Assignment 6: Designing our Product

Group 11A

Lucian Negru
Radu Constantinescu
Kendra Sartori
Ahmed Ibrahim
Manar Al-Robayi

TA: Oskar Lorek
Coach: Thomas Overklift
Client: Unetiq BV

May 2022

Contents

Chapter 1

Problem Analysis: Order Processing

To deliver a satisfying product which tackles the issue at hand, we have to first understand the problem. Ordering large quantities of products can be tedious and time-consuming for the customer as well as for those responsible for the processing. Manufacturing businesses are greatly affected by this, as they often order large quantities of small, unnamed and similar products. Because of this, businesses can spend a lot of time having manually processing these large orders. This chapter aims to elaborate on the problem to better understand what needed to be done.

Section 1.1 will describe the four main issues with Order Processing which we have tackled. Section 1.2 will go into detail about the use cases of the system and research the potential impact such a product will have on the market. In Section 1.3 we will discuss the already existing solutions for Order Processing, specifically what they accomplish and what their shortfalls are. Finally, in Section 1.4 we will state the goals of our project.

1.1 The Four Dimensions of Order Processing

The problem we were assigned to solve consists of the following four parts:

1. **Structuring the data from the Excel files**
   Many businesses keep their order lists in Excel documents only to have to manually add everything to the basket when the time comes. This wastes a lot of the time for the person filling out the order when dealing with large lists. Our system accepts Excel files, in various levels of structure, and compiles the shopping basket automatically for the user. The challenge was structuring the data when there is faulty or missing information.

2. **Processing the data from the PDF files**
   The system also supports orders in the form of PDF documents as many companies prefer to send invoices of their order list directly to the shop for them to process. The PDF documents can be very unstructured and may have completely different formats from one another. Therefore, manual processing of the order is usually done on the shop's side, again wasting time. We have decided on a solution that can adapt to any of the formats they may have and accurately construct the shopping basket.

3. **Designing the user interface for the frontend of the online shop**
   Many online stores are not well optimised for devices such as phones or tablets, leading to a worse experience for customers which prefer those. For the customer to have a good experience, we have ensured that the interface is friendly and intuitive, as well as compatible with multiple sized displays.

4. **Satisfying the Stakeholders**
   As will be mentioned in the requirements section, the project has multiple stakeholders. One of our biggest priorities is that we satisfy all of them.

1.2 Market Impact of Order Processing

In this section, we evaluate if this can be a viable and useful product by determining the factors and context in which it can be successfully used. Afterwards, we will look over the use cases of the system. We can ask ourselves questions such as: Is there a demand for such a product in the market? Would this product have a meaningful impact? Is there proof that such a product would benefit the targeted stakeholders? This section will be backed up by empirical research. To accurately retrieve data on this issue, we can look at the companies that implement a similar product and have conducted and published market research on this.

1.2.1 Research on Order Processing

One of the companies which have researched the problem and developed software for it is Conexiom [1]. Conexiom states that almost half of purchased orders are received as emailed documents and must be manually entered into a system, often this being done by an employee. In a study that Conexiom has conducted [1], they say that annually in the US, 17 trillion dollars are spent in manufacturing and distribution, out of which 50% of it is manually processed. In this study, they also point out additional interesting factors which say that per day a CSR spends on average 2-3 hours on re-entering mistyped orders and that another side effect of the manual transaction is that it is unable to scale when the business is growing.

[1] https://conexiom.com

From another perspective, if we look at another competitor, Palette Software [2], and look at a customer case study they have done on another sector, namely transportation, we can immediately see the huge impact that their product had on the customer's perspective. Emily Grantham, AP Supervisor at Landstar System inc., says "Before we deployed PaletteInvoice, our invoicing system was labor-intensive and time-consuming for staff. Automated processing has significantly increased our overall efficiency, shortened payment cycles, and helped to improve the manageability of transactions" [2]. This means that customers do see immediate benefits from automation and that such products will have a positive impact on any organisation.

Moreover, we can see that automation in any part of a process can only provide benefits overall, most of them can be seen when talking about capital, by reducing costs, and time and increasing customer satisfaction by having a more efficient processing system. According to a 2016 Accenture report [3], 52% of the surveyed companies had changed their providers in that year due to poor customer satisfaction. Another big company, Capgemini, tells us that 69% of the organisations managed to decrease this issue of customer satisfaction by integrating intelligent automation [3], the rest still prefer human interaction or a mix of the two. This means that automation could be the first step of implementation for any organisation that wishes to tackle this issue.

1.2.2 Use Case Analysis

For this analysis we decided to look over the use cases of the product for our client, this will allow us to give a technical description which takes advantage of the company's background. Two types of users can take advantage of the product, the online shop customer and the purchase order reviewer.

The online shop customer

- Customer wants to order a large list of products
- Customer wants to order an unordered list of products
- Customer wants to efficiently manage and order the set of wished products.
- Customer wants to order a list of products with (possibly) wrong data
- Customer wants to order a list of products with (possibly) empty data
- Customer wants to receive an ordered list from the unordered list of products
- Customer wants to upload the invoice of orders directly to the online shop
- Customer wants to upload multiple PDF formats to the online shop

The online shop employee

- The employee wants to receive an ordered list from the customer
- The employee wants to review only a subset of orders, the rest are automatic

[2] https://www.palettesoftware.com
[3] https://www.accenture.com/us-en

1.3 Already Existing Solutions for Improving Order Processing

The Automated Order Processing System is an eCommerce Automation product. To be able to compare our project to existing products in this category we first have to define what the category is and then look at existing solutions. In this section, we will describe what eCommerce Automation is and discuss three existing products for Order Processing.

1.3.1 eCommerce Automation

eCommerce Automation refers to "the software that helps your online store convert most or all of the manual, repetitive tasks into self-fulfilling, automated tasks." [4] This means that employees can shift their focus to other areas of the business and consumers can rely on a more efficient method of shopping.

Most automation can be described by a Trigger, Condition, Action workflow. Triggers are events initiated by the client directly, in the case of our system it would be the client uploading a file to the online shop interface. Then various conditions can be checked to determine the following action. In our case, the conditions would be the type of processing that should be used depending on the file type, whether the items on the list of orders are available, etc. Lastly, there is the Action stage where the automation process determines what should happen and makes it happen. In the case of our System, that would be the desired items being added to the basket.

eCommerce is one of the world's fastest-growing sectors, shown in Appendix A, with a consensus that it helps businesses thrive and improves customer experience [4]. Therefore, we find it important to compare those factors between our project and existing solutions.

1.3.2 Conexiom

Description: The conexiom software [5] offers processing of orders through the use of machine learning to save time. They do not disclose more information on the use of AI other than to identify data fields. It receives different file types of unstructured data and runs them through a processing algorithm to label, structure, and identify the content. Then it forwards the extracted data to the system for various tasks for every unique vendor.

Shortfalls: The software doesn't allow for files to be sent through the user interface, only via emails. Our system aims to improve that by allowing customers to add the files to the online shop directly through the interface and keep things simpler.

[4] https://www.bigcommerce.com/ecommerce-answers/what-is-ecommerce-automation/
[5] https://conexiom.com/sales/

1.3.3 Tipalti Approve

Description: Tipalti Approve [6] is a platform that allows for PO (purchase order) generation by having users upload invoices in multiple formats and levels of structure, then processing the information and display it back to the user. The returned data can then be ordered or further analysed. The exact process of what happens in the background is not disclosed, unfortunately.

Shortfalls: The user has to upload the invoices to the app, which then sends an order. Our implementation will streamline this by allowing the users to upload directly to any website using the system. Therefore, the platform has to support the different online shops and introduces a middle-man in the ordering process.

1.3.4 Palette Automated PO Software

Description: The Automated Po Software [7] is software that can be implemented into a business either stand-alone or in an ERP (Enterprise Resource Planning) system. The system receives the orders either through emails or scans and matches them to the recipient store. If a match is not met then it is forwarded through to a manual pipeline where an employee will handle it. The matches are then forwarded to the ERP where the data can be used for various actions, including ordering.

Shortfalls: The POs received have to be rather structured and the software does not make much use of machine learning for more complex and unstructured files. The processing consists of parsing rather than adapting, forcing the users to conform to a certain structure of invoice and compromising on user experience.

1.3.5 Research Summary

In conclusion, there exist products that implement similar functionality to the one proposed by our client. There are even products, such as Palette Automated PO Software, which contains most of the requirements. However, we have not found any to completely satisfy all of the requirements, this leads us to conclude that implementation of the Automated Order Processing System was necessary.

[6] https://www.approve.com/po-management-lp/
[7] https://www.palettesoftware.com/resources/purchase-order-automation

1.4 The Two Main Goals of Our Project

The Automated Order Processing System is a project already started, in part, by Unetiq [8]. Our main goal was to implement new functionalities to the project in such a way that they can be integrated into the existing system and be easily expanded upon. Since we started the functionalities from scratch, we aimed to be efficient with our time and take great consideration in our choices as they will affect the continuation of our's as well as Unetiq's work. Our project goals were as follows:

1. **Develop the System which processes orders**
   Processing of orders from Excel documents must be implemented and functional by the end of the course. Processing from PDF documents is the next step of the project and is not expected to be finished by the end of the course, however, what we do should be well documented so that developers can work on it in the future.

2. **Scalable implementation**
   Everything we implemented has been well documented and all research made is readily available for future developers to continue to expand the project. We aimed for our features to be easily integrated with the project as a whole by the end of the course.

[8] https://www.unetiq.com/

Chapter 2

Requirements for the Order Processing System

To fully understand the goal of the project and the wishes of our client we will use 'Requirement Engineering'. Requirement Engineering is the process used to recognise and express the software specifications that allow us to solve application problems [5]. It makes it possible to translate the client's needs and identify incomplete or inaccurate specifications that potential users may have. The following subsections discuss the various methods used to determine these specifications.

2.1 Stakeholders Involved

To identify all the requirements for our project, we first need to know who the stakeholders are. Stakeholders are the parties that are affected by a development project [6]. This project involves three different parties, each with different requirements for the application. The following subsections discuss each stakeholder and their goals.

Unetiq BV

Unetiq BV are our main stakeholders and provide us with the necessary information to complete the project. Unetiq is a company that develops and customises AI software for various other companies to accommodate the automation of manual processes [7]. Their goal is to create a web application that allows customers of online shops to upload their orders through unstructured Excel and PDF files. The orders should be processed in such a way that no human interaction is needed to complete an order. The application should scan the uploaded file and extract a structured order that can then be processed further.

Customers

There are two types of customers that we have identified as stakeholders. The first ones are the customers of Unetiq, they will be discussed in the next subsection. The second type of customer is the customers of online shops. The goal of these customers is to be able to place an order by uploading an Excel/PDF file, instead of going through online shopping and adding each product one by one into a virtual basket.

Online Shops

Online shops are the customers of Unetiq. In this report, we use the term 'online shops' as a term for identifying online shops that want to automate the process of handling orders through Excel and PDF files. The goal of this stakeholder is to allow their customers to upload Excel/PDF files to place their orders. This can be realised by integrating an option to upload Excel/PDF orders on their online shop.

2.2 Requirements Elicitation

There are several types and techniques for elicitation that allow developers to identify all requirements. The two types of elicitation techniques are:

- Direct Approach
- Indirect Approach

Indirect approaches are used to obtain information that could be challenging to extract and articulate. Direct approaches focus on understanding the problem that the client faces and the way they think this problem can be solved [8].

To collect the requirements for the project we have used a direct approach, namely the 'interview' elicitation technique, by meeting with Unetiq. These meetings have also provided us with a clear understanding of the requirements of the other stakeholders.

The interview was unstructured. The questions were formulated on spot during the interview to allow as much flexibility as possible.

The answers to these questions gave us a clear and adequate understanding of the requirements that had to be formed. The requirements were then split into two groups; functional and non-functional requirements. Functional requirements focus on the ability of the system to perform certain tasks, while non-functional requirements focus on the behavioural aspect of the system [9].

The list of all requirements can be found in Appendix D.

2.3 Definition of Requirements for the Order Processing System

The requirements in Appendix D are divided into four categories. The categorisation is done using the MoSCoW technique. This categorisation allows the team to work sustainably and efficiently. One of the key factors of MoSCoW categorisation is that it defines the priorities for each requirement [10].

The requirements are divided into four sections:

- **Must Haves:** The implementation of these requirements gives a minimal viable product on top of which all other requirements are built.
- **Should Haves:** This category includes all requirements that are not vital for a system to work, but are of such importance that they need to be implemented.
- **Could Haves:** These are features that offer improvements to the product but are not necessary for the deliverable.

- **Won't Have:** These are requirements that are seen as infeasible to implement within the development team.

Chapter 3

Designing the Order processing System

In the following chapter, we will be looking over the design of our project. Designing is a step we have taken very seriously as it set the roadmap for the entire course duration. We have started by looking over the requirements stated in **Chapter 2** and discussing the overall feasibility in various categories. We have then analysed the risk given these requirements. Lastly, we have made an architectural design based on the knowledge previously stated.

In the following section, we will take a look at the various categories of feasibility, and how realistic we believe this project is. Then, we will take a look at the risks we have to consider when tackling this project. Finally, we will discuss the implementation for the frontend, backend and server, as well as the reasoning for all the choices we made.

3.1 Feasibility Analysis of Order Processing System

This study has been conducted after we had several meetings with our client, in which we discussed exactly the way and setting in which the final product will be used, to properly understand and assess the practicality of the proposed project. The final analysis has been done after considering multiple factors some of which are technical, operational and time-related.

In addition to those factors, we also looked at other factors that will guarantee that this project can be of use to multiple clients, and it can serve as a solution to the recurring problem that a lot of companies are facing as specified in **1.2.1 Research on Order Processing**.

In the following sections, we will talk about all the different types of feasibility components we should take into consideration. Finally, we will summarise everything to determine the overall feasibility of the project.

3.1.1 Technical Feasibility

In terms of technical feasibility, our team looked at the currently existing technology, and if it suffices our needs. Since this project will be built from scratch, we consider that, after thorough research, several options exist in which we can combine currently existing technologies to achieve the final product. Moreover, we consider that the team has the necessary skills in terms of technical knowledge to accomplish this project. In addition to that, the existence of products that achieve similar functionality to our project ensures that there would not be any technical barriers present.

3.1.2 Operational Feasibility

Regarding operational feasibility, this project matched the business objectives and goals of the company, since their team is also worked on it, however with slight differences in terms of requirements. In addition to this, the integration within the organisation's main focus, was easy since the company is already working on a similar product. Also, the company has provided us with a sample database, and orders, both in the Excel and PDF formats, that we have made use of to test and develop the product. The difference between the two formats was that the data in Excel is more structured as some columns and rows are used. However, the data in the PDF was unstructured and consisted of many different formats.

3.1.3 Time Feasibility

Time is another constraint that this project was faced with. Given that the project is part of the course provided by the university, we - along with our stakeholders - were aware of the limited time frame in which this product has to be delivered. With this situation, we considered that a prioritisation technique was of utmost importance, therefore we decided to use the MoSCoW method, to set our targets straight. We considered that, despite this constraint, the team will be able to accomplish this task and in the end at least deliver the minimal viable product; this has been established in the 'Should have' section of the MoSCoW analysis.

3.1.4 Other Factors

Other factors that could be taken into consideration could be economic and legal feasibility. For the first one, we considered that given the setting in which the project is done, it fell outside of our scope, even if it can be easily recognised that the impact of such a project would be a positive one on the company. Regarding the legal feasibility, the project has been done under legal requirements since all laws and regulations are met.

3.1.5 Feasibility Summary

In conclusion, since all the aforementioned factors were met, this project was feasible. As a team,it was our responsibility to make sure we considered everything and respected all feasibility factors to deliver a viable product in time.

3.2 Risk Analysis of Order Processing System

It was of utmost importance to assess the risk associated with a project in terms of the different criteria that could influence the development process. Although, many aspects might be considered irrelevant when looking at the bigger picture, including them in our risk analysis has ensured that we took into account issues in an anticipative manner to prevent a failure to deliver a highly qualitative project.

In the following subsections, we will describe the different types of problems that we anticipated encountering throughout the past weeks and their relevance. Furthermore, we will explain the solutions that were feasible and suitable for all group members.

3.2.1 Schedule and communication capabilities

After making a thorough team analysis, also including Belbin's team role management theory [1], we have gained insight into how each individual in our team, but also our team composition might affect the way we work as a group.

The team members' prior experience consists of other projects which are part of the curriculum imposed by our study program. However, the Software Project we are currently part of is far more permissive in terms of time allocation and resource management, as it is the first project in which the students have a connection to a real-world client. This might pose a challenge, whereas the impression of freedom could easily affect keeping a proper schedule and making sure we are on track with our project.

On the other hand, the way that people interact has changed significantly in the past couple of years. In addition to the commodity everyone has from working from home, online meetings feel like a much more effective way to discuss further developments, especially given that the group members live in different cities, which makes it harder to meet in person. This can however also affect productivity and morale, leaving people feeling rather lonely than part of a team.

We aim to minimise the risk of running into these problems by having frequent meetings, at least three times a week while keeping in touch daily, to make sure we are all on track with the work that needs to be done. Furthermore, we plan on meeting at least once a week to work together in-person to create the general feeling of belonging to a team.

3.2.2 Possible growth in requirements

Requirement elicitation is the foundation on top of which our project is built. The requirements dictate all other software engineering processing which can also influence productivity, quality and risk, requirements engineering constituting the foundation of the whole software development process [11]. For that reason, we must have a clear overview of what we have to do and what the different stakeholders expect from us. However, some of the requirements are probably less obvious and there is always a chance that we miss some of them, which might negatively impact our process and cause unnecessary delays.

[1] https://www.belbin.com/about/belbin-team-roles

In this regard, we believe it is important to have a good relationship with our client and constantly keep him updated regarding the development of the product, through the weekly meetings we have set. Furthermore, we have decided to follow the guidelines described by the agile development framework and structure the process into sprints that last one week each.

3.2.3 Lack of experience with React

Our client has expressed his preference for us to make use of React for building a user interface for our final deliverable. However, none of the team members has experience with this JavaScript library, which could hinder and delay the process by a significant amount of time, equivalent to the time required for us to get acquainted with it and start making use of it.

We wish to overcome this lack of expertise by setting a clear deadline by which each team member needs to try getting acquainted with React. Additionally, we hope that a great collaboration between the members will make so that mutual help is offered between us to ensure we are all up for the challenge that creating this component poses.

3.2.4 Lack of knowledge about GDPR and other legal aspects

Given that we are developing a tool which will be, hopefully, widely used by different categories of stakeholders, it is our job to ensure that it is easy to use and does not have any negative impact on these stakeholders. In other words, it is our job to protect the users of our tool from possible harm that it might do, especially in terms of privacy and security. In this regard, we need to make sure that it is compliant with the guidelines described by the GDPR directive as well as other laws and regulations that we might not yet be aware of.

We would like to avoid storing sensitive personal data retrieved from the customer (i.e. the address) anywhere in our database. However, in the eventuality that we need to store the data to achieve a more accurate result, we will make sure to comply with the GDPR, making sure that the customers are aware of how their data is being processed as well as deleting the stored data after a certain amount of time.

Given that we are not yet aware of all the legal implications of our project, we wish to carry out further research into the possible dangers that a user could be faced with and get more informed on these matters. Furthermore, we will ask our client for further information about things that we need to take into consideration regarding the legal aspects of our product.

3.3 Architecture of Order Processing System

In the following section, we will describe our research and implementation for the various components of our project. We will look over the frontend implementation, the backend implementation, and how we set up the database.

3.3.1 Frontend

In terms of frontend, we decided to use React [2] since it is one of the most popular frameworks and this can be useful for further implementations on this project. On top of this, using React was one of the non-functional requirements that the client provided us.

React is an open-source library that is built to simplify the building process of advanced user interfaces. The core of the React user interface consists of components which are parts of the user interface, that can be integrated to become a whole functioning user interface which simplifies the UI building process. There are several advantages of using react over regular JavaScript, first, it confines options of coding and so the code is easier to write and also becomes cleaner. Furthermore, the open-source aspect of React and its wide usage make it so that a great number of new tools are released regularly [12].

To have an overview of what our team decided that the implementation would look like, we created a low fidelity prototype. This prototype will serve as a boilerplate on which we can brainstorm with our client to achieve their desired implementation. The prototype can be found in Appendix B. A low fidelity prototype is helpful in the early stages of development, offering a visualization of alternative design solutions, which provokes innovation and improvement, alongside offering users a more comfortable way of making suggestions [13]. There are three pages created for the application, namely the Home page, the Order page and the Basket page. The Home page lets users upload a file, the order page makes it able for the user to review the processed order and the basket is the final page with all of the orders. As for the popup for the file upload, the team decided to change the popup to an appearing form as it provides better space to view the uploaded files.

3.3.2 Backend

Python

For the backend, we agreed on using Python [3] as the main programming language. The choice of choosing Python is one connected to the non-functional requirements that we were provided by the client. There are a few advantages of using Python for the backend of this application.

The biggest advantage of using Python is that it is one of the easiest programming languages. This is caused by the simplicity of Python which not only makes it also easier to understand programming in Python but also makes it easier to improve your programming skills and it allows products to require less coding. This simplicity also causes the programmers to focus on the product instead of having to spend a lot of time understanding the programming language.

Python also provides an extensive library which allows for more possibilities in programming. Another advantage of Python is that is a very well-known programming language which results in lots of tutorials, guides and documentation available online. This way the programmer has to spend less time being stuck and can get help easily online [14].

[2] https://reactjs.org/
[3] https://www.python.org/

Django

We agreed on using the Django [4] framework. The other option for the main framework – that wasn't chosen – was Flask. One of the reasons Django was chosen is that it is multi-threaded while Flask is single-threaded. Another reason for choosing Django is that it has more functionalities that are out-of-the-box and can be used. Further comparisons between the two can be found in Appendix C.

Django uses the Model View Template which is a software design pattern. There are a few advantages to using Django as the main framework. An advantage of Django is that it performs well on security as it offers protection against common security attacks. Django is also a production-ready framework which could be useful if our client decides to push this application to production. The inclusion of templates ensures that beginners, like ourselves, implement the best practices. Django is also a very well-known framework which results in lots of tutorials, guides and documentation available online. This way the programmer has to spend less time being stuck and can get help easily online [15, 16].

3.3.3 Database

In terms of databases, we first decided to use a serverless option, because of the small scale of the product and the additional benefits that it provides. An advantage of using this serverless option is the cost efficiency, with the option to host it on-demand. On-demand refers to paying only for the actual read and writes operations that our application would perform. Hosting the server can be very costly, but managing the server is costly as well, as there is licensing, maintenance, support etc. The scale potential is another reason for using the serverless database as the automatic scalability is based on the workload [17]. The database we chose firstly was DynamoDB, provided by Amazon, a non-relational database which is optimised for intensive read workloads. DynamoDB [5] was used as it does not require SQL, it is easier to read and it is not expensive [18].

The team tried to query the orders with DynamoDB, but the primary key was needed to process the queries, which in the case of the orders is not available all of the time. Another downside of using this database is that it then has to scan the whole order and it uses pay per query, which would be too expensive and not feasible. To resolve these complications, another database called Postgres was used. PostgreSQL [6], also known as Postgres, is a well-known relational open-source database. This database was used because of the higher performance in processing the queries. The advantages of Postgres are that it has a better combination with Python, it is well compatible with the framework and it does not require a pay per query. Postgres also performs well on scalability as it supports the technology for it and it performs well on security due to extensibility [19].

[4] https://www.djangoproject.com
[5] https://aws.amazon.com/dynamodb
[6] https://www.postgresql.org/

Chapter 4

Values to Take Into Consideration

As software developers for a project, it is easy for us to focus on the technical requirements and ideas of the project proposed by our client. However, this narrow view impedes us from considering the direct or indirect effect our work could have on other people. When we aim to be responsible and accountable for our work and its effects on others, we have to look at which *Values* we want to achieve, what norms have to be respected, and the technical requirements necessary. One way of designing our product with these ideas in mind is through Value Sensitive Design (VSD).

VSD is "...a theoretically grounded approach to the design of technology that accounts for human values in a principled and comprehensive manner throughout the design process" [20]. In other words, it is the philosophy of considering values for your design and specifying them as technical requirements. This helps the technical requirements shine a light on ethical components and helps guide the developer to being more responsible and accountable.

The chapter aims to identify the ethical categories relevant to our project and explain how we incorporated VSD to mitigate them. The ethical categories will be **Ethics Regarding Data Management and Transparency** and **Ethics Regarding Artificial Intelligence**.

4.1 Ethics Regarding Data Management and Transparency

The software we have designed accepts files – in either Excel or PDF format – from the user and temporarily stores them while running an algorithm. Therefore, we thought it would be appropriate to consider the value of our user's privacy in our design, specifically how we manage data and transparency. The backend also contains data on the products offered through the Furning [1] website, this database is solely queried and not accessible by the user.

[1] https://www.furning.com/

The data stored from the user cannot be used to identify a person as it only contains products for an order. Our system simply processes the data and compiles a shopping basket. The user manually adds personal information such as name and address when completing the order, this is out of the scope of our system so no such data will be stored. To respect the value of privacy for our users, we opted to not store any data they have submitted after the processing stage. Once the files have been processed and the basket returned, the data collected by the system will be discarded. This can be verified by the empty file list and conveys trust to our users.

Furthermore, to ensure transparency, the processing system will be made available for other developers to test and implement in their online shops. This includes the algorithm used to process the data as well as how the data is temporarily being stored.

4.2 Ethics Regarding Artificial Intelligence

Artificial Intelligence and Machine Learning are topics of controversy when it comes to making ethical designs. The GDPR [21] states that users must be informed of the use of AI in decision-making and must be offered the choice to opt-out of said decision-making. Therefore, we thought it would be appropriate to discuss our use of AI for the processing of the user data, and how we handle the GDPR conditions.

Our system uses AI to analyse the Excel and PDF files uploaded by the users and determine which products they are referring to. This process may include decisions on what label corresponds to the different fields of the data, correction of misspelt data and correction of missing data. The resulting basket is then shown to the user, who has the option to review and edit it. Furthermore, the availability and documentation of the project after its deployment ensures that the AI is explainable and accessible to others.

We believe our system does not raise any ethical concerns when it comes to the use of AI. The user can review the decision making and opt out of it, the user is informed of the use of AI, and the AI is explainable. These conditions correspond with the GDPR statements of an accountable AI and enforce the backwards-looking responsibility of the project developers.

Bibliography

[1] Conexiom. (2021) Don't Wait to Automate. [Online]. Available: https://conexiom.com/resources-category/infographics/

[2] Palette. (2021) Landstar benefits from Automated Invoice Processing with PaletteSoftware. [Online]. Available: https://www.rillion.com/case/landstar/

[3] Capgemini. (2018) Consumers are embracing AI and will reward organizations that offer more human-like AI experiences. [Online]. Available: https://www.capgemini.com/us-en/news/consumers-are-embracing-ai-and-will-reward-organizations-that-offer-more-human-like-ai-experiences/

[4] A. Maier. (2020) Expert insights into Customer Experience. [Online]. Available: https://codecoda.com/en/blog/entry/expert-insights-into-customer-experience

[5] Z. Jin, "Chapter 3 - importance of interactive environment," in *Environment Modeling-Based Requirements Engineering for Software Intensive Systems*, Z. Jin, Ed. Oxford: Morgan Kaufmann, 2018, pp. 29–39. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128019542000030

[6] Concepta. (2018) How to Define Stakeholders for Your Software Development Project. [Online]. Available: https://www.conceptatech.com/blog/how-to-define-stakeholders-for-your-software-development-project

[7] Unetiq. (2022) Unetiq - Your Agency for AI and Data Science Services. [Online]. Available: https://www.unetiq.com

[8] S. Khan, A. B. Dulloo, and M. Verma, *International Journal of Information and Computation Technology*. International Research Publications House, 2014.

[9] M. Glinz, "On non-functional requirements," in *15th IEEE International Requirements Engineering Conference (RE 2007)*, 2007, pp. 21–26.

[10] Komodo. (2022) How to Use MoSCoW Prioritisation to Help Deliver Better Software Products. [Online]. Available: https://www.komododigital.co.uk/insights/how-to-use-the-moscow-prioritisation-technique-to-execute-software-projects

[11] N. T. More, B. S. Sapre, and P. M. Chawan, "An insight into the importance of requirements engineering," *International Journal of Computer and Communication Technology*, vol. 8, no. 1, pp. 29–31, Jan. 2017.

[12] U. Pisuwala, "The benefits of reactjs and reasons to choose it for your project," Mar 2022. [Online]. Available: https://www.peerbits.com/blog/reasons-to-choose-reactjs-for-your-web-development-project.html

[13] A. Affairs, "Prototyping | usability.gov," 2022. [Online]. Available: https://www.usability.gov/how-to-and-tools/methods/prototyping.html

[14] "Advantages and disadvantages of python - how it is dominating programming world," Aug 2021. [Online]. Available: https://data-flair.training/blogs/advantages-and-disadvantages-of-python/

[15] A. Joy, "11 advantages of django: Why you should use it," 2022. [Online]. Available: https://pythonistaplanet.com/advantages-of-django/

[16] V. Singh, "Flask vs django in 2022: Which framework to choose?" 2022. [Online]. Available: https://hackr.io/blog/flask-vs-django

[17] A. Arkhipov, "Serverless databases: Pros cons, benefits, examples on the market," 2021. [Online]. Available: https://www.techmagic.co/blog/serverless-databases/

[18] A. Newbies, "AWS DynamoDB: Non-Relational Database," 2022. [Online]. Available: https://awsnewbies.com/dynamodb/

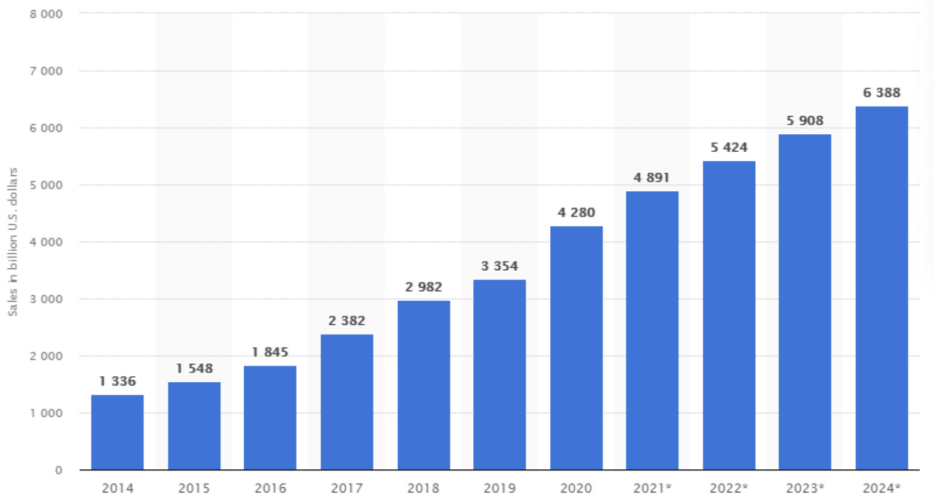
[19] "Advantages of PostgreSQL," 2022. [Online]. Available: https://www.cybertec-postgresql.com/en/postgresql-overview/advantages-of-postgresql/

[20] B. Friedman, P. H. Kahn, and A. Borning, *Value Sensitive Design and Information Systems*. John Wiley & Sons, Inc., 2008, p. 69–101.

[21] (2018) 2018 reform of eu data protection rules. European Commission. [Online]. Available: https://ec.europa.eu/commission/sites/beta-political/files/data-protection-factsheet-changes$_en$.*pdf*

Appendix A

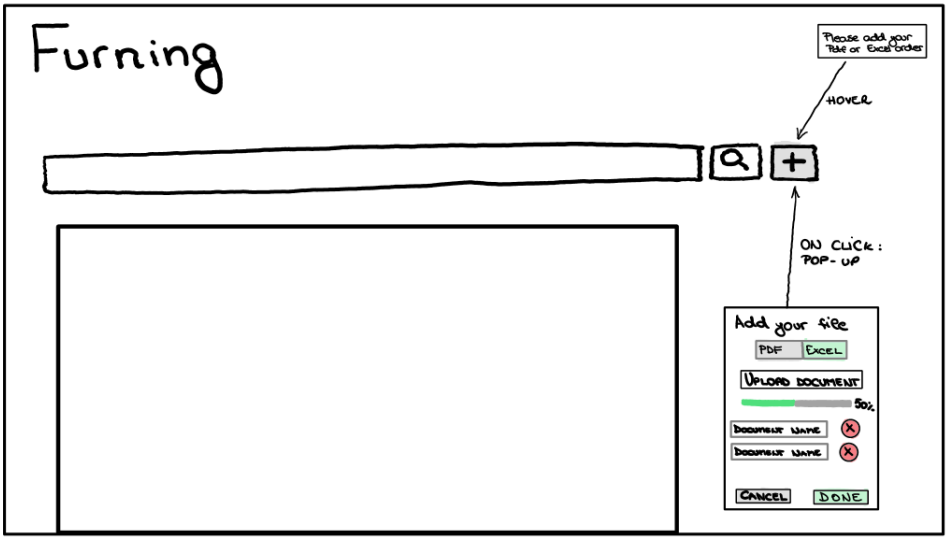
Graph displaying the growth of the eCommerce market (note: * signifies prediction)

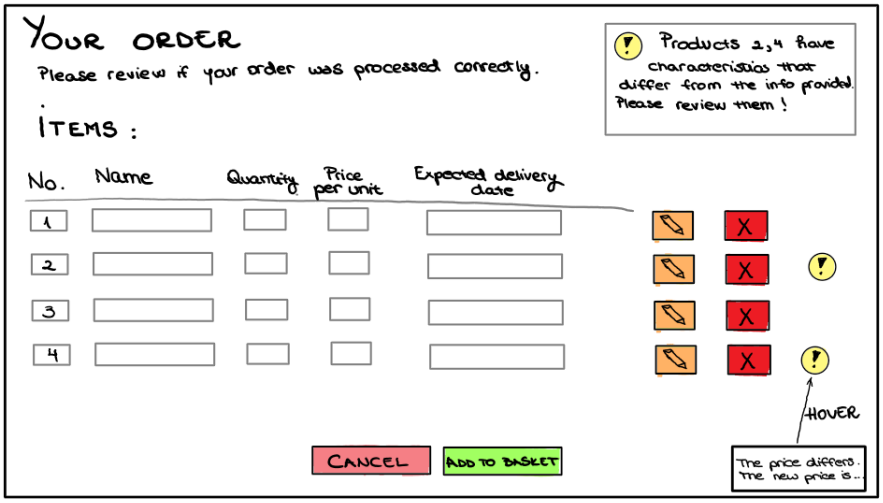


Source: statista.com

Appendix B

Low-Fidelity Prototype





Appendix C

Table of comparisons between Django and Flask

Parameter	**Django**	**Flask**
Type of framework	Django is a full-stack web framework that enables ready to use solutions with its batteries-included approach.	Flask is a lightweight framework that gives abundant features without external libraries and minimalist features.
Working of Framework/Data Model	Django follows an object-oriented approach that enables object-relational mapping (linking databases and tables with classes)	Flask works on a modular approach that enables working through outsourced libraries and extensions.
Project Layout	Django is suitable for multiple page applications.	Flask is suitable for only single-page applications.
Bootstrapping Tool	-Django-admin is the in-built bootstrapping tool of Django that allows the creation of web applications without any external input.	Flask does not come with an in-built bootstrapping tool.
Database Support	Django supports the most popular relational database management systems like MySQL, Oracle etc.	Flask does not support the basic database management system and uses SQLAlchemy for database requirements.
Flexibility	Django is less flexible because of its in-built features and tools. Developers cannot make changes to the modules.	Flask is a micro-based framework with extensible libraries making itself a flexible framework for developers.
Template Engine	Django is inspired by the Ninja2 template but has its built-in model view template that makes the development process easier.	Flask used Ninja2 template design.

Control	Developers do not have full control over the modules and functions of Django because of built-in libraries.	Flask allows developers full control over the creation of applications with no dependencies from external libraries.
Working Style	The working style of Django is Monolithic	The working style of Flask is diversified style.
Debugger	Django does not support any virtual debugging.	Flash has an in-built debugger that offers virtual debugging.
Routing and Views	Django framework supports the mapping of URL to views through a request.	Flask web framework allows mapping of URL to class-based view with Werkzeug.
Structure	Django framework structure is more conventional.	Flask web framework structure is random.
HTML	Django supports dynamic HTML pages	Flask framework does not support dynamic HTML pages
Usage	Django is suitable for high-end technology companies like Instagram, Udemy, Coursera etc.	Flask is suitable for companies and projects that want experimentation with the module, architecture of the framework like Netflix, Reddit, Airbnb, etc.

Appendix D

The requirements of our project

Non-Functional Requirements:

- The system will be scalable to the extent that it can also process orders of other file-types.
- The system will make use of a server-less database.
- The system will process at least 70% of the excel orders in a correct way.
- The system will run on a web application with React Framework.
- The system will have a Python back-end.

Must Haves:

- As a customer I want to be able to search for a product on a search bar.
- As a customer I want to get a list of products that match my search keys.
- As a customer I want to be able to upload an excel file with an order.
- As a customer I want to get an 'automatically created structured order' after uploading my order in an Excel file.
- The system will filter out unnecessary information from the excel orders.
- The system will map the column names for each column in an order.
- The system will map the types for each column in an order.
- The system will convert a filtered order into a structured Python datatype.

Should Haves:

- As a customer I want to be able to download a structured CSV file with the automatically created order from the Excel file that I have uploaded.
- As a customer I want the system to notify me if my order contains an unavailable product.
- As a customer I want the system to notify me if my order contains a product that does not exist.
- As a customer I want the system to notify me in case my order can not be processed.
- As a customer I want to be notified if an actual price differs from my expected price.

Could Haves:

- As a customer I want to be able to manually add products to the automatically created order.
- As a customer I want to be able to change an automatically created order.
- As a customer I want to be able to delete items in an automatically created order.
- As a customer I want to be able to let the system know whether my order was processed correctly
- As a customer I want to be able to upload a PDF file with an order.
- As a customer I want to be able to upload multiple files at the same time.

Won't Haves:

There are no requirements that we will certainly not do.