Edinburgh Napier
UNIVERSITY

SET10101 Software Architecture

Coursework

Hari Kharel – 40312824

Contents

List of Figures

1 Proposal

This proposal is for the development of the distributed store management system for retail branches of DE-Store. In this proposal, the requirements of the platform are evaluated and multiple architectures for designing and developing the platform are also discussed.

As per the requirements, the following features are considered when recommending the development of architectural style. Price Control, Inventory Control, Loyalty Card, Finance Approval, Report and Analysis. All these requirements have CRUD operations, Create, Read, Update & Delete. Additionally, since the platform is distributed, a centralised machine is required to host the server which the clients can connect to and request a CRUD operation.

With this considered, the following two architectures may be well suited for the platform, Microservices and three-tier Architecture. The benefits and the drawbacks of each of the systems are detailed in the following subsections.

1.1 Microservices

Microweb services, coined initially in 2005 by Dr Peter Rodgers during a conference on cloud computing in 2005 is a relatively new software architecture to emerge. In 2011, it was officially called microservices with early pioneers being Netflix and Amazon. (Mauersberge, 2017)

The main philosophy of microservices is developing complex applications using small and individual components that do one specific thing which can be deployed individually, like handling authentication or managing inventory. This avoids the coupled nature of a monolithic application.

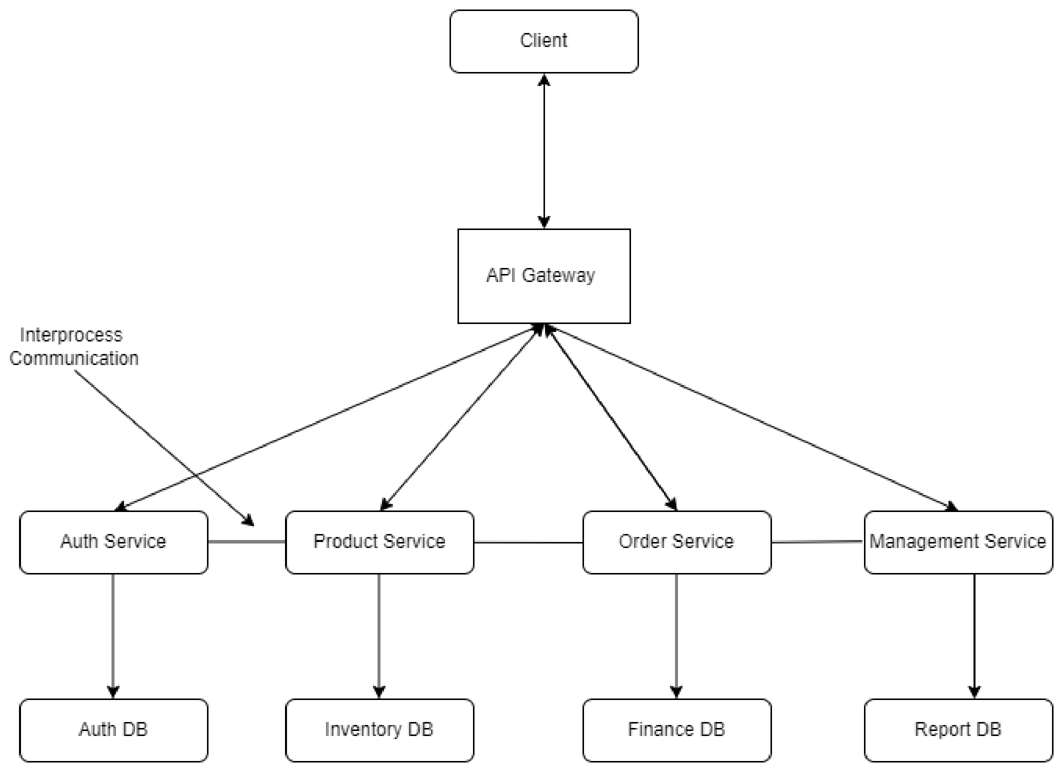


Figure 1 Microservices Architecture overview

In figure 1 above, a flowchart of a simple microservices architecture is shown. A client connects to a server proxy which then redirects the request to the correct service. For example, In DE-Store, a manager wishes to log in, the request will initially be sent to the server proxy which is then redirected to the correct service.

1.1.1 Benefits of microservices

Scalability

Unlike some of the other architectures, microservices allows for the scaling of individual services. In the context of DE-Store, the inventory service might be the busiest as all the stores are constantly checking inventory. In an event of higher capacity requirements, the hardware for the inventory service can be upgraded individually and precious resources are not wasted on upgrading the entire server-side.

Additionally, when requirements grow, vertically scaling a service can become very expensive. To combat this, microservices can also be scaled horizontally where more machines are added, rather than typical vertical scaling by increasing the power of a single machine.

Resiliency

Another key benefit of microservices is that it is very resilient to failures as failures are isolated within a specific service. In DE-Store’s case, if the hardware running the Reporting service failed, all the other services will continue to work as normal. The fault is isolated to the reporting service thus rendering it the only service that is not operational. Additionally, it also makes identification of fault simpler as it can be narrowed down to 1 specific service.

Rapid Development

Rapid development is perhaps one of the most important key features of microservices. With software becoming increasingly integrated and autonomous, it can take an extremely long time to develop. However, with microservices, features can be deployed as they are developed. In DE-Store’s scenario, while the Finance service is being developed, all the services can be deployed thus getting to marker much quicker by utilizing the Minimum Viable Product (MVC) philosophy.

1.1.2 Microservices drawbacks

Initial Development Overhead

For DE-Store, one of the drawbacks of microservices is that it could be complex overhead to implement the infrastructure initially. Although once it's up, it makes development rapid, configuring different services, proxies, gateways may increase the time needed for architectural development as well as the cost associated with the initial development.

Complexity & Cost

The complexity of the architecture is directly proportional to the number of services needed. This means that as the DE-Store grows, the system will also become more complex and the cost more for maintenance will also increase. Furthermore, integration testing the microservices architectures is more complicated as different services are deployed on different machines and may run in a different environment.

1.1.3 Three Tier Architecture

Another massively popular architecture that can be utilized for developing DE-Store is a simple 3 tier Client-Server-Database architecture. The tiers can often be called, the presentation tier – the front end of the platform, the application tier – where the calculations and data is processed and finally the data tier – where the data is stored and retrieved.

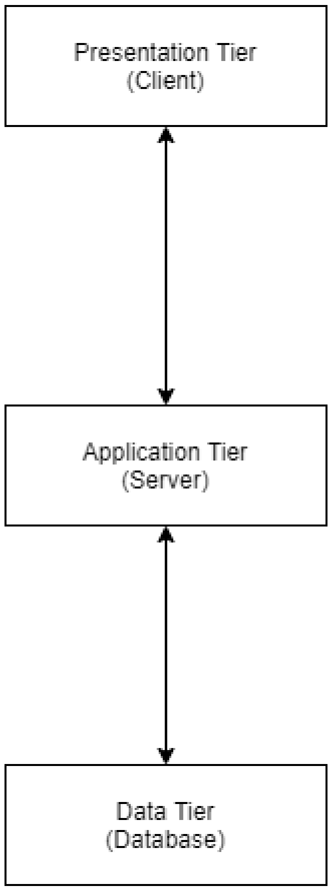


Figure 2 Three Tier Architecture

In Figure 2 above, the typical representation of the tiered architecture is shown. The presentation tier communicates with the application tier and the application tier communicates with the data tier.

1.1.4 Three Tier Advantages

Security

Like microservices, the presentation tier does not directly communicate with the data tier, which improves security as only permitted CRUD operations are carried out via the application tier. Competed to a 2 tiered or an MVC architecture, where the presentation tier could access the raw data, this architecture prevents it as the flow is linear and any access to data must pass through the application tier.

Simplicity

Compared to microservices, the three-tier architecture is much simpler to design and implement initially. All that’s needed for a basic system is a client host, server host and database host. While not recommended for security risks, all of them can be run on a single machine. Additionally, it also makes testing a lot simpler as the entire system is on its sandbox and can be tested as a complete system.

Modularity

Although not as modular as microservices architecture, the 3-tier architecture still separates the client, server, and database. This allows for plug and play style. For example, if DE-Store wanted to develop a mobile app, all that needs to be developed is a new client for mobile, provided that the server uses a universal method of communication, such as REST.

1.1.5 Three Tier Disadvantages

Extensibility

As DE-Store wishes to expand the system and make it adaptable, microservices architecture is far superior to the three-tier one. When new functionality, such as delivery management is added, with a 3-tier architecture, it must be implemented in a way that allows for the features to work efficiently as well as not break anything else in the application tier. As the complexity grows, so will the overhead for extending the system. In contrast, with microservices, a new server can be spun up just to handle the newly implemented feature. Any changes mean the entire system has to be rebuilt and re-run tests.

Independent Scaling

In 3-tier architecture, all the business logic is contained within a single application tier. If one service, such as inventory lookup needs more resources, it is not possible to scale that independently. Doing so also scales aspects of the application that are not needed thus wasting resources.

1.1.6 Architectural review

With the advantages and disadvantages considered for both microservices and three-tiered architecture, it is evident that microservices architecture makes the most sense for DE-Store. Although the initial infrastructure setup may take longer, it makes extending and scaling DE-Store as needed in the future. As such, a full system prototype will be designed using the microservices architecture.

2 Prototype Design

In this section, the design of the microservices will be discussed and evaluated. It will detail the design pattern of the individual services, feature class design as well as inter-process communication design. As per the requirements, the prototype for the following services will be designed; Auth Service, Price Control Service, Inventory Control Service, Loyalty Card Service, Finance Service, Reports & Analysis service.

To design the services, a dynamic controller-service pattern will be used. The reason behind this decision is that it allows to manage functions for incoming requests and functions that carry out the actual requested operation. This also separates concerns of each of the classes, making it easier to understand the codebase at a later stage.

The design and development of the prototype will be based on Node.js & Nest.js for backend, React.js for front end, MongoDB for the database and finally, Docker for deployment.

2.1 Auth Microservice

The Auth microservice for DE-Store handles the authentication of staff and managers. It allows for each staff to log in to the DE-Store portal and carry out operations.

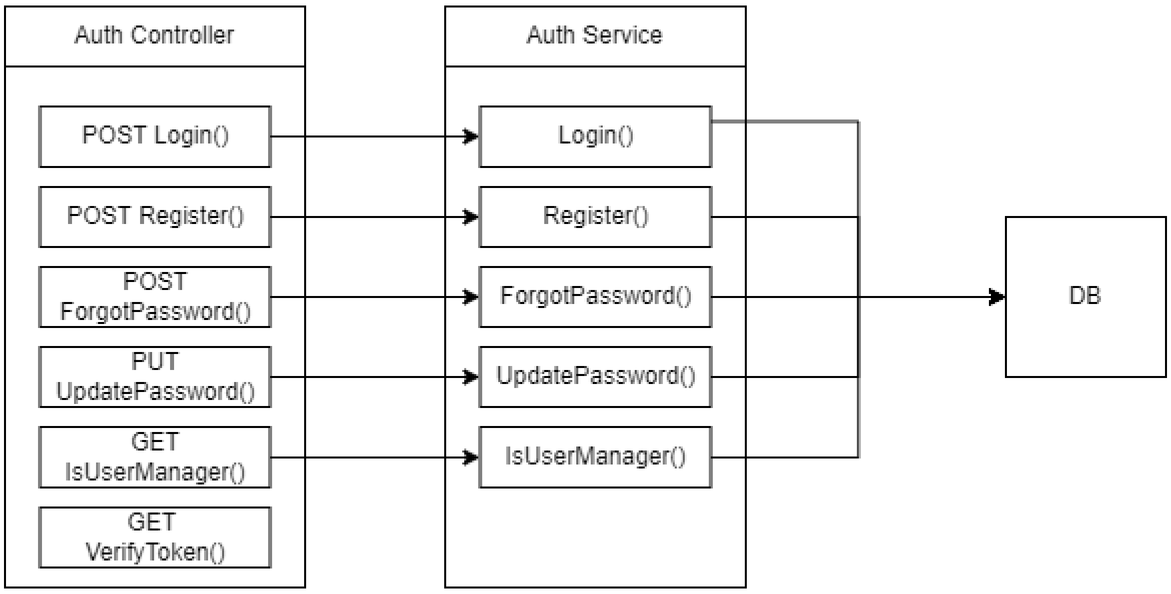


Figure 3 Auth Micro Service

The auth service also handles verifying the users JWT token, as requested by the other services. This is done via the usage of custom route annotations, called guards. The usage of guards allows the code to be reused and the service function is only called if the request gets authorized by the guard.

2.2 Product Microservice

The product microservice handles everything that has to do with a product, from creation, updating and deleting. The product service is divided into two controllers and internal services, as shown in Figure 4. This is done to separate concerns and address the requirements. Each of the routes in the product service is protected.
As mentioned previously, the services are protected using guards, the Auth guard checks if the user is authorized to make the request and the Manager guard checks if the user has the permission to manage products. Some routes, like deleting or creating a product should not be accessible by store assistants.

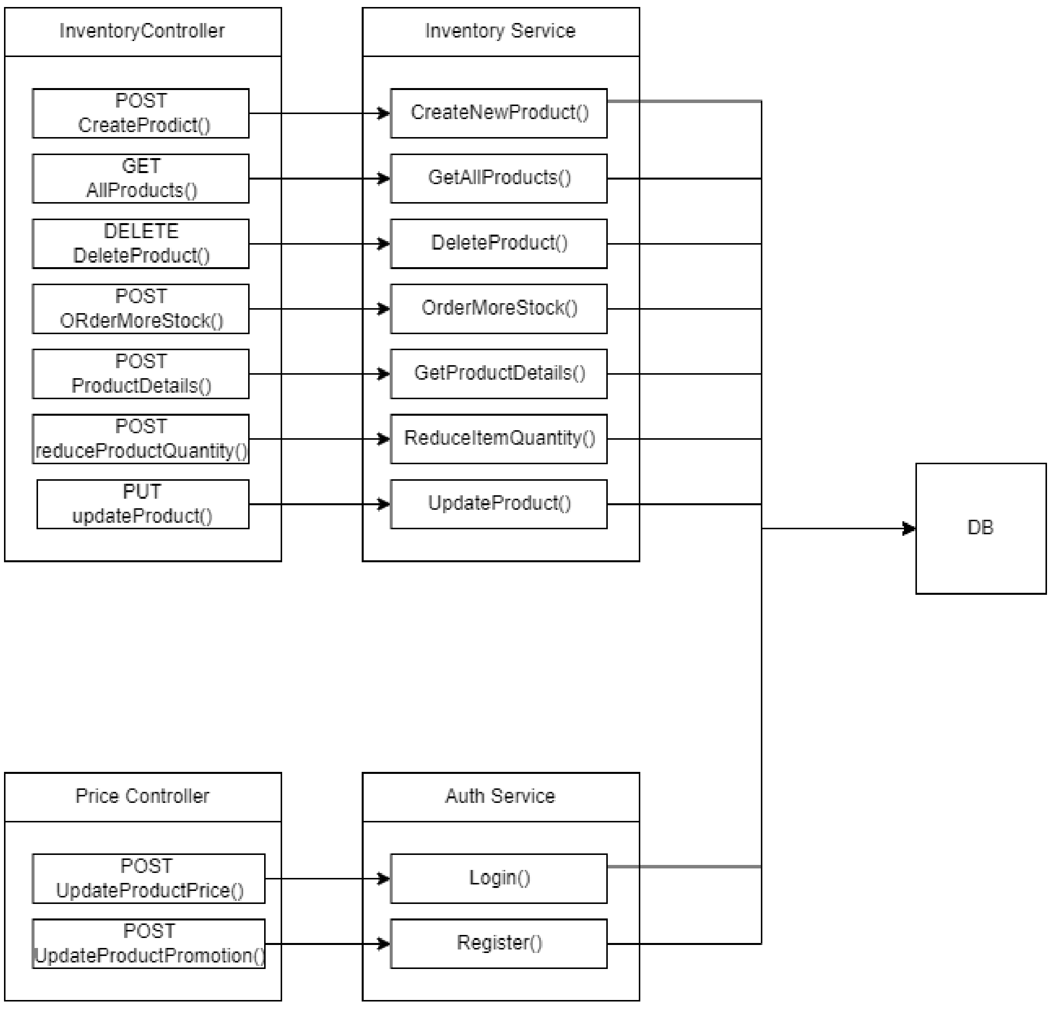


Figure 4 - Product Microservice

2.3 Order Microservice

The order microservice is responsible for handling orders and finance decisions by the store manager. Since DE-Store is not customer facing, the products are ordered by retail staff, on behalf of the customers. After placing an order, if the product becomes out of stock, it will automatically order more inventory from the central inventory system. Similarly, if the product becomes low on stock, a message is sent to the managers' inbox from where they can order more. This is done by using interposes communication via the HTTP protocol, between the order microservice and the management microservice. Like the other microservices, this service also makes use of guards for authentication and authorisation. In Figure 5 below, the functions and the architecture of the order microservice are outlined. Each HTTP function, like POST and GET, is mapped to a corresponding function in the service class, which carries out the requested operation.

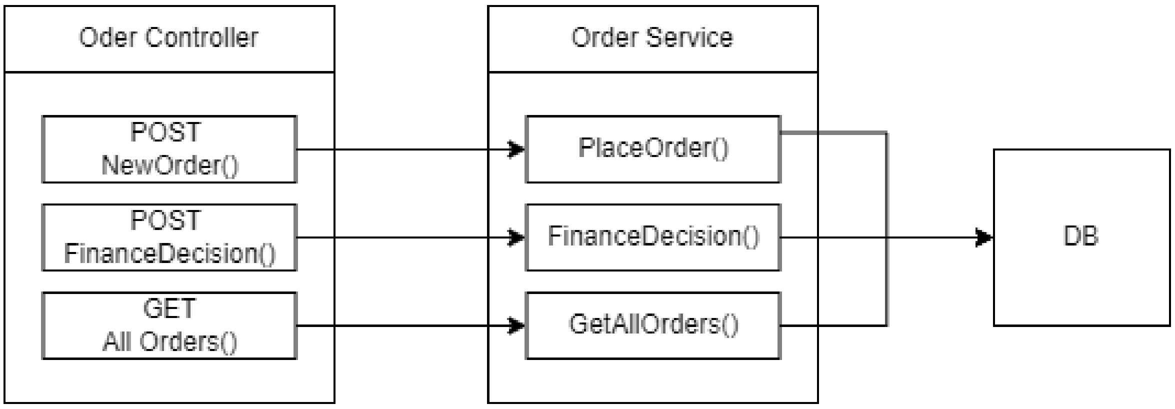
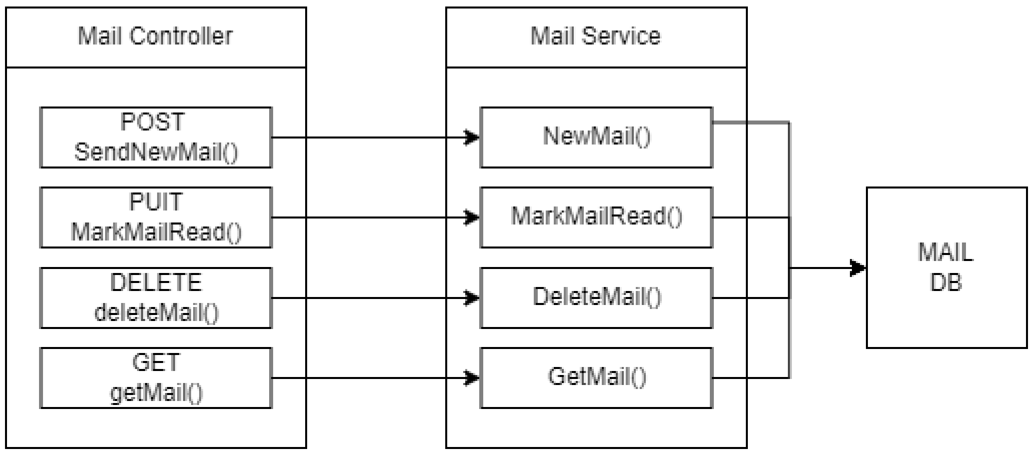


Figure 5 Oder Microservice

2.4 Management Microservice

Like the product microservice, the management microservice features 2 different controllers that perform different functions, the Mail controller and the reporting controller, as shown in Figure 6. Ideally, the main controller could be its own microservice, it is not necessary. Existing services like SendGrid should be utilised for sending mail for reliability. For this implementation, the mail service is an internal DE-Store solution that uses a purpose-built database, rather than email.

The reporting controller handles store performance requests and recent sales data requests. Due to DE-Store being just a prototype, the performance metrics are mocked. A new DB should be created to so store the metrics and the accounting data.



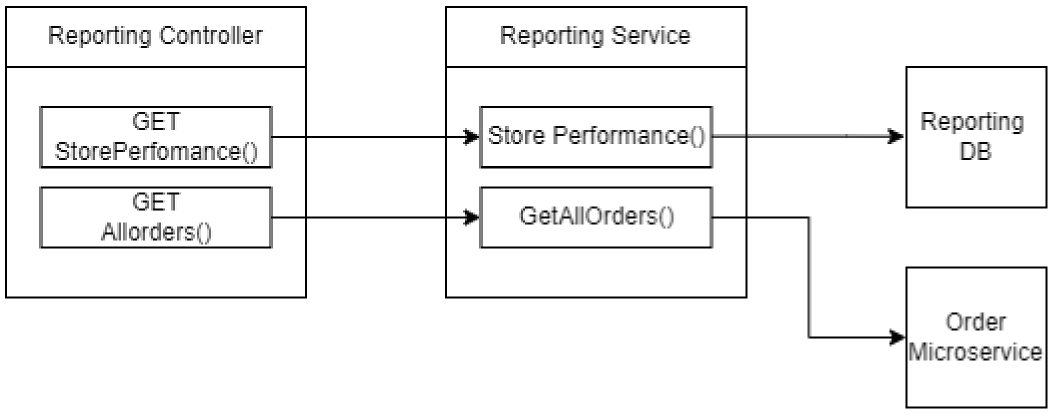


Figure 6 Management Microservice

2.5 Loyalty Microservice

As per the requirements, the architecture for the customer loyalty program is also designed. The purpose of the Loyalty service is to keep track of customer spending, award points and offer discounts to repeat customers.

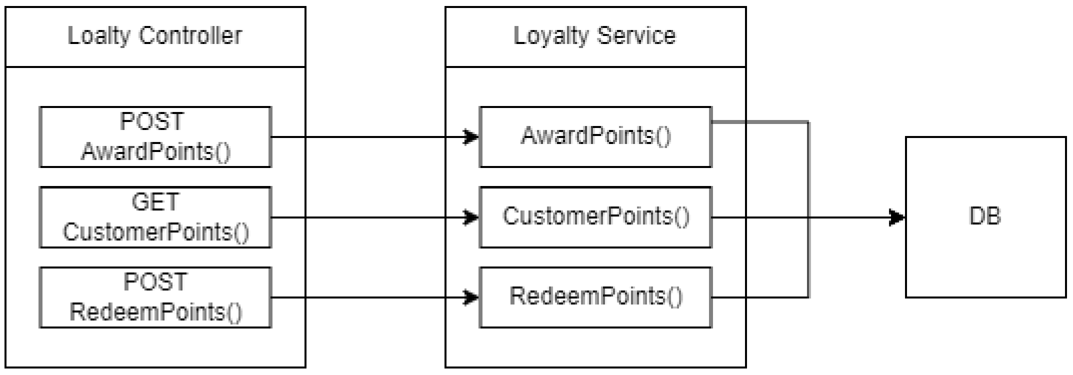


Figure 7 Loyalty Microservice

3 UI Screenshots

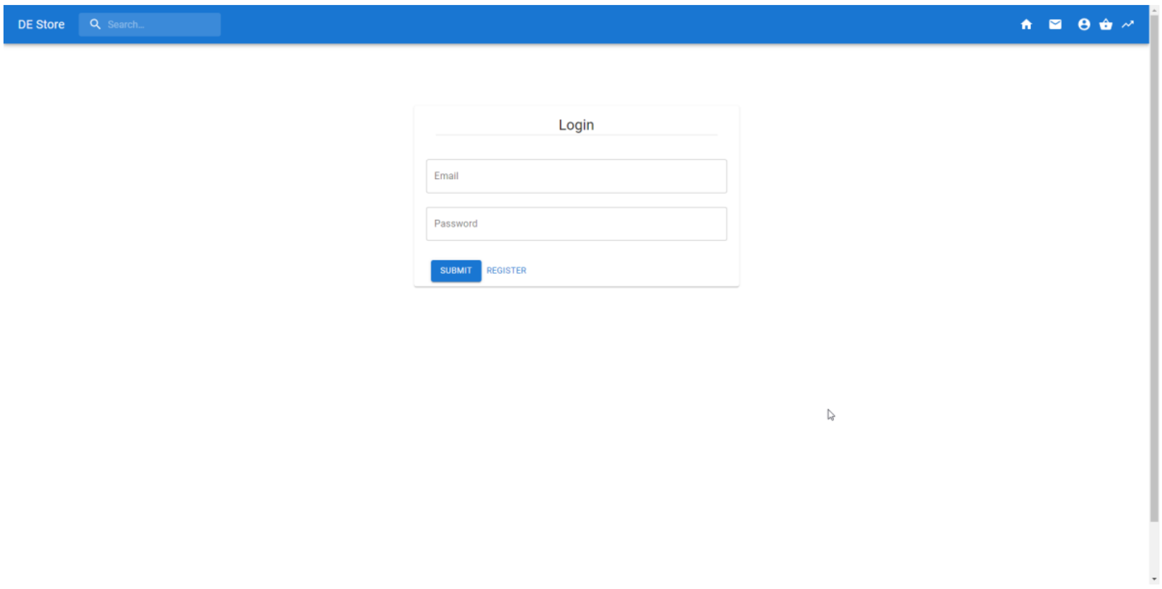


Figure 8 Login Page

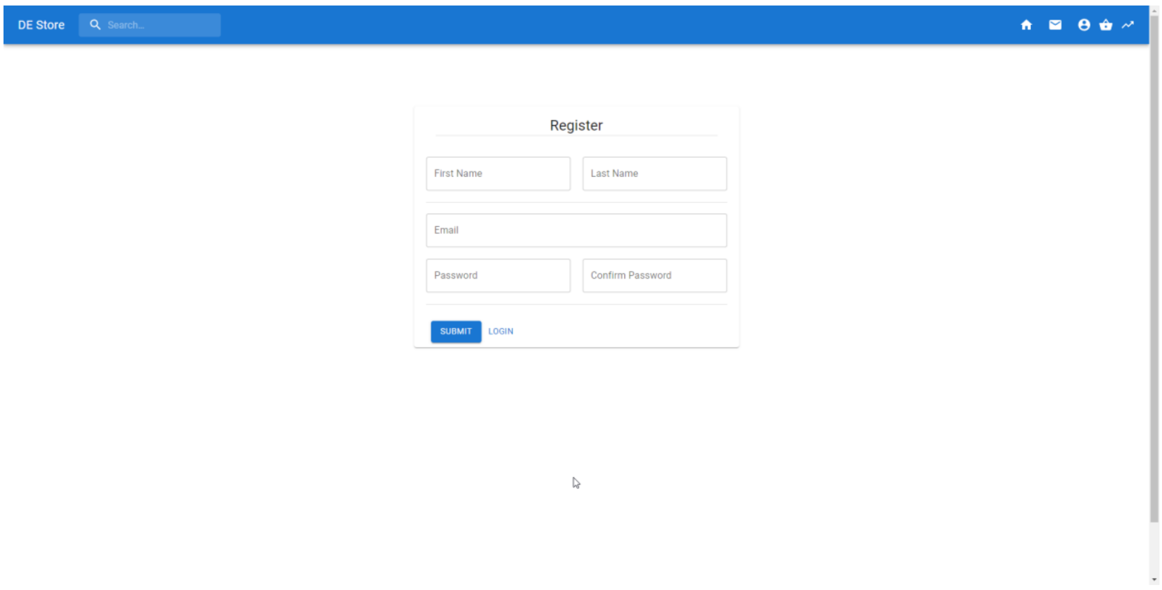
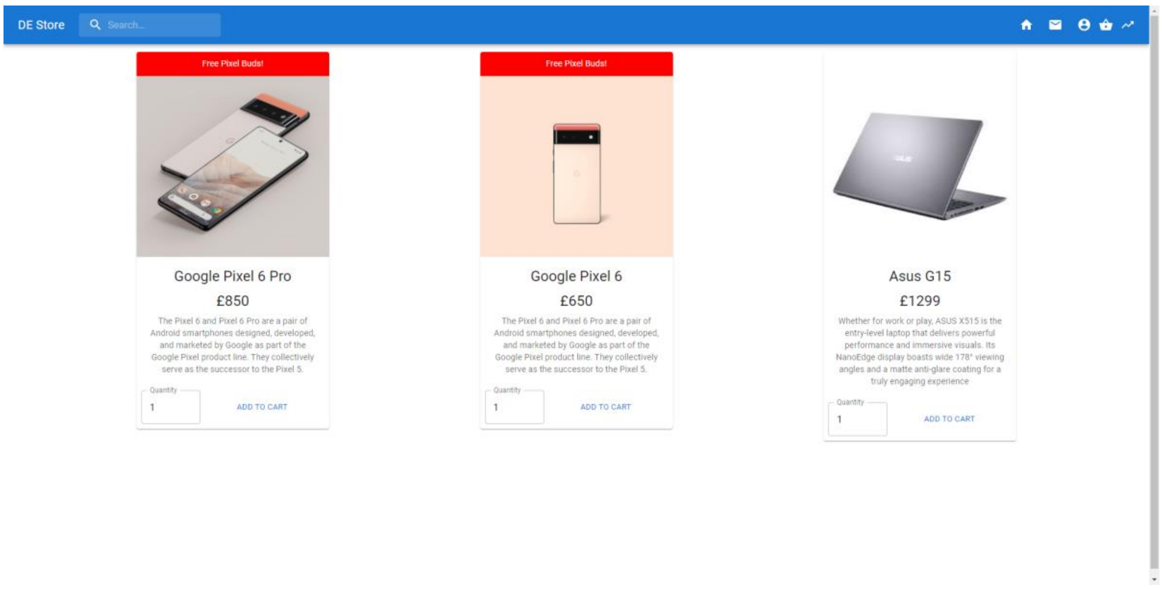


Figure 9 Register Page



Google Pixel 6 Pro

£850

The Pixel 6 and Pixel 6 Pro are a pair of Android smartphones designed, developed, and marketed by Google as part of the Google Pixel product line. They collectively serve as the successor to the Pixel 5.

Quantity: 1

ADD TO CART

Google Pixel 6

£650

The Pixel 6 and Pixel 6 Pro are a pair of Android smartphones designed, developed, and marketed by Google as part of the Google Pixel product line. They collectively serve as the successor to the Pixel 5.

Quantity: 1

ADD TO CART

Asus G15

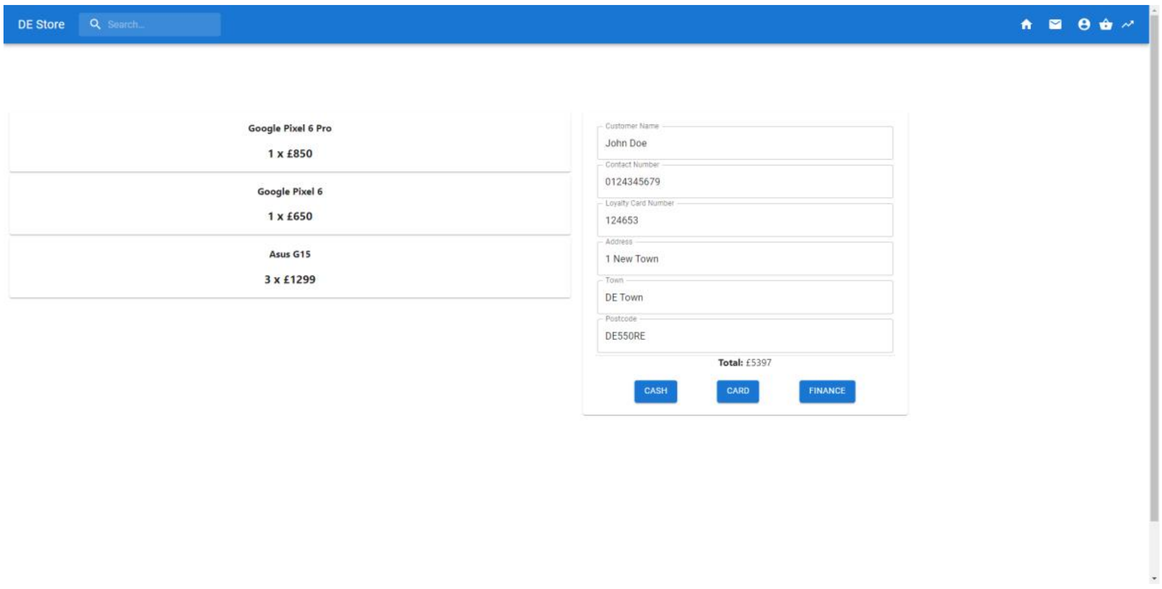
£1299

Whether for work or play, ASUS X515 is the entry-level laptop that delivers powerful performance and immersive visuals. Its NanoEdge display boasts wide 178° viewing angles and a matte anti-glare coating for a truly engaging experience

Quantity: 1

ADD TO CART

Figure 10 Products Page (Accessible by all the employees)



Google Pixel 6 Pro

1 x £850

Google Pixel 6

1 x £650

Asus G15

3 x £1299

Customer Name: John Doe

Contact Number: 0124345679

Loyalty Card Number: 124653

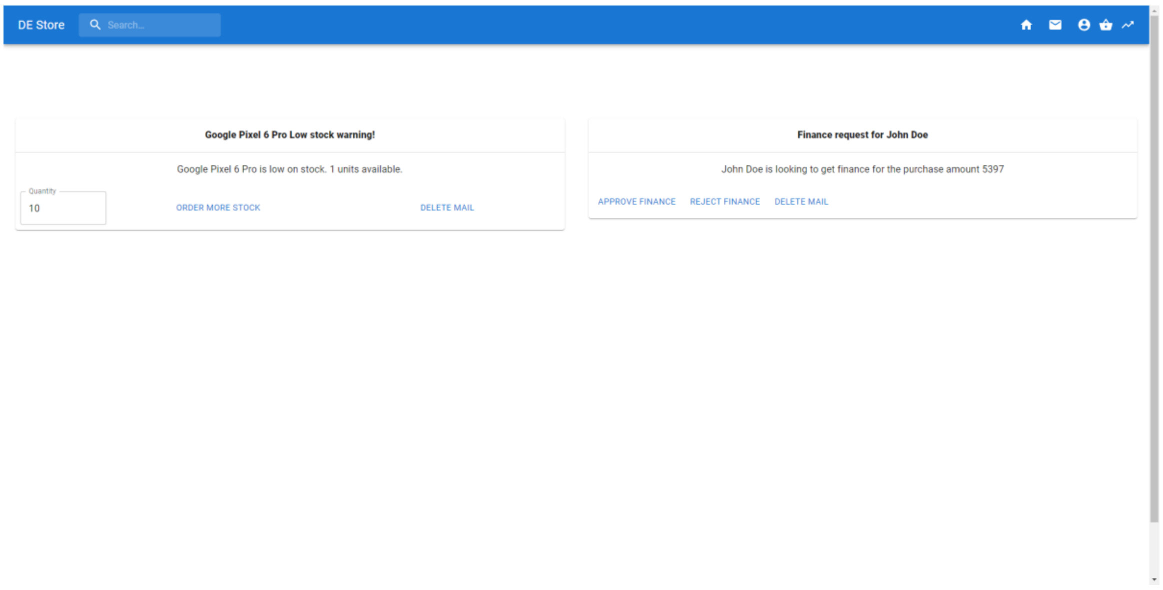
Address: 1 New Town

Town: DE Town

Postcode: DE550RE

Total: £5397

Figure 11 Cart Page (Accessible by all the employees)


DE Store

Search...

Google Pixel 6 Pro Low stock warning!

Google Pixel 6 Pro is low on stock. 1 units available.

Quantity: 10

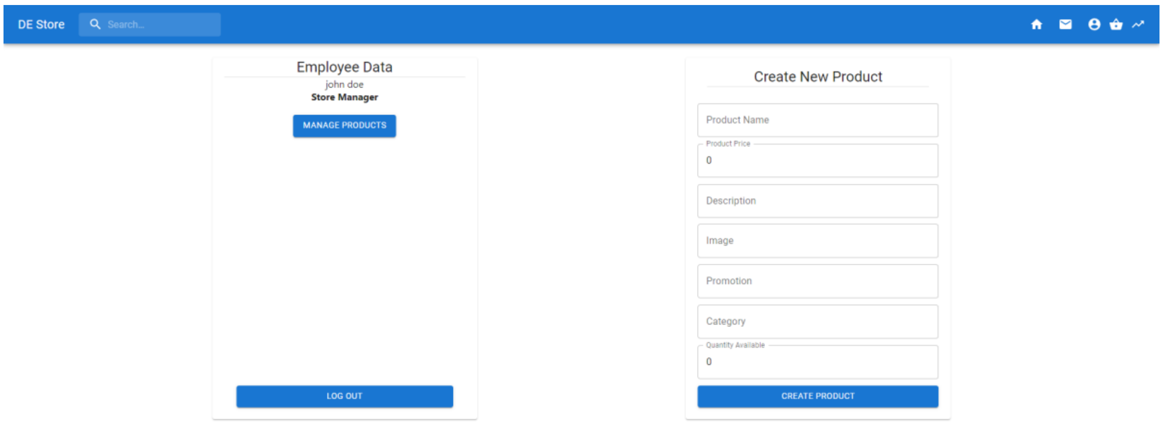
ORDER MORE STOCK | DELETE MAIL

Finance request for John Doe

John Doe is looking to get finance for the purchase amount 5397

APPROVE FINANCE | REJECT FINANCE | DELETE MAIL

Figure 12 Manager Inbox Page (Accessible by all the managers only)


DE Store

Search...

Employee Data

john doe

Store Manager

MANAGE PRODUCTS

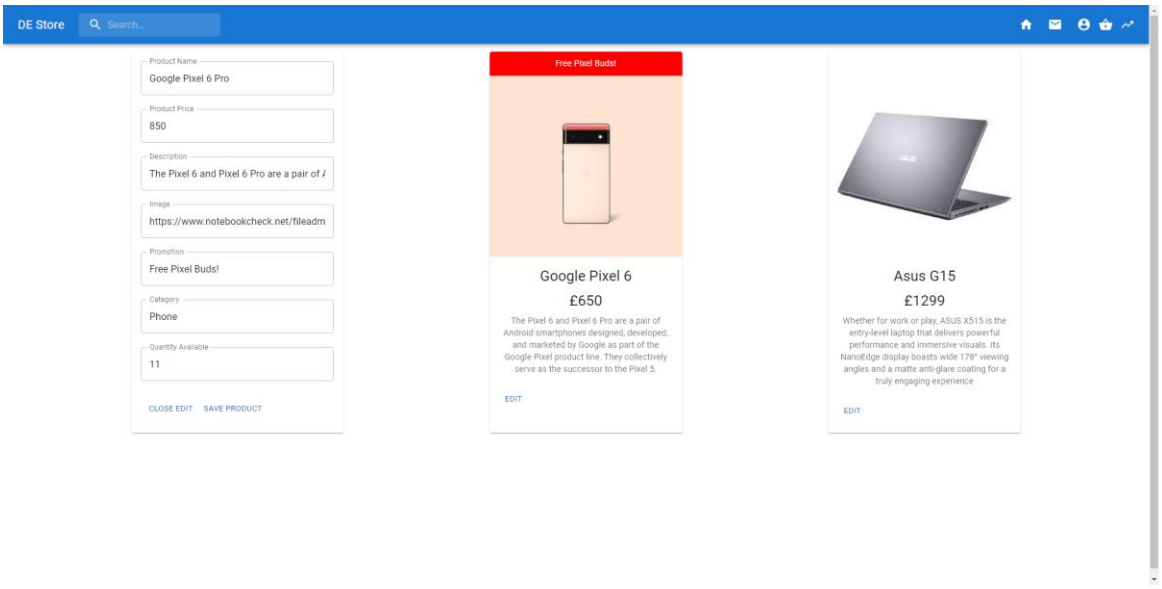
LOG OUT

Create New Product

Product Name	
Product Price	0
Description	
Image	
Promotion	
Category	
Quantity Available	0

CREATE PRODUCT

Figure 13 Account Portal and Product Creation (Product creation & management only available to managers)



Product Name	Google Pixel 6 Pro
Product Price	850
Description	The Pixel 6 and Pixel 6 Pro are a pair of A
Image	https://www.notebookcheck.net/fileadm
Promotion	Free Pixel Buds!
Category	Phone
Quantity Available	11

CLOSE EDIT SAVE PRODUCT

Free Pixel Buds!

Google Pixel 6

£650

The Pixel 6 and Pixel 6 Pro are a pair of Android smartphones designed, developed, and marketed by Google as part of the Google Pixel product line. They collectively serve as the successor to the Pixel 5.

EDIT

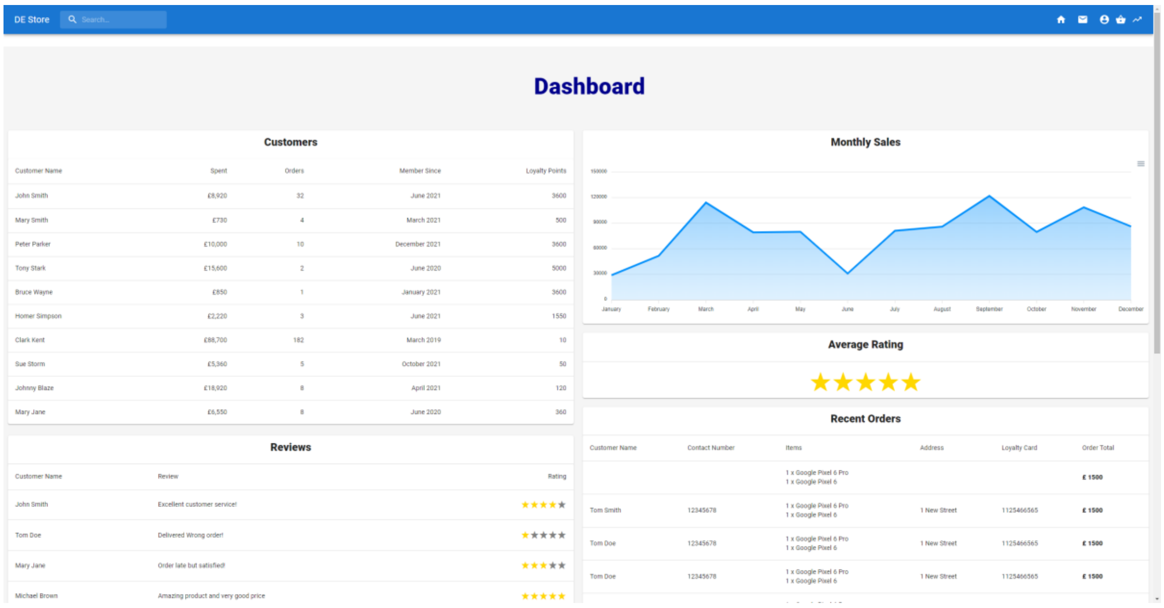
Asus G15

£1299

Whether for work or play, ASUS X515 is the entry-level laptop that delivers powerful performance and immersive visuals. Its NanoEdge display boasts wide 178° viewing angles and a matte anti-glare coating for a truly engaging experience

EDIT

Figure 14 - Product Management Page (Accessible to managers only)



Dashboard

Customers

Customer Name	Spent	Orders	Member Since	Loyalty Points
John Smith	£8,920	32	June 2021	3600
Mary Smith	£730	4	March 2021	500
Peter Parker	£10,000	10	December 2021	3600
Tony Stark	£15,600	2	June 2020	5000
Bruce Wayne	£850	1	January 2021	3600
Homer Simpson	£2,220	3	June 2021	1550
Clark Kent	£88,700	182	March 2019	10
Sue Storm	£5,360	5	October 2021	50
Johnny Blaze	£18,920	8	April 2021	120
Mary Jane	£6,550	8	June 2020	360

Reviews

Customer Name	Review	Rating
John Smith	Excellent customer service!	★★★★☆
Tom Doe	Delivered Wrong order!	★☆☆☆☆
Mary Jane	Order late but satisfied!	★★★☆☆
Michael Brown	Amazing product and very good price	★★★★★

Average Rating

★★★★★

Recent Orders

Customer Name	Contact Number	Items	Address	Loyalty Card	Order Total
		1 x Google Pixel 6 Pro 1 x Google Pixel 6			**£ 1500**
Tom Smith	12345678	1 x Google Pixel 6 Pro 1 x Google Pixel 6	1 New Street	1125466565	**£ 1500**
Tom Doe	12345678	1 x Google Pixel 6 Pro 1 x Google Pixel 6	1 New Street	1125466565	**£ 1500**
Tom Doe	12345678	1 x Google Pixel 6 Pro 1 x Google Pixel 6	1 New Street	1125466565	**£ 1500**

Figure 15 Reporting and Store Performance Dashboard

4 Evaluation

From the design and development of the prototype, it is evident that the microservices architecture is the most suitable for DE Store. The full Minimum Viable Product (MVP) development of the project solidifies the theory and the reasoning behind microservices. From this MVC, the DE-Store can expand both vertically and horizontally without having to worry about breaking unrelated parts of the platform.

Due to the reasons set out in the prototype design chapter and the full development of the MVC, it is highly recommended that DE Store adopt this Architecture pattern. With the rapid development of the MVC, we expect that DE-Store to seriously consider awarding the contract to our firm as we have demonstrated excellent knowledge of the domain and will provide the best product for DE-Store, going above and beyond the specified requirements.

5 Bibliography

Mauersberge, L. (2017, August 14). *Microservices: What They Are and Why Use Them*. Retrieved from leanix: https://www.leanix.net/en/blog/a-brief-history-of-microservices