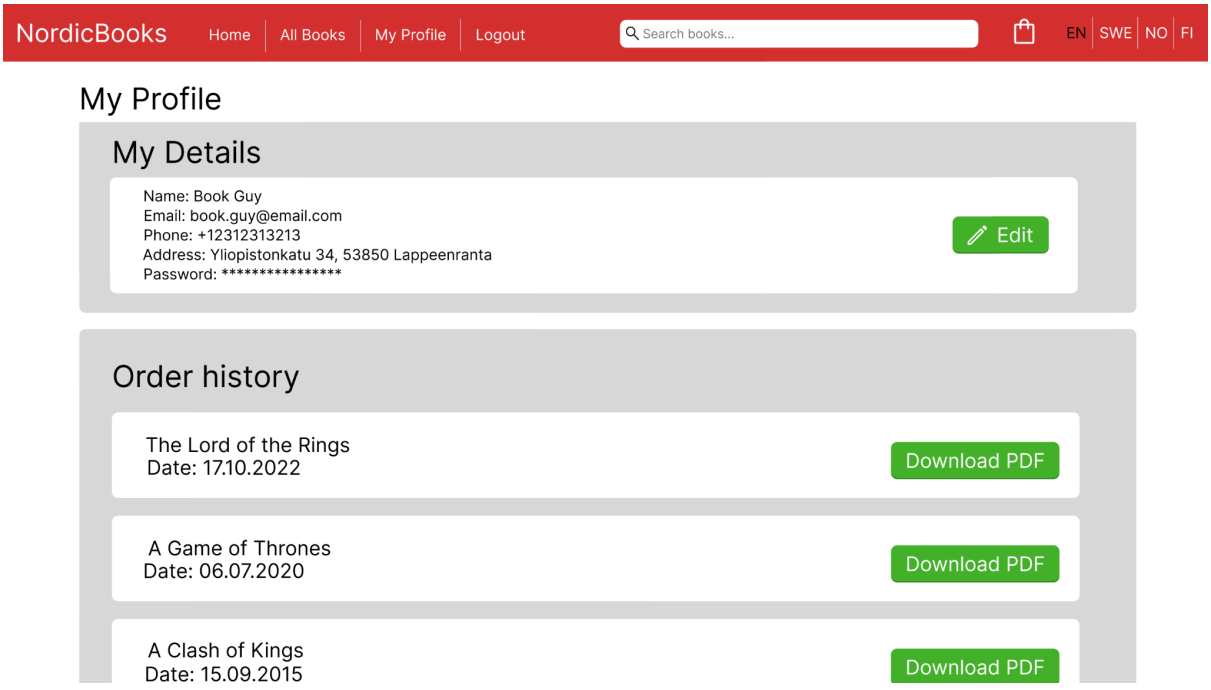
Picture 2

Picture 2 shows the detailed book page where the description of the book and other more detailed information can be found. Also from this page, customers can add the book to the cart.



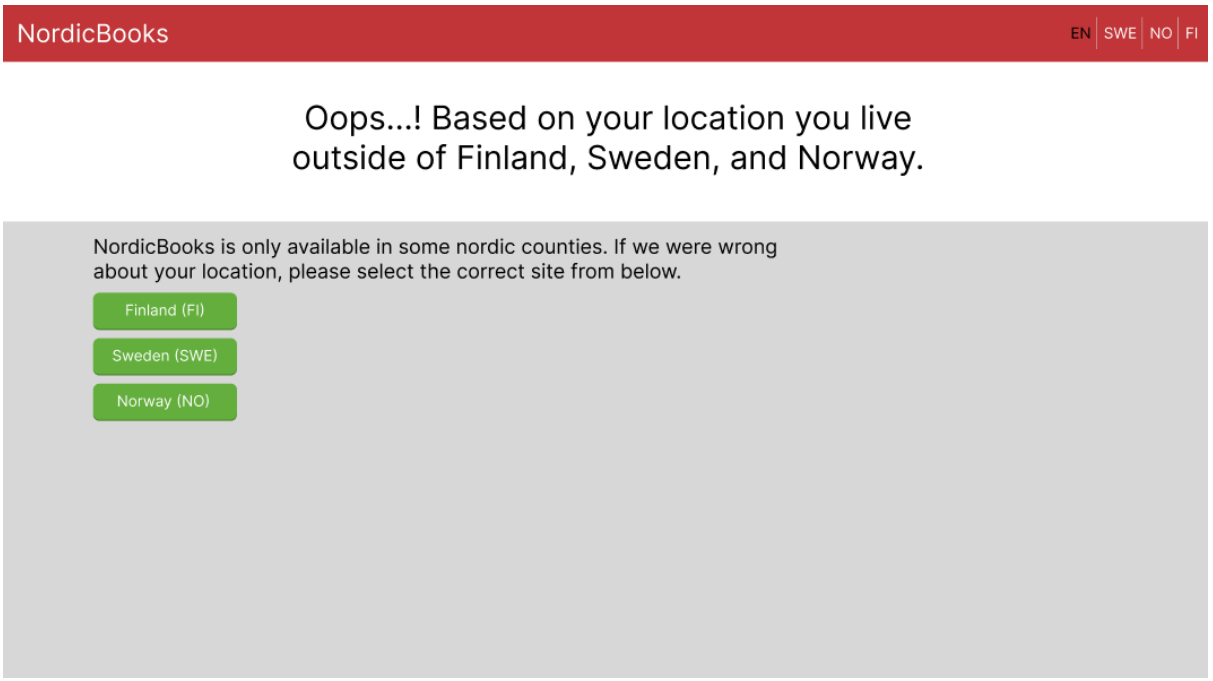
Picture 3

Picture 3 shows the cart page of the application. From this page, customers can buy the books they have added to their cart. On the left side of the page, all of the books that are currently in the cart can be seen and the user can remove them from the cart by pressing the trashcan icon. The total of the cart is shown on the right side of the screen and the customer can buy the books by pressing the “Buy now” button.



Picture 4

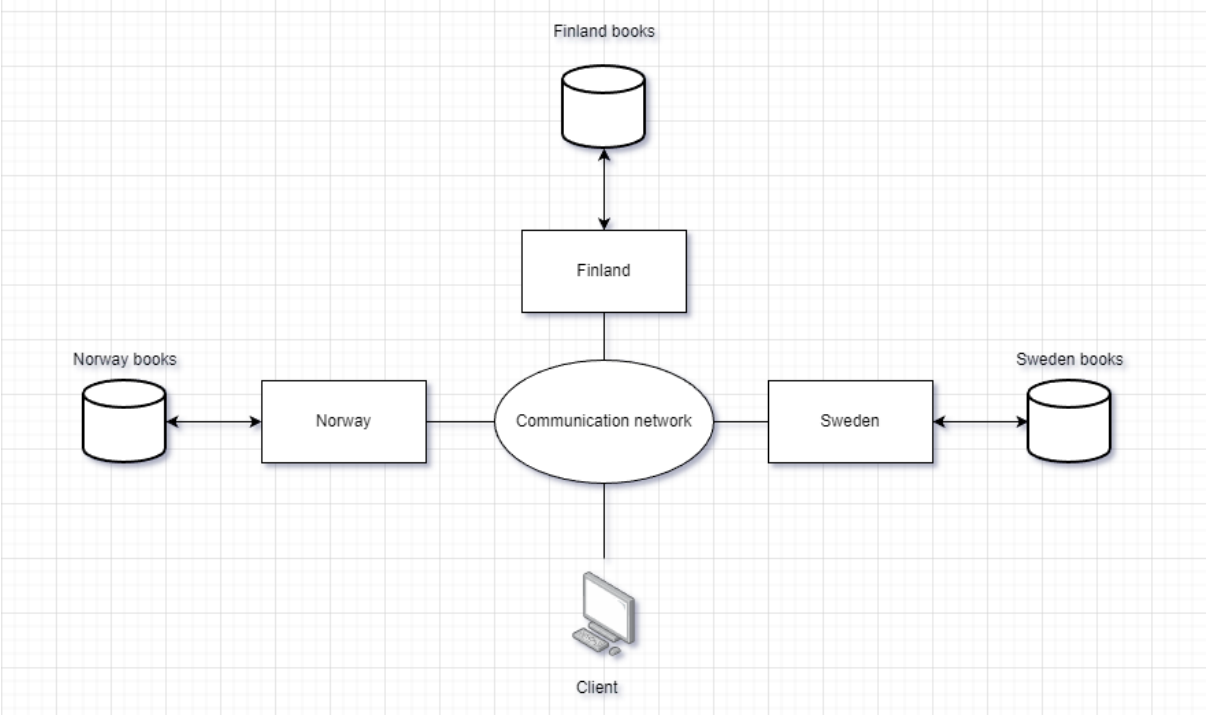
Picture 4 shows the customer's profile page where the customer can see their own detailed information, their order history, and download bought books in a pdf format. Customers can also edit their details from this page.



Picture 5

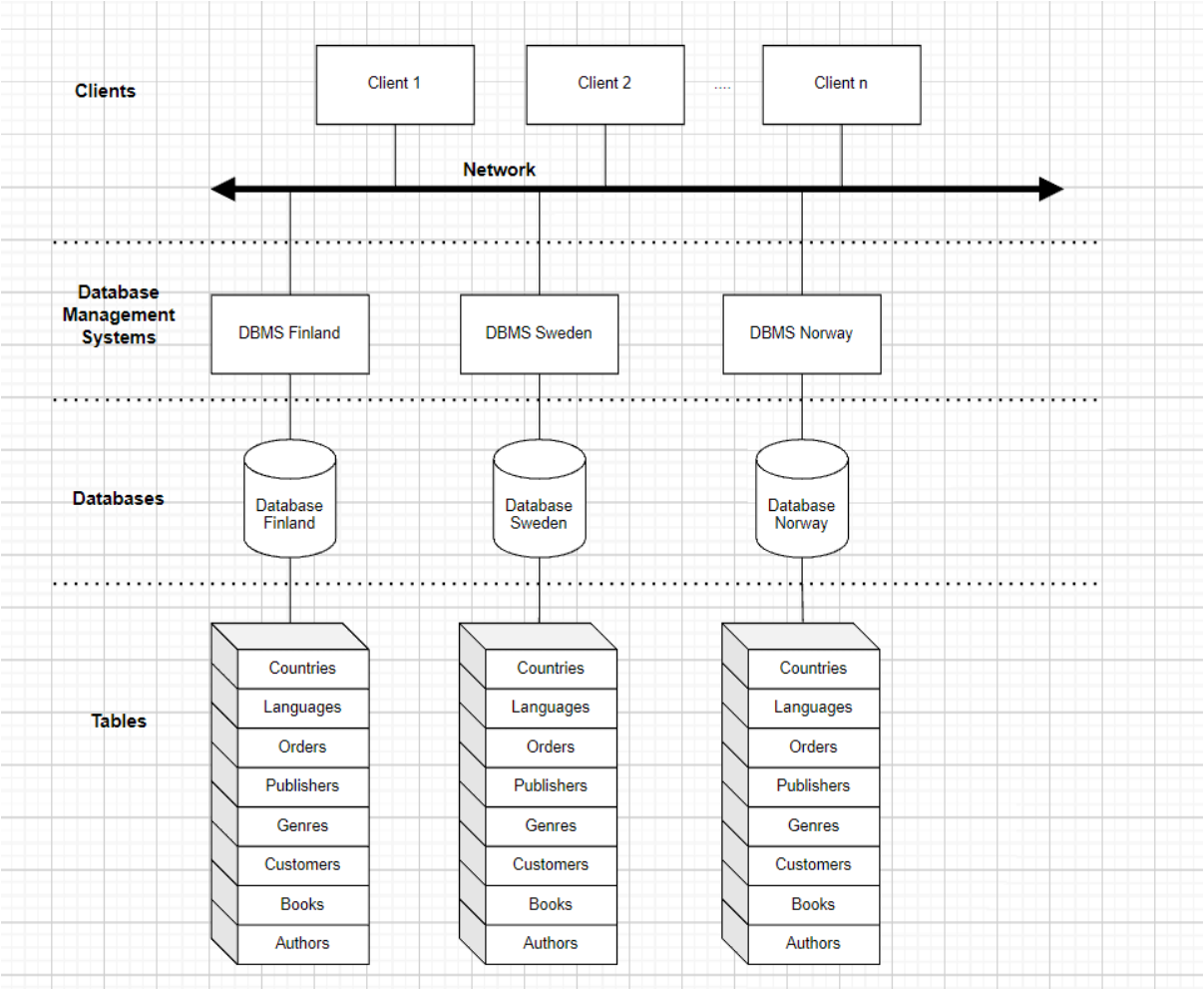
Picture 5 shows the page that will be shown to the customer if their current location is not one of the countries where NordicBook is operated. They can then choose what site they want to use from the available sites.

3 A high-level network model



Each geographical location has its own instance. When a user in a certain area accesses the platform, the communication network directs the user to the correct instance based on their location.

4 Distributed database system and justification



Communication network diagram

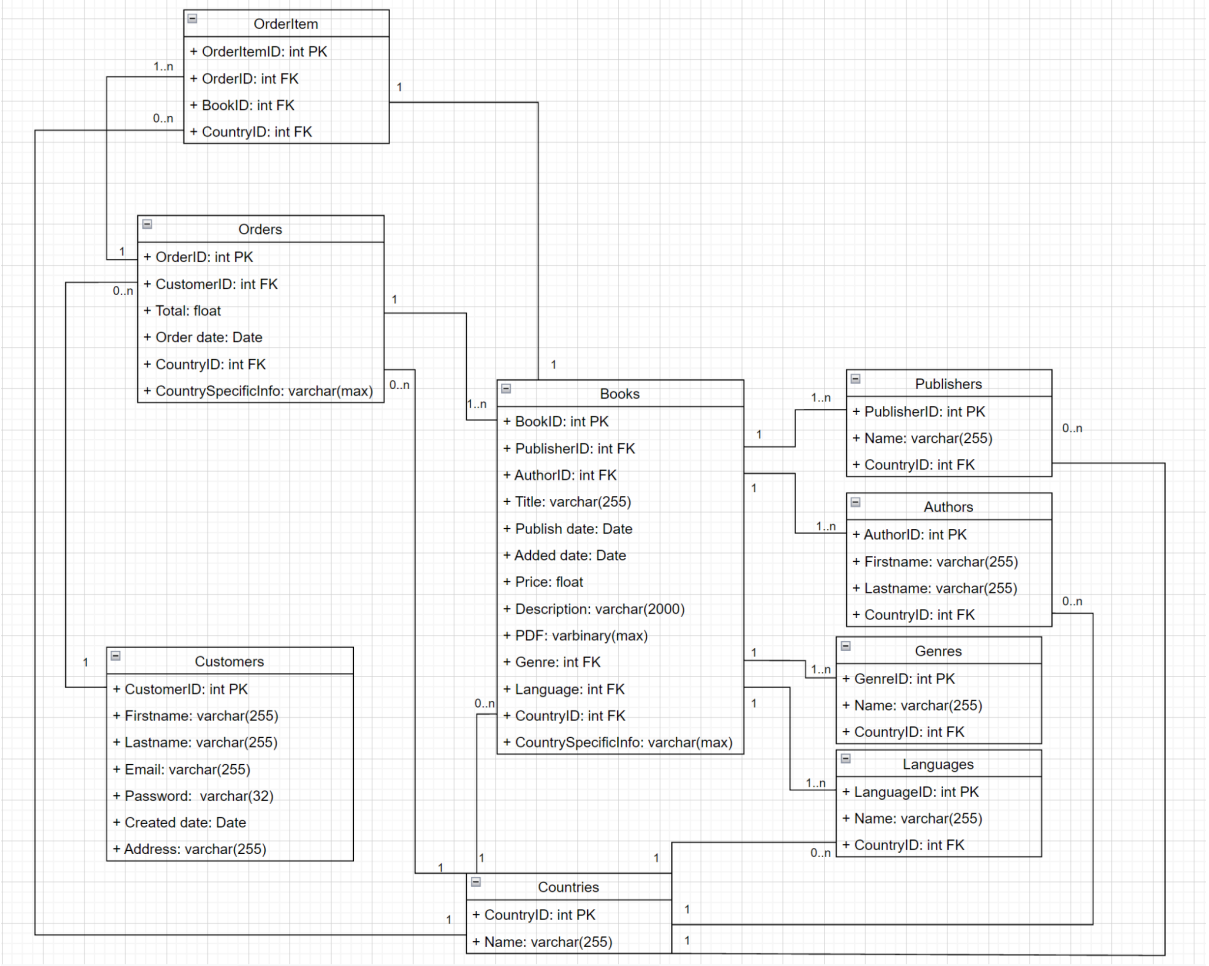
The decision to use a distributed database for the system is to distribute each country's book selection to its geographical location. Distributing the database will help with the system's performance, scalability, reliability, and availability. The performance benefits can be seen for example in the response times. Because each county's data is physically located in its own country, the data transmission times will be minimized. The response time benefits won't be as noticeable in this situation since all of the databases are located close to each other. Distributing the database will also help the company in the future if they want to spread its business to other countries. When the company wants to expand to other countries they only need to add a new local database for that country and connect the website to it. Distributing the database will affect the reliability and availability of the service. If one of the databases goes down, it will only affect that region thus not crashing the whole system for all of the

customers. It can be said that having a more reliable online service will help with customer satisfaction.

The database architecture of the project's database is done using client-server database management system architecture. It uses multiple server multiple client architecture where each database server handles its own database [1]. Because we use this database architecture each of our local schemas have the same fields. The global schema is all local schemas merged into one single schema [2]. But as our local schemas are the same our global schema is thus identical to local schemas.

5 Relational schemas

Global & Local Schemas



The global schema above is for the admin users only as each user is only connected to one of the local schemas based on their location. The local and global schemas are the same as

mentioned in the previous chapter. Every data from different local sites can be accessed using the global schema. Books can be added to all of the sites using the global schema at the same time if the database administrator (admin in this case) wants to. This helps to save time as then admins do not need to access each site individually. The “Customer” is a global table that is replicated on each site so that users can log in to each site with their existing account. Local schemas for each site are derived from this global schema.

Each site in the system will use this schema to store country-specific data. The instances are separated from each other using the “CountryID” as a foreign key. The “CountryID” is selected for this purpose since the databases are distributed based on their physical location. The foreign key is thus used in every table in the local schema to help in the identifying process. The tables are created for customers, orders, books, countries, publishers, authors, and languages. The customer table is used to store the data of each customer and is linked to the orders table which will be used to store the user’s order history. The book table uses the publishers, authors, genres, and languages tables to add more information on each of the books. The “CountrySpecificInfo“ field can be for example a country's unique value-added tax.

6 Fragmentation

6.1 Reasons and degree

Fragmentation will be done to increase performance and efficiency. The main guidelines for fragmentation are country-based. For example, every country has its own fragment that has region-specific e-books, other data related to them, and data related to orders. All available books for a certain region can be queried with global and region-specific fragments. As we have different databases for different sites every other table except the customer table is fragmented horizontally using “CountryId” to keep the performance better in a global scope.

Book												
BookId	PublisherId	AuthorId	Title	PublishDate	AddedDate	Price	Description	PDF	GenreId	LanguageId	CountryId	CountrySpecificInformation
1	1	1	Book1	Date	Date	1234	Book is good	BinaryData	1	1	1	20
2	1	1	Book2	Date	Date	123412	Book is bad	BinaryData	1	1	1	20
1	2	1	Book1Sweden	Date	Date	32131	Book is ...	BinaryData	1	1	2	40
2	2	2	Book4	Date	Date	123123	Book is good	BinaryData	2	1	2	40
1	2	2	Book5	Date	Date	123123	Book is good	BinaryData	2	2	3	30

Example global scope Books table instance

Customer						
CustomerID	Firstname	Lastname	Email	Password	Created date	Address
1	Matt	Smith	matt@email.com	**********	Date	Address1
2	Milly	Alcock	milly@email.com	**********	Date	Address2
3	Fabien	Frankel	fabien@email.com	**********	Date	Address3
4	Emma	D'Arcy	emma@email.com	**********	Date	Address4
5	Paddy	Considine	paddy@email.com	**********	Date	Address5

Example replicated Customer table on each site

Because the e-book's contents are also stored in this database, vertical fragmentation is done to separate the large binary file from the e-book information. Along with the e-book content, other information such as author and genre details are fragmented. There is also potential for the e-books to be fragmented horizontally by title due to the search function being done by book title.

6.1.1 Finland site fragmentation

Book												
BookId	PublisherId	AuthorId	Title	PublishDate	AddedDate	Price	Description	PDF	GenreId	LanguageId	CountryId	CountrySpecificInformation
1	1	1	Book1	Date	Date	1234	Book is good	BinaryData	1	1	1	20
2	1	1	Book2	Date	Date	123412	Book is bad	BinaryData	1	1	1	20

CountryId fragmentation

BookId	PDF
1	BinaryData
2	BinaryData

BookId	PublisherId	AuthorId	Title	PublishDate	AddedDate	Price	Description	GenreId	LanguageId	CountryId	CountrySpecificInformation
1	1	1	Book1	Date	Date	1234	Book is good	1	1	1	20
2	1	1	Book2	Date	Date	123412	Book is bad	1	1	1	20

Vertical fragmentation based on the e-book content(PDF column). Also, the table is fragmented horizontally by title but the example fragment is not shown here.

6.1.2 Sweden site fragmentation

Book												
BookId	PublisherId	AuthorId	Title	PublishDate	AddedDate	Price	Description	PDF	GenreId	LanguageId	CountryId	CountrySpecificInformation
1	2	1	Book1Sweden	Date	Date	32131	Book is ...	BinaryData	1	1	2	40
2	2	2	Book4	Date	Date	123123	Book is good	BinaryData	2	1	2	40

CountryId fragmentation

BookId	PDF
1	BinaryData
2	BinaryData

BookId	PublisherId	AuthorId	Title	PublishDate	AddedDate	Price	Description	GenreId	LanguageId	CountryId	CountrySpecificInformation
1	2	1	Book1Sweden	Date	Date	32131	Book is ...	1	1	2	40
2	2	2	Book4	Date	Date	123123	Book is good	2	1	2	40

Vertical fragmentation based on the book content(PDF column). Also, the table is fragmented horizontally by title but the example fragment is not shown here.

6.1.3 Norway site fragmentation

Book											
BookId	PublisherId	AuthorId	Title	PublishDate	AddedDate	Price	Description	PDF	GenreId	LanguageId	CountryId	CountrySpecificInformation
1	2	2	Book5	Date	Date	123123	Book is good	BinaryData	2	2	3	30

CountryId fragmentation

BookId	PDF
1	BinaryData

BookId	PublisherId	AuthorId	Title	PublishDate	AddedDate	Price	Description	GenreId	LanguageId	CountryId	CountrySpecificInformation
1	2	2	Book5	Date	Date	123123	Book is good	2	2	3	30

Vertical fragmentation based on the book content(PDF column). Also, the table is fragmented horizontally by title but the example fragment is not shown here.

6.2 Correctness rules

Completeness

The fragments should be complete to prevent data loss. To ensure that there is no data loss, all of the data must be found on at least one of the fragments. For example, every book or customer will have a region attached to them.

Reconstruction

To ensure that the tables can be reconstructed, the unique identifier for a certain row must be available on every vertical fragment.

Disjointness

To ensure that the data can not be found on multiple fragments, it should not satisfy more than one predicate. Since the data is fragmented mainly by region, there should not be much trouble with the fragments not being disjoint.

6.3 Data replication strategy

Each country has its own database and data is replicated on each site only locally. This decision is made for fast data access and shorter query execution times. Data is also replicated locally, for example, the books table can have different books depending on the site it is accessed from i.e. the Norwegian AllBooks site doesn't have all the Finnish books that are available in Finland.

The only special case is the "Customers" table which stores the user data for all sites globally. User information is replicated to all local customer tables and users can log in to any of the three available sites in Finland, Norway, and Sweden with their already existing account.

6.4 Cost model

CPU used in the cost model is Intel i5-10600K which is a 4.10 GHz 8-core processor and the hard drive used is an average hard drive given in the "Cost information" section. The cost model is calculated for downloading a pdf for a book as it is the most costly query. The cost model will be calculated using the following formula

*Total time = T_{CPU} * #insts + $T_{I/O}$ *#I/Os+ T_{MSG} *#msgs + T_{TR} * #bytes*

- T_{CPU} is the time of a CPU instruction.

- $T_{I/O}$ is the time of a disk I/O.
- T_{MSG} is the fixed time of initiating and receiving a message.
- T_{TR} is the time it takes to transmit a data unit from one site to another. [3]

The cost model tells us how long a query will take. T_{CPU} is calculated using the following calculation: T_{CPU} = *CPI x Clock time ==> CPI / frequency*. CPI number in these calculations is CPU clock speed/millions of instructions per second (MIPS). The CPU used in these calculations is Intel i5-10600K whose CPU clock speed is 4.1 GHz and it can do 375 000 MIPS as it is an 8-core processor. MIPS value provided in the moodle “Cost information” section was used to approximate our MIPS value. Thus the CPI of the CPU is 4 100 000/ 375 000 = 10.93. We assume that one read is equal to one instruction and that we have 1000 reads in our example. Then our T_{CPU} * *#insts* = (10.93 * 1 000) / 4 100 000 = 0.002669s = 2.669ms . [3]

$T_{I/O}$ *#I/Os* is calculated using the average seek time and average speed of the hard drive (Input-Output Operations Per Second). The average seek time of the $T_{I/O}$ is 1/ IOPS. The average SSD speed is 100 000 IOPS for a read operation and 16 000 IOPS for a write operation. The $T_{I/O}$ value for this cost model is 1 / 1 00 000 = 0.00001s = 0.01 ms for reading and 1 / 16 000 = 0000.6 ms = 0.06ms for writing data to the drive as these are the average read and write speeds for the SSD. [3]

The average network speed (T_{TR}) is 168 Mbps (21 MB/s) in Norway, 135 Mbps (16.9 MB/s) in Finland, and 175 Mbps (21.9 MB/s) in Sweden [4]. Based on these let's assume the Average speed is around 20 MB/s. T_{TR} * *#bytes* is (3 MB) / (20 MB/s) = 0.15s = 150ms. The size of one data unit (*#bytes*) is 3MB as the average book size is 300 pages and we assume that one page is 10kt [5].

T_{MSG} is the fixed time it takes to initiate and receive messages. We assume that T_{MSG} is 0.002s = 2ms. The message amount (*#msgs*) is estimated to be 384 as the maximum request length of HTTP GET request is 8KB as 3MB is 3072KB → *#msgs* = 3072KB / 8KB = 384 [6].

Total time for this query:

T_{CPU} * *#insts* + $T_{I/O}$ *#I/Os*+ T_{MSG} *#msgs* + T_{TR} * *#bytes*

- T_{CPU} * *#insts* = 2.669ms = 0.002669s
- $T_{I/O}$ *#I/Os* = 0.06ms or 0.01ms = 0.00006s or 0.00001s
- T_{MSG} *#msgs* = 2ms * 384 = 768ms = 0.768s
- T_{TR} * *#bytes* = 150ms = 0.15s

For read operation total time is 2.669ms + 0.01ms + 768ms + 150ms = 920.68ms = 0.921s

7 Database integration and access control

Since the project is started from scratch there is no need for database integration from already existing data structures. Customers cannot create an order if they are not logged in. This access is not related to database management but the client-side authorization will handle this. More about the specific database access control for different users can be found in the security section of the document.

8 Query Processing

8.1 Customer Registration

```
INSERT INTO customers
(firstname, lastname, email, password, createdDate, address, countryId)
VALUES
('firstname', 'lastname', 'firstname.lastname@email.com','Password1234', GETDATE(), 'Address', 1)
```

Using the registration page, users can fill out their personal details and register to the site. These details are then sent to the database.

8.2 Customer Login

```
SELECT firstname, lastname, email, password, address, id
FROM Customers
WHERE email = @givenEmail and password = @givenPassword
```

After registering, users can log in using their email and password. Login credentials are compared to those stored in the database.

8.3 All Books

```
SELECT bk.id, bk.title, bk.price, a.firstname, a.lastname
FROM Books as bk
INNER JOIN Authors as a
ON bk.authorId = a.id
```

All Books page has a list of all the books in the database. The whole book table requested from the database.

8.4 Featured Books

```
SELECT  bk.id, bk.title, bk.price, a.firstname, a.lastname
FROM Books as bk
INNER JOIN Authors as a
ON bk.authorId = a.id
ORDER BY addedDate DESC
```

Features Books shows a list of 20 books that have recently been added to the website/database. The books are queried from the books table based on the date they were added.

8.5 Search Books

```
SELECT  bk.id, bk.title, bk.price, a.firstname, a.lastname
FROM Books as bk
INNER JOIN Authors as a
ON bk.authorId = a.id
WHERE bk.title = @bookTitle
```

Using the search bar, books are queried from the books table based on the title of the books.

8.6 Books Details

```
SELECT
    bk.id,
    bk.title,
    bk.price,
    a.firstname,
    a.lastname,
    bk.publishDate,
    bk.description,
    g.name as Genre,
    lang.name as Language
FROM Books as bk
INNER JOIN Authors as a
ON bk.authorId = a.id
INNER JOIN Language as lang
ON bk.language = lang.id
INNER JOIN Genre as g
ON bk.language = g.id
WHERE bk.id = @bookId
```

The book details page shows the details of a single book. The book is queried from the books table based on book ID.

8.7 Customer Profile Details

```
SELECT firstname, lastname, email, password, address
FROM Customers
WHERE id = @customerId
```

The customer profile page displays the customer's personal details. The details are queried from the customer table based on the customer's ID.

8.8 Edit Profile Details

```
UPDATE Customers
SET
    password = @newPassword,
    address = @newAddress,
    email = @newEmail,
    firstname = @newFirstname,
    lastname = @newLastname
WHERE id = @customerId
```

The customer can edit their details from the profile page. The details are edited by updating the customer table based on the customer's ID.

8.9 User Order History

```
SELECT
    bk.title AS BookTitle,
    bk.PDF AS BookPDF,
    o.orderDate AS OrderDate,
    o.total AS Total
FROM Orders AS o
INNER JOIN OrderItem AS oi
ON o.id = oi.orderId
INNER JOIN Books AS bk
ON oi.bookId = bk.id
WHERE o.customerId = @customerId
```

The customer's profile page also has the order history of the customer. The order history is queried from the orders table based on the customerID.

8.10 Download Book

```
SELECT title, PDF
FROM Books
WHERE id = @bookId
```

The customer can download books based on the books they have in their order history. The PDF file is queried from the books table based on the bookID.

8.11 Create an order

```
INSERT INTO Orders
(customerId, total, orderDate, countryId)
VALUES
(@customerId, @orderTotal, @orderDate, @countryId)
```

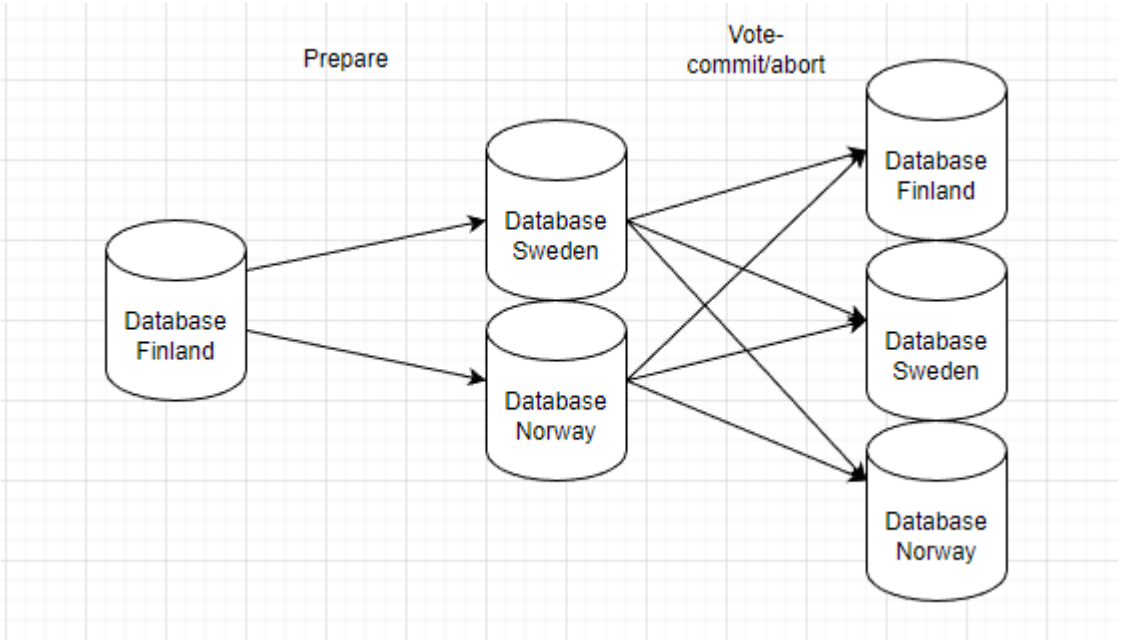
When the customer creates new order the order information is added to the Orders table and each has its own unique orderID and customerId foreign key

```
INSERT INTO OrderItem
(orderId, bookId, countryId)
VALUES
(@orderId, @bookId, @countryId)
```

9 Concurrency control and deadlocks

This project will use a two-phase commit (2PC) for its concurrency control. It will maintain the integrity of the database even when multiple users are accessing the database simultaneously. More specifically, a distributed version of 2PC will be used. This will help to maintain the atomicity of the operations meaning that whenever a write operation is done it will be completed at all of the sites or if it is needed to abort it will abort on all of the sites. A strong consistency is desired between the user tables meaning that all of the copies of the tables are updated at the same time when the original update transaction is run. To achieve this eager distribution control will be used with the 2PC. The main idea behind eager distributed control is that whenever a change is made on a site it will propagate the made changes to all of the other sites before commitment. The coordinator of this process will be the site where the original write operation is executed at. [3] [7]

When a new user is created for example in the Finnish database, that database becomes the coordinator. It will contact the other databases with a “prepare” message. If the other sites receive, in this case, the Swedish and Norwegian sites, the “prepare” message they can vote on the decision of whether they can commit or do they need to abort the execution of the operation. They send their votes to the other sites as well as themselves. The participants and the coordinator will wait for the other sites' votes and will make the decision to either commit or abort independently. This operation is shown in the graph below. [3] [7]



A deadlock might arise if a site is already using the same resource as the global commitment is requesting to use. In this case, the site can abort the global transaction process by sending a vote-abort to other sites. Aborting the transaction will cancel it on all of the other sites. The coordinator site that requested the global transaction can retry the process at a later time to check if the other site has released its lock. [3] [7]

10 Security

As with any site that works within the European Union, GDPR has to be followed especially when the site deals with customers' personal information such as home address. Not only are the passwords encrypted, but the home address of the user is encrypted as well so that the customer's name can't be connected to a home address in case of a malicious actor. The encryption is done using SHA-256 in both cases. Customer payment details will not be stored within the databases and payment is done via a third-party service.

The site uses HTTPS, or SSL/TLS protocol, for encryption and authentication. This makes sure that sensitive data like login credentials as well as payment details and other personal details will be transmitted securely [8].

Multiple users can access the database and different permissions are granted to each user. The admin_user has access to all of the tables in all of the sites and they can read, write, update and delete data from all of the tables in each of the sites. Each site has its own local_user that can only access the current site's specific database. The local_user can only read from all of the tables on the connected site but they have limited write and update privileges to different tables. Local_user does not have access to delete data from the tables.

References

[1] ‘Distributed DBMS - Database Environments’. https://www.tutorialspoint.com/distributed_dbms/distributed_dbms_database_environments.htm (accessed Oct. 27, 2022).

[2] J. Parsons, ‘Effects of Local Versus Global Schema Diagrams on Verification and Communication in Conceptual Data Modeling’, *J. Manag. Inf. Syst.*, vol. 19, no. 3, pp. 155–183, 2002, doi: 10.1080/07421222.2002.11045730.

[3] M. Tamer. Özsu and Patrick. Valduriez, *Principles of Distributed Database Systems*, 4th ed. 2020. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-26253-2.

[4] ‘Internet Speeds by Country 2022’. https://worldpopulationreview.com/country-rankings/internet-speeds-by-country (accessed Oct. 27, 2022).

[5] G. Press, ‘How Many Pages Should a Book Be to Publish?’, *Gatekeeper Press*, Mar. 17, 2020. https://gatekeeperpress.com/how-many-pages-should-a-book-be-to-publish/ (accessed Oct. 27, 2022).

[6] BalusC, ‘Answer to “Maximum length of HTTP GET request”’, *Stack Overflow*, Apr. 17, 2010. https://stackoverflow.com/a/2659995 (accessed Oct. 27, 2022).

[7] Jiri Musto, D.Sc., ‘CS31A0820 Data-Intensive Systems Lecture slides 6 & 5’. LUT.

[8] ‘What is HTTPS?’, *SSL.com*. https://www.ssl.com/faqs/what-is-https/ (accessed Oct. 27, 2022).