Project Charter

PRJ4D – Group 9

Authors: Afonso Fernandes da Cruz,
Daniel Schmidt, Mathijs Vandooren

Location: Venlo

Date: 23.06.24

Contents

1. Introduction

In this project, we will be working with an ecommerce company called “Vadotex” but the trade name under which the business operates for the direct to consumer side of the business is called “pyjamaonline”.
The goal is to make an analytics dashboard on a website in which the employees can easily see all incoming orders as well as seeing all customers and products that they have.
This company is based in Maastricht with over 40 years of experience in the textile business. At first, they started off as a big warehouse close to the center of Maastricht as a B2B business selling their own brands to other businesses that would sell the product on their webshop as well as on a market and other places online. After this, they started selling their own excess products that did not sell well or that they had some leftovers of online. This turned out great and quickly increased in revenue. Most of their B2C sales comes from the marketplace "Bol", but their goal is to expand to sell more on their own webshop as well as other marketplaces such as Amazon to increase revenue.

Their logistics skills are well-developed, but they lack knowledge of marketing and interpreting their data.
That is where we come in, helping them make sense of their data as well as develop a machine learning algorithm to predict future sales will be beneficial for this company. These algorithms will be the focus of this project and will consume most of our time.
Sales of pyjamas fluctuate a lot, there are several factors influencing the sales of pyjamas. Holidays and the time of the year play a big role, but other factors include the day of the week, the weather, the area in the Netherlands and the state of the economy. The company mainly sells their products in the Netherlands, but 25% of their sales also come from Belgium.

2. Business Case

The Business Case chapter gives an overview of why the project gets realized from a commercial standpoint.

1.1 Rationale

Currently, there are many companies that already display sales data in different ways. Most of them are doing a poor job at giving our customer exactly what they need. Lightspeed for example offers sales data in a limited time window of 12 months, while bol.com shows a more detailed view, but only for its marketplace. This data is not integrated for all sales channels and lacks precise information like sales for a product specific size. EffectConnect and Exact Online are also used but fall short in providing specific crucial data.

1.2. Problem Outline

The existing solutions offer no unified view for the sales data across multiple channels. The data from Lightspeed is limited to a 12-month window, without giving a more detailed view.

There is a need for a solution that can integrate sales data from all channels and provide detailed analytics.

The lack of such access to their data hinders the company to make decisions on important topics and optimize operations.

1.3. Objectives

Connection between the webshop and marketplace:

Develop an API connection to collect orders from different marketplaces and display on a single interface, similar to the already existing Lightspeed content management system. This will include syncing stock levels, prices, and product information like images, descriptions and attributes.

SARIMA-Alogrithm:

Implement a SARIMA-Algorithm to predict future sales based on historical data based on external factors like holiday-plans, weather, weekdays and seasonality.

Analytics Dashboard:

Create an analytics dashboard on a website to display a comprehensive sales information and detailed analysis that allows the company to make better-informed decisions.

3. Approach

This chapter talks about why the Scrum methodology is the most effective approach for developing our Sales Product Analytics and Benchmarking site. While waterfall and hybrid approaches have their advantages, Scrum is best on adaptability, continuous improvement and efficiency, making it perfectly suited to the needs of this project and our small team.

Benefits of Scrum for our project:

Adaptability to evolving needs: Building a data platform involves some unknowns. User needs and functionality can adapt based on data gathered during development. Scrum's nature allows us to easily apply these changes into sprints, ensuring the site remains relevant.

Faster delivery and user feedback: Scrum prioritises the delivery of working features in short sprints. This allows us to quickly launch a basic version of the site with core functionality. Early user feedback from these sprints is invaluable in prioritising features and refining the overall user experience.

Efficient workflow for a small team: With a limited team size, clear communication and collaboration are critical. Scrum encourages this through daily stand-up meetings and sprint reviews. Team members stay on the same page, identify roadblocks early and address them together, resulting in a more efficient development process.

Reduced risk and early course correction: Scrum focuses on delivering working features in increments. This allows us to identify and address potential issues early during sprints, minimising rework and development costs associated with significant changes later in the project.

Continuous improvement based on data: The nature of Scrum allows for continuous improvement of the site. Data collected throughout the development process (user behaviour, feature usage) informs future sprints.

3.1. Quality Management

To ensure that our dashboard meets the expected performance, reliability, and usability, we apply the following quality management practices.

Quality Assurance: Every line of code is reviewed by a team member to ensure quality and to ensure we keep up to our coding standards.

Quality Control: Weekly meetings are held to review progress and discuss sprints. In addition, we will have three stand-up meetings per week to answer questions, discuss challenges, and reassign work if needed.

GitHub is used for collaboration, branches are set up for isolated tasks and are reviewed when merged.

Coding Readability:

- Using indentations to mark the beginning and end of control structures to specify code blocks.
- Writing compact code and avoiding longer functions. Ideally, each function should have a single task.
- Using the DRY (Don't Repeat Yourself) principle. Automating repetitive tasks whenever necessary.
- Avoiding deep nesting. Keep code straightforward and easier to follow.
- Keeping the length of lines manageable for better readability

Standardizing Headers:

- Having consistent headers in each module containing:
    - Module Name
    - Creation Date
    - Creators Name
    - Summary of Functionality
    - Functions and Variables

Unique Identifiers:

- Assigning meaningful, unique names to each variable.

Comments and Documentation:

- Writing comments to explain complex code. Making sure that the reader gets guided through the logic and algorithm.

Exception Handling:

- Use try-catch blocks to manage exceptions.

Peer Review:

- **Conducting peer reviews for all code changes**

Testing and Verification:

- Writing unit tests to verify logic of each module and function.

Closing Notes:

- Test all code on real devices, browsers and on live servers to ensure it is working as intended.

3.2. In Scope and out of Scope

During this project, we can only focus on a certain amount of features to show data within the analytics dashboard. There are many features that could be implemented, therefore it's important to put boundaries in place and decide on the most important features that we can implement. These are features that will give the most value to the stakeholders as well as making sure that that certain feature can be done within the given timeframe.

The goal of this project is to help the ecommerce company make additional profits through analysing their data. This is not a clearly defined objective, so our real objective will be to deliver a fully functional website with the necessary API connections from the marketplace and content management system, graphs showing insights in sales and the amount of units sold per product as well as a database that will save all the incoming data.

We will acquire customer reviews for the seller (ecommerce company) via the marketplace API. These are only the seller reviews, not the product reviews. We can only access the product reviews via web scraping, which is not within the scope of this project.

During week 6 of the project, we will make sure we have a working API connection, we've set up our database as well as having a web hosting service to host the domain name. We've gathered a lot of information about the company and what kind of data would be valuable, but we're still thinking of ways to solve some of the problems they are having such as not knowing if they should keep more stock for B2C clients because the profit margins are way higher as well as other issues that could potentially be solved with the software application.

In week 7 we'll implement our first feature which is to show a graph with the amount of sales for that day, in the deliverables section we'll go into more detail on these features mentioned here in the timeline.

In week 8 we'll give more insights on these sales, particularly the amount of revenue separated for the webshop and marketplace. The amount of profit can be calculated as well with information given from the company such as transport fees, cost of goods, commission, software costs etc.

This will be helpful to know how much discounts will do for profit, because software costs, warehousing costs and employees have a set monthly price, increasing sales

with lower profit margins could be a way to increase profits with more information on this data.

During the 9th week we'll track warehousing stock so we have more insights if the amount of stock is going up or down overall, each month new products will be published on the marketplace but the amount of stock for these products is never the same because of the pre-sale for B2B only clients. Tracking this will give more information on how much inventory to buy in.

3.3 Constraints

The first and most notable constraint is the API limit of both the Lightspeed and Bol API which makes it initially hard to load in all the data in the database.

The Lightspeed API has a limit of 300 requests per 5 minutes for all the API endpoints per account, Bol has an API rate limit per API endpoint which usually ranges from 5 to 25 per minute.

There are other bottlenecks that could be solved by increasing the budget but are not needed for this project which includes increasing the Dyno type which is what powers the backend. It currently has 512mb Ram and supports up to 10 process types which is sufficient for this project.

For the PostgreSQL database we have an essential-1 PostgreSQL database on Heroku which supports up to 10 GB of disk space which is enough as well for this project.

For Redis we're using Premium 0 on Heroku which supports 50 mb of memory.

Our React frontend is hosted on Vercel where we use a pro plan.

If we were to make it faster, the API calls and other background tasks for celery would get executed faster. For now, this is not needed for this project.

3.4. Problems and Solution

Current System:

- The company uses EffectConnect to send and retrieve data between the Bol marketplace and Lightspeed.
- Plans to switch from Lightspeed to Shopify in the future.

Issues:

- Monthly expense for EffectConnect.

- Suboptimal connection. This means that the in and output of data coming from EffectConnect could be improved. For example, when we change a product description on Lightspeed it can take up to 6 hours before that changed data gets sent to Bol.
- Suboptimal data quality. Because we can't control what data gets sent to Lightspeed, we can't save the extra JSON data fields that could've been sent from Bol to Lightspeed such as the 'updatedAt' which indicates when an order changed its status from for example 'not_shipped' to 'shipped' which could be valuable information.
- It's not personalised, because EffectConnect is a plug and play solution that doesn't account for small details that only occur for a couple of ecommerce companies. For example, for a company that sells clothes there are different sizes, and it'd be best to list total sales of all sizes for that product instead of all the sizes mentioned separately.

Solution:

- Making a cheaper version of EffectConnect for which the frontend (React application) is hosted on Vercel and the backend (Django REST) is hosted on Heroku. Both the PostgreSQL and Redis databases are also hosted on Heroku. EffectConnect costs around 450 euros per month, whilst this solution only costs about 18 euros a month.
- Fetching new orders and product information more frequently so that the company can make faster changes to possibly the product description or other attributes of the product, such as the image.
- By Sorting the data ourselves, we can make sure to include all clothing sizes together when giving a total revenue per product. This feature is special for only clothing brands that want to couple all their sizes together, which is why it makes sense that EffectConnect doesn't offer this.

4. Deliverables

Website:

- Domain hosted on AWS Amplify or Vercel.
- Server-Side Rendering (SSR) and Incremental Static Regeneration (ISR) using Next.js for fast loading and graph display.
- Possibility of using React for the frontend

Backend:

- Django Rest Application hosted on Heroku
- JWT authentication
- Celery workers for background tasks

API Connection:

- Bidirectional data streams between the CMS and marketplace.
- Sync data between channels.

Database:

- PostgreSQL database to store all business data.
- Includes API-gathered data and downloadable CSVs.

Features:

- All features stored in a GitHub repository.
- Custom version of EffectConnect to replace expensive service:
  1. One-way API connection to Shopify for syncing Bol marketplace data.
  2. Orders sync every 5 minutes, content sync hourly.
  3. Option to enable/disable specific data streams to avoid SEO issues.
- Data mapping with search bar for barcodes/SKUs.

Sales Data Analysis:

- Using Postman and Lightspeed API to filter and gather order data.
- Display sales by period (day, week, year) with comparison to previous periods.
- Feature to identify bestsellers for restocking.
- Review display for products when restocked.

5. Timeline

Week 1 + 2: Discuss project ideas, list out pros and cons of each project idea and think about ways on how to execute on the idea such as which tech stack we would be using, how reasonable the project idea is, how beneficial this idea can be for that company as well as other minor factors.

Week 3 + 4: Start working on the project charter, talking to the ecommerce company about how we can add the most value to their company, what kind of data would be the most beneficial for them and have more knowledge about ecommerce in general. In these 2 weeks we will go more in depth in what the best tech stack would be for our project.

Week 5 + 6: We started with getting a domain and our initial Next.js website. We had little to no experience working with API's, so in these 2 weeks we tested a lot with the data that we could gather via the Bol and lightspeed api. The company had to get information on the sales per country and per sales channel, so gathered this information using both api's and cleaned out the data so we could give them an accurate revenue number per sales channel and country for the first quarter of the year.

Week 7: We finished our first version of the project charter as well as learning building with the Next.js framework for the website and connecting a database to the website. We had another meeting with the company talking about all the deliverables that we are going to implement as well as setting expectations as to what we can and won't implement during this project.

Project week 1: In this week we will focus on displaying a data from lightspeed on the website https://www.ecom-dashboard.nl/ such as sales per day/month/year, average order amount graph over time, sales per product for a period of time, amount of orders per day graph as well as a map of the Netherlands with dots giving more information on where the most sales in the Netherlands comes from.

Project week 2: In this week we will build a data stream going from Bol to Shopify, the business currently uses lightspeed for their webshop but in the future they want to use Shopify because of its ease of use. Because of this we want to make sure all the product information, orders, stock levels are already accurately displayed on the Shopify platform. Shopify and Lightspeed are both Content management systems which are used by companies to make and manage their webshop. On this platform you can see incoming orders, stock levels and change the look and feel of the webshop.

Week 8 + 9: In this period we have enough knowledge and experience working with their data to start working on the main task for this project which is building the algorithm. In this period we will get more information on this algorithm as well as making a first implementation of this Prophet/Sarima algorithm.

Week 10 + 11: Working on more specific features on the website, such as a button that will download an Excel sheet with sales for that specific sales channel and country for that quarter. This will automatically update every quarter. Another feature we will be working on is when a product is not in stock and set to LVB (“logistiek via Bol”) on the marketplace, it checks if the webshop does still have any stock for this product left and it switched from LVB to logistics by the seller. Meeting with the company to talk about the current progress and planning of the work for upcoming weeks.

Week 12, 13 + 14: Working on other features that will be displayed on the website, the first feature is calculating the amount of commission paid to Bol for selling products on their website. Depending on the product that you sold you have to pay a percentage of the retail price to Bol as well a fixed price which is 0,85€ in most cases. The percentage differs but, in most cases, this is around 12%. Another feature is looking at how many inventories gets added per drop to the webshop, is this more or less than the amount of orders during that period? Is inventory in the warehouse increasing or decreasing?

Project week 12: Working on features that are not finished yet and polishing up the website, working on devOps, making it easy to find certain information on the website.

Project week 13 + 14: If all the features that we said we would implement are done then will be working on graphs that will keep track on ad spend across all sales channels such as google ads and sponsored products on Bol. In this week we have a final meeting with the business to talk about all the features that we implemented and feedback on how this project went.

6. Success Criteria

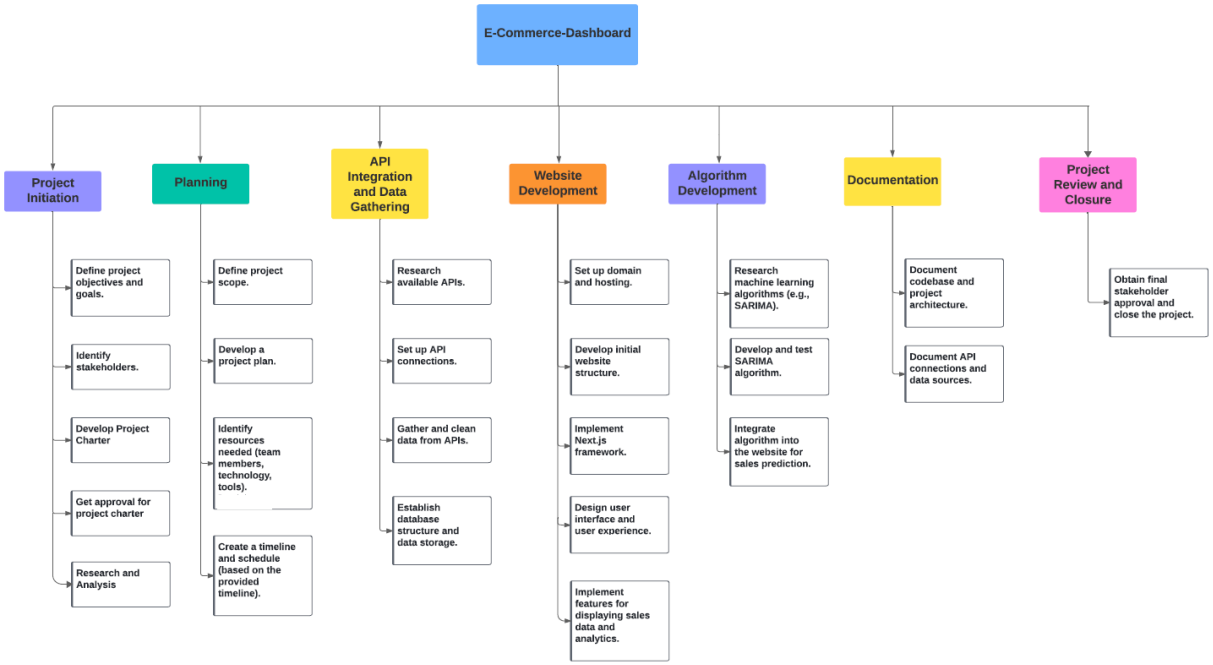
The following must be provided to call the project successful:

A satisfied stakeholder and a working machine-learning algorithm, which will either be a Prophet or Sarima model. We want to reach these goals by taking care of the following: data accuracy and integrity. This includes making sure that the schema of the database is complete and includes all the data that we need during this project.

We will take care of the data visualisation effectiveness, this includes sales graphs and overall visualisation of data that is stored in the database to the frontend.

Security and compliance play a big part in this as well because we do not want unauthorised users accessing the data, for this we will implement JWT authentication as well as placing constraints on the API views that can be called and making sure that sensitive data is stored in env files.

7. Work Break Down Structure



8. Project Team

Our project-team and what areas we focus on:

Afonso Fernandes da Cruz: Frontend, Machine-learning algorithm

Daniel Schmidt: Analysis (creation of various artefacts), Organisation (Task creation and distribution, meetups, repository-management), Frontend (website), Machine-learning algorithm (using openweathermap.org and python)

Mathijs Vandooren: Backend, Frontend, PostgreSQL storing data, Redis caching data and message broker for Celery, Prophet Machine-learning algorithm, Celery background tasks