DJILLALI LIABES UNIVERSITY OF SIDI BEL ABBES
FACULTY OF EXACT SCIENCES
DEPARTMENT OF COMPUTER SCIENCES

FOR ACADEMIC LICENSE IN THE
DOMAIN OF MATHEMATICS AND COMPUTER SCIENCES
WITH SPECIALITY IN
SOFTWARE AND INFORMATION SYSTEMS ENGINEERING(ISIL)

Developing a web platform for managing wholesalers and their customers

---

Author:
SAHRAOUI M.Taher Amine
HADJAZI M.Hisham

Supervisor:
Pr.GAFFOUR Abdelkader

A project paper submitted in fulfillment of the requirements for the
License in Software and Information Systems Engineering. LMD Degree

June 25, 2021

"Computer Science is a science of abstraction -creating the right model for a problem and devising the appropriate mechanizable techniques to solve it."

Alfred Aho

"If debugging is the process of removing software bugs, then programming must be the process of putting them in."

Edsger Dijkstra

"Computer science is the operating system for all innovation."

Steve Ballmer

"Computers are useless. They can only give you answers."

Pablo Picasso

DJILLALI LIABES UNIVERSITY OF SIDI BEL ABBES
FACULTY OF EXACT SCIENCES
DEPARTMENT OF COMPUTER SCIENCES

Abstract

Faculty of Exact Sciences
Software and Information Systems Engineering

Licence in Software and Information Systems Engineering

Developing a web platform for managing wholesalers and their customers

by SAHRAOUI M.Taher Amine
HADJAZI M.Hisham

The purpose of this project is to analyse, design and develop a web platform to manage the daily tasks for wholesalers and connect them to their everyday customers, we are targeting in this platform only wholesalers that provide and deal with restaurants, coffee shops, cafeterias and any bushiness that falls in this category. we also try to solve the problem of logistics with a smart system that schedule the deliveries in a manner that helps both the wholesalers and their clients. The project that we submit is only a small part of our solution as we will be presenting the core of the solution with website to showcase the solution, other parts of the project that we are planning to keep working on includes a mobile app that runs in both Android and IOS and a multi-platform desktop software that can run offline to help the wholesalers run their business regarding internet availability.

Acknowledgements

First and foremost, praises and thanks to the Allah, the Almighty, for His showers of blessings throughout our work to complete the graduation project successfully. We would like to express our deep and sincere gratitude to our project supervisor, Professor Gaffour Abdelkader, for giving us the opportunity to do this project and providing invaluable guidance throughout our work. His dynamism, vision, sincerity and motivation have deeply inspired us. He has taught us the methodology to carry out the project and to present the work as clearly as possible. It was a great privilege and honor to work and study under his guidance. We are extremely grateful for what he has offered us. We would also like to thank him for his friendship, empathy, and great sense of humor. We are extending our heartfelt thanks to his wife, family for their acceptance and patience during the discussion we had with him on the project and thesis preparation.

بِسْمِ اللهِ الرَّحْمنِ الرَّحِيمِ

الحمد لله عز وجل على نعمة الاسلام و الحمد لله عز وجل على نعمة الايجاد و الحمد لله عز وجل على نعمة الامداد. و الحمد لله عز وجل على نعمه كافة و من بينها نعمة العلم. اشكر الله الذي وفقني انا و اخي الكريم حجازي هشام الذي توج الله حياتي بمعرفته فمعرفة اهل الخير نعمة و الذي شاركته و شاركني هذا الدرب الطويل في هذه السنوات المباركة التي تعلمنا فيها من العلم النافع كما معتبرا و مباركا. فالحمد لله و اهدي عملي كله خالصا لله و لا حول و لا قوة الا بالله العلي العظيم. اما بعد فالقي تحياتي لسيد الاسياد الذي علمنا ان الاتقان العمل و جهاد في سبيل العلم شيم المسلم و المؤمن القوي. و اهدي هاذا العمل لمن كان قدوة لنا في كل عمل فيه خير و علم فيه نور سيدنا رسول الله صلى الله عليه وسلم. و من ثم اشكر و اهدي هاذا العمل الى جوهرة الحياتي الام الكريمة التي علمتني الخير و العلم و سعي و تفائل و عدم الاستسلام و الى الوالد الكريم الذي كان سبب في اختياري هاذا الدرب المبارك بان اكون جزء من خيرة ما انجب هاذا الوطن العزيز من طلاب و اساتذة في احد اهم علوم العصر و هو الاعلام الالي. اشكر بعد ذلك اهلي كل من اخواتي و الخال الكريم و جدة الكريمة و كذلك اهدي هاذا العمل و شكري لاخي مبارك حجازي هشام الذي لم يكن مقصرا في عمله و الذي كان قدوة لنا في إرادته الصلبة و تحقيق الغايات الحميدة و أظيف بذلك شكري لوالديه اللذان كانا سببا في اعطاء هاذا الوطن الغالي رجلا و ابا من معدن النادر. اهدي عملي كذلك لاخي صغير بوسهلة عبد الرحمن و للاختي صغيرة حجازي تسنيم. اشكر كذلك كل الاساتذة الكرام الذين ساهمو في بنائنا عامة و كل من غافور عبد القادر و ميغنونيف سمير و خالفي محمد و ظريف و شايب يزيد و زلاط صالح و عبد الرزاق بشير بويجرة و طموح عادل و بوكلي محمد و بنعوم فراح و بنهامل نبيلة و بحري و حمو و عيساوي سهام و عباد هدى و برباح و رفيق بالخوجة و فحصي محمود خاصة و اتمنى ان يكون هاذا العمل محصلة لجهودهم معنا. اشكر كذلك كل اخوة من صلاة الاسلام و عائلته رايس اكرم و عائلته كريمة و صلاة علي وعائلته الكريمة و اخ عادل موريوا و عائلته كريمة و اخ مقران ايمن و عائلته الكريمة و اخ مداح عبد الستار و عائلته الكريمة و دحمان عماد الدين و عائلته الكريمة و كل اخوة الكرام من طلبة و طالبات مع تماني التوفيق و مزيد من ابداع للجميع.

الطالب. صحراوي محمد طاهر أمين

بِسْمِ اللهِ الرَّحْمنِ الرَّحِيمِ

الحمد لله رب العالمين والصلاة والسلام على أشرف الأنبياء والمرسلين سيدنا محمَّد وعلى آله وصحبه ومن تبعهم بإحسان إلى يوم الدين، وبعد . أشكر الله العلي القدير الذي أنعم عليَّ بنعمة العقل والدين. القائل في محكم التنزيل "وَفَوْقَ كُلِّ ذِي عِلْمٍ عَلِيمٌ" صدق الله العظيم . وقال رسوله الصادق الأمين (صلى الله عليه وسلم):"من صنع إليكم معروفاً فكافئوه, فإن لم تجدوا ما تكافئونه به فادعوا له حتى تروا أنكم كافأتموه". فإني أشكر الله تعالى على فضله حيث أتاح لنا إنجاز هذا العمل، فله الحمد أولاً وآخراً. ثم أشكر والداي اللذان لم يذخرا جهدا في نصحي ودعمي وتربيتي وكذالك أشكر الزوجة الكريمة وإبنتي الحبيبة على صبرهم معي خلال هذه السنوات. ثم أشكر أولئك الأخيار الذين مدّوا لنا يدَ المساعدة، خلال هذه الفترة، وفي مقدمتهم أستاذي المشرف على الرسالة فضيلة الأستاذ الدكتور/ غافور عبدالقادر الذي لم يدَّخر جهداً في مساعدتنا، فله من الله الأجر ومنا كل تقدير حفظه الله ومتّعه بالصحة والعافية ونفع بعلومه. كما أشكر القائمين على جامعتنا الحبيبة الجيلالي اليابس وعلى رأسهم عميد كليتنا/أ.د لاكريب مصطفى، ورئيس قسمنا/ د.حمودة محمد. وجميع أساتذتي الكِرام خلال مشواري الدراسي الذين كانو سببا في تعليمي وتثقيفي، وأتمنى من كل أستاذٍ أن يسامحني إن كنت قد أسأت إليه أو قصرت في حقهِ، وأريد أن أشكرَ بالأخص د.بشير بويجرة عبدالرزاق الذي كان سببا في عودتي لمقاعد الدراسة بعد سنوات طويلة من دخولي الحياة المهنية ولولا مشورته علي وإصراره على عودتي للدراسة لما عدتُ إليها، فله مني جزيلَ الشكرِ والتقدير على حرصه علي. ثم أشكر أخي وزميلي وحبيبي صحراوي أمين الذي كان له كل الفضل في إتمام هذا العمل وأشكر له صبره معي وثقته التي وضعها في، وأشكر له أهله الذين أحسنو تربيته واللذين كانو لي بمثابة أهلي أيضاً. وأشكر كل زملائي الطلبة خلال مشواري الدراسي الذين كانو لي صحبةً طيبةً ورزقاً حسناً. أكررُ شكري لكلِ الأساتذةِ الكرام الذين علموني وربوني وأحسنو لي وصبرو معي وعلى أسئلتي الكثيرة لهم خلال مشواري الدراسي. فَاللهُ اكبرُ كبيراً والحمدُ للهِ كثيراً وسبحانَ اللهِ بكرةً واصيلاً.

الطالب. محمد هشام الحِجازي

Contents

Abstract ii

Acknowledgements iii

List of Figures vii

1 Introduction 1
- 1.1 Background . . . 1
  - 1.1.1 Detecting the problem . . . 1
  - 1.1.2 Solutions used in the market . . . 2
- 1.2 Obstacles . . . 2
  - 1.2.1 Legacy IT software . . . 2
  - 1.2.2 Unreliable payment methods . . . 3

2 Methodology of the solution 4
- 2.1 UML Case Studies . . . 4
  - 2.1.1 Use-Case for the wholesalers . . . 6
  - 2.1.2 Use-Case for customers . . . 6
  - 2.1.3 Use-Case for delivery system . . . 7
- 2.2 UML Sequence Diagram . . . 8
  - 2.2.1 Sequence Diagram for the Client order . . . 9
- 2.3 Deployment Diagram . . . 10
- 2.4 Choosing the right design pattern . . . 10
  - 2.4.1 What is the problem with PHP ? . . . 11
  - 2.4.2 What is MVC desigen pattern ? . . . 11
  - 2.4.3 How MVC can help ? . . . 13

3 Front End Development 15
- 3.1 General website theme . . . 15
  - 3.1.1 Colors . . . 17
  - 3.1.2 Fonts . . . 17
  - 3.1.3 Logo and animation sketches . . . 18
- 3.2 Landing page . . . 19
- 3.3 Registration page . . . 20
  - 3.3.1 Wholesalers Dashboard . . . 20
  - 3.3.2 Form Validation . . . 22
- 3.4 Editing Client/Wholesalers Info . . . 23
  - 3.4.1 Client Dashboard . . . 24
  - 3.4.2 Notifications . . . 24

4 Back End Development **27**
4.1 Database (relational databases) . . . 27
4.2 Client/Server Systems . . . 28
4.3 Over View Of the database . . . 31
4.4 Routing pages . . . 31
4.5 Registration verification with Algerie Poste API . . . 32
4.6 Hosting . . . 32
4.7 Domain Name . . . 32
4.8 Unauthorized Access (ERROR 403) . . . 34
4.8.1 Restricting access to web server directories . . . 34
4.8.2 Restricting access to visitors from Algeria only . . . 35
4.8.3 Restricting access media files . . . 36
4.9 SSL/TLS Encryption . . . 36
4.9.1 What is SSL/TLS encryption? . . . 36
4.9.2 End-to-end HTTPS with Cloudflare . . . 37
4.10 Gzip Compression . . . 38
4.10.1 Web application size . . . 38
4.10.2 Preparing backend in PHP . . . 39
4.10.3 Setting up the server . . . 39
4.10.4 Verifying Improvements . . . 40
4.11 Code Optimization . . . 41
4.12 SQL Injection Protection . . . 41
4.12.1 prepared statements and parameterized queries . . . 41
4.12.2 DBMS Privileges . . . 42

5 Results and Benchmarks **43**
5.1 Verification and Validation . . . 43

6 Conclusion **45**
6.1 Final thoughts . . . 45
6.2 Future updates . . . 45

A Technologies and Terminologies **46**
A.1 Development Method . . . 46
A.1.1 DevOps . . . 46
A.2 Development Tools . . . 47
A.2.1 Visual Studio Code . . . 47
A.2.2 Beekeeper Studio . . . 47
A.2.3 StarUML . . . 48
A.2.4 AMPPS . . . 48
A.2.5 GitHub . . . 48
A.2.6 Filezilla . . . 49
A.2.7 Geany . . . 49
A.2.8 LaTeX . . . 49
A.2.9 Cloudflare . . . 50
A.3 Front End Side . . . 50
A.3.1 HTML . . . 50
A.3.2 CSS . . . 51
A.3.3 Java Script . . . 51
A.3.4 Bootstrap . . . 52
A.4 Back End Side . . . 52

A.4.1 PHP 52
A.4.2 MySQL 53
A.4.3 Apache web server 54
A.4.4 Domain Names with freenom 54
A.4.5 Web Hosting with Infinityfree 54

B Data Base Tables **55**

List of Figures

1.1 Algerie Poste API . . . 3

2.1 WholeSalers Case Study . . . 6
2.2 Customers Case Study . . . 7
2.3 Delivery System Case Study . . . 8
2.4 Sequence Diagram . . . 9
2.5 Deployment Diagram . . . 10
2.6 MVC pattern . . . 12
2.7 Model View Controller II . . . 13
2.8 MVC Buisness Logic . . . 14

3.1 Elements of user experience . . . 16
3.2 example of a sketch . . . 18
3.3 digitized sketch . . . 19
3.4 Landing Page . . . 19
3.5 Registration Page . . . 20
3.6 Wholesalers dashboard . . . 21
3.7 Add-modify items . . . 21
3.8 Add-modify items 2 . . . 21
3.9 Form Validation 1 . . . 22
3.10 Form Validation 2 . . . 22
3.11 Edit profile info 1 . . . 23
3.12 Edit profile info 2 . . . 24
3.13 Client main dashboard . . . 24
3.14 Notification 1 . . . 25
3.15 Notification 2 . . . 25
3.16 Notification 3 . . . 26

4.1 Client/server reference architecture . . . 29
4.2 Distributed database servers . . . 30
4.3 Dispatcher.ml database . . . 31
4.4 Algerie Poste API 2 . . . 32
4.5 Infinityfree Hosting service . . . 32
4.6 Domain name registration . . . 33
4.7 Domain name linking 1 . . . 33
4.8 Domain name linking 2 . . . 33
4.9 Domain name linking 3 . . . 34
4.10 Page 403 ERROR . . . 34
4.11 Locating .htaccess file . . . 35
4.12 IP range of Algeria . . . 36
4.13 Disabling media access . . . 36
4.14 How SSL/TLS works . . . 37
4.15 Changing Namerservers . . . 37

4.16 Activating SSL/TLS on Cloadflare . . . 38
4.17 Uncompressed landing page . . . 38
4.18 Compressed Landing page . . . 39
4.19 PHP code for index.php . . . 39
4.20 Code inside htaccess file . . . 40
4.21 Network Tab in Chrome . . . 40
4.22 prepared statements examples in PHP . . . 42

5.1 Software validation . . . 44

B.1 Client account Table . . . 55
B.2 Achets Table . . . 55
B.3 Customer Table . . . 55
B.4 Delivery Table . . . 56
B.5 Delivery images Table . . . 56
B.6 Items Table . . . 56
B.7 Category Table . . . 56
B.8 Output Images Table . . . 56
B.9 Products Table . . . 57
B.10 Profile Images Table . . . 57
B.11 Statistics Table . . . 57
B.12 Types Table . . . 57
B.13 Wholesalers Table . . . 57
B.14 Wholesalers Images Table . . . 58

Chapter 1

Introduction

In this chapter we talk about the project on general an how the idea came to be realized, in the first section **Background** we talk about briefly about the background of us "students" of the project and the project idea on general,This section has two subsections in the first subsection **Detecting the problem** we talk about how we saw the problem and the troubles that can be tackled and solved by technology, the next subsection **Solutions used in the market** talks about the solutions applied in the Algerian market with their share of problems and difficulties and how customers are dealing with it.

The second section **Obstacles** talks about our own problems and difficulties we faced during the realization of this project. this section has also two subsections first **Legacy IT software** where we discuss how our clients are using many old technologies and old IT equipment in their daily work. and in the second subsection **Unreliable payment methods** we talk about the problems of online payment methods in Algeria and present the current solution on the market.

1.1 Background

We are students in the "Djillali Liabes University" studying Computer Science and share friendship and love to IT and problem solving of problems using knowledge we gained from our journey as students. both of us came from very different backgrounds as one was a military student studying Aviation and the other was a Finance student abroad. but differences has brought strength to the table as different point of views and different way of thinking can open many subjects and bring new ideas to any problem we face.

1.1.1 Detecting the problem

Before joining University one of us was working in a Fast food shop in Sidi Bel Abbes for some years and saw that restaurants and fast food chains share a very irritating problem on a daily basis as they need to shop for their needs from vegetables to meat to bread, this takes hours every morning from their time and every day has its problem with their suppliers from not opening early to not having the needs of the clients, some have found solutions by dealing with suppliers that work with Restaurants only and do a daily delivery to the clients but they are usually specialized in one thing only like vegetables or bread, and the client still find himself dealing with at least 6 different suppliers daily.

Another problem is how to keep track of the daily purchases and track growth and spending by weekly or monthly basis. some use log books and keep recites and do the calculations manually at night or at the weekends. which takes time and can suffer from human errors.

1.1.2 Solutions used in the market

As mentioned in the last Section some business owners have found solutions to save some time but not as effective as they hoped for, we will discuss some of those solutions on the market. First is phone ordering from suppliers and using delivery services to bring their needs, the first problem with this solution is that they have to deal with different suppliers for each of their needs for example bread from the bakery, soft drinks from drinks suppliers, vegetables and meat from other suppliers and finally packaging suppliers now they need to contact these suppliers on a daily basis if they decide each day has its needs, others decide on an average of each need and order this as a fixed daily order, and only call the suppliers if they need to modify the daily order, this can save them some time but takes from the flexibility of ordering what they need only, since not all days are equal and holidays tend to bring more customers than normal days, not forgetting the climate role in visitors per day, the other problem with this method is logistics cost and overhead as they have to pay for every separate delivery an amount which can quickly eat from their budget. but it saves them time spent daily on shopping from the suppliers which is in average 2 hours every morning and in businesses time is sometimes far more valuable and business owners are willing to pay in order to save every minute. another problem is accounting and how to keep track of their daily spending, some are using traditional methods like text logs and recites and do their calculations every night or at the end of the week. but we saw others who use some phone apps or computer software to do the accounting part, these days phone apps seem to be more convenient and faster to record any transaction, **BevSpot** and **7shifts** are two of the well known apps used right now.

1.2 Obstacles

In this section we will be talking about some of the obstacles we faced during the development of our platform, we will not discuss the short time we had to develop our solution or the difficulties we faced during our studies and exams, but mainly obstacles that forced us to take other traditional solutions than the solutions we wanted due to the IT infrastructure of the country and the legacy hardware and software on our customers machines.

1.2.1 Legacy IT software

Our by far biggest problem was that our clients use very old computers with very limited resources and they got very old operating systems that have no mores support from the parent company Like Windows XP, Windows Vista, Windows 7. not to mentions some are 32-bit and some are 64-bit. So developing a computer software that can run on all of those machines was out of the question and we had only one choice is to build an online App to, since as long as they have a web browser and an internet connection the website will run on all those machines just fine. some web browser companies like Firefox and Opera still make up to date web browsers for these legacy machines and OS's and even 32-bit versions which is great for us and even if the client had an older web browser he would still be able to browse our web-app as we have made sure to use Technologies that are very well supported and avoided using any new wen technology which may have lesser support on old web browsers.

1.2.2 Unreliable payment methods

FIGURE 1.1: Algerie Poste API

Another problem we faced is the lack of real infrastructure and even lack of acceptance from customers to online payment methods, as for infrastructure the only reliable payment method is the dhahabia card from Algerie Post, which has 2 problems first it is only available to Algerie Post customers and not available to any other banks, which makes it useful only for a very limited number of clients.

The other problem is the cost of using the API is very high and implementing it would cost us with no real actual benefit as we will see next.

The last problem with online payment is that people are not trusting it, and still fear losing money.

Due to these problems mentioned above we decided to use a traditional technique and adopted widely in Algeria which is "Pay as you go" and only pay when they delivery is made.

Chapter 2

Methodology of the solution

In this chapter we have 2 sections, In the first section we discuss the solution using UML language to create the important use cases in our application and the sequence diagrams to explain the time frame of the important actions in our application.

The second section explains the design patterns we have chosen in developing and why we have chosen that pattern and how it works with the Programming language we used.

The realization viewpoint of a thin Web client architecture can vary greatly, depending on the scalability strategy. In fact, the specifics of the entire system's architecture tightly hinge on this strategy. Discussing and describing all the ways a system can be architected to scale is definitely beyond the scope of this book. At the highest level, we can describe some points in the realization viewpoint that are common to all thin Web client systems. Despite the efforts of the W3C to define browser interface standards, the reality is that browser vendors' implementations of this interface vary. Additional issues are related to each vendor's definition and implementation of the event mechanism.[8]

2.1 UML Case Studies

The analysis of problem domain and design of desired solution within software development process has a major impact of the achieved result developed software. While the software developer community uses a set of tools and different techniques to create detailed specification of the solution, the proper analysis of problem domain functioning is ignored or covered insufficiently. One of such techniques is object-oriented software analysis and development which states that there are two fundamental aspects of systems modeling: analysis and design. The analysis defines what the solution needs to do within the problem domain to fit the customer's requirements, and the design states how the solution will be implemented. The design of object- oriented software is leaded by the Unified Modeling Language (UML). UML is an approved standard modeling notation for visualizing, specifying, constructing, and documenting the artifacts of a soft- ware intensive system. While the UML has elements for designing and specifying artifacts of a software system, it lacks the ability to document the functioning of a problem domain by using computation independent constructs. To solve the previously mentioned UML issue, a new extended version of UML is developed Topological Unified Modeling Language (Topological UML). Topological UML is a combination of UML and formalism of Topological Functioning Model (TFM). It captures system functioning specification in the form of topological space consisting of functional features and cause-and- effect relationships among them and is represented in a form of directed

graph. The main aim of improving UML is by transferring topology and mathematical formalism of TFM to UML thus strengthening the very beginning of the software development lifecycle. Sometimes it is very hard to pay appropriate resources and time at the very beginning of the software development lifecycle to detect and analyze aspects of desired software system as much as possible. If we pay appropriate attention at the beginning of the software development project, we tend to avoid wasting valuable resources, including time; otherwise it could lead to unnecessary reworking or even recoding parts of the system or the system as whole. Just like Benjamin Franklin has said: ***If you fail to plan, you are planning to fail!***. [28]

Flowcharts were the initial attempt at modeling software, occasionally interspersed with coding sheets. The model-view-controller (MVC) pattern provided the initial basis for structuring a computer program. SSADM followed, with a focus on analysis as well as design, but it was still used for procedural approaches. The 1980s saw an explosion of relational databases and their modeling based on Chen's highly popular entity-relationship diagrams (ERDs) and data flow diagrams (DFDs). When languages like C++ and Java became popular together with object orientation, many modeling approaches came to the fore eventually culminating in the UML. Modeling approaches are still evolving to incorporate software solutions for mobile applications (and IoT), those that deal with Cloud computing, and the use of Big Data analytics and services.[34]

While the UML is a notation and as such its specification does not contain any guidelines of its application during software development process, the UML modeling driven methods fulfill this gap. Unfortunately, not every UML modeling driven method covers all the software development lifecycle. In addition usually only a small part of UML diagrams is used to specify both problem and solution domains. Due to the partial UML and software development lifecycle coverage and the fragmentary application of UML diagrams the soft- ware developers are forced to combine UML with several modeling methods and techniques (instead of taking UML as a notation and one UML modeling driven method) thus the application of UML gets more complicated and incomprehensible. To address this issue the developed UML extension is provided together with a proper modeling method Topological UML modeling. Topological UML modeling for problem domain modeling and soft- ware systems designing is a model-driven modeling method. In the con- text of Model Driven Architecture (MDA), the Topological Functioning Model (TFM) considers problem domain information separate from the solution domain information and holistically represents a complete functionality of the system from the computation independent viewpoint while Topological UML has elements for representing system design at the platform independent viewpoint and platform specific viewpoint. The Topological UML modeling method covers model- ing and specification of systems in computation independent and platform independent viewpoints. Problem domain analysis and soft- ware system design with Topological UML modeling method consists of six activities the first one is problem domain functioning analysis followed by behavior analysis and design, structure analysis and design, state change and transition analysis, structuring logical layout of design, and concluding with components and deployment design.[28]

A use case diagram is a model of the requirements of a system at a high level. Use case diagrams are primarily used to visualize use cases, corresponding sectors, and their interactions. The diagram itself is not a use case but rather a visual of actors and a group of related use cases. Visual models of use cases facilitate understanding the business processes and aid in communication with stakeholders. The specification and documentation of the use cases shown in the use case diagrams

form the crux of requirements modeling. Use case diagrams are behavioral-static in nature. This is because they help organize and evaluate the requirements of the system in the problem space. The behavioral aspect of the requirements is not visible in the use case diagrams. Because the relationships between two use cases, or between actors and use cases, do not represent the concept of time, use case diagrams are categorized as static diagrams. Therefore, care should be taken to consider use case diagrams as depicting the flow or behavior of a system. The flow of a process is part of the textual documentation within a use case and the corresponding activity diagrams.[34]

2.1.1 Use-Case for the wholesalers

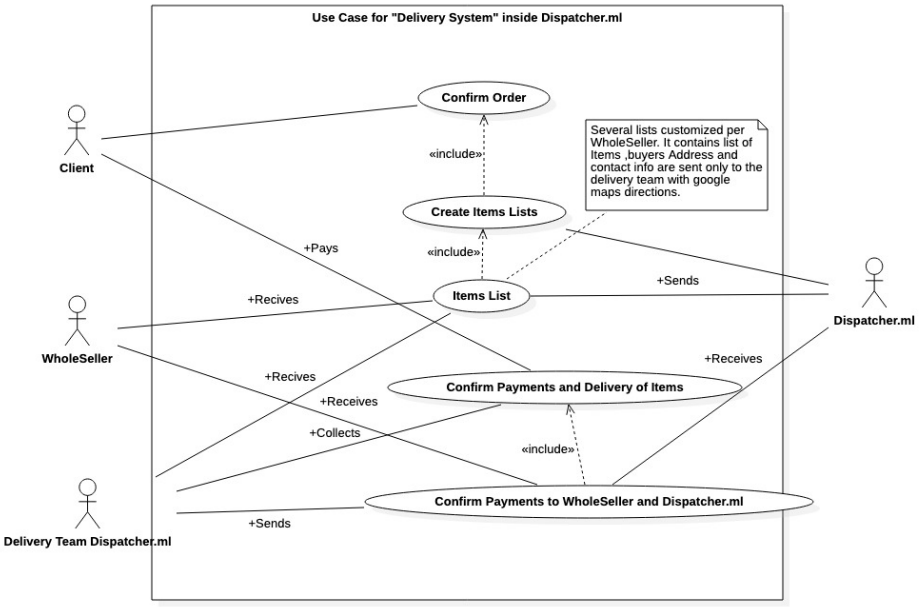


FIGURE 2.1: WholeSalers Case Study

The wholesaler main tasks in the use case are **selecting** his stores where he wants to **Add** items to it, he can also **Fill** details about the items like name quantity, price, expiration date and other details. after finishing building the store he can **confirm** his store items to be displayed to customers.

It is noted that during **adding** items he can **Select** an existing category in the database or**Add** a new category if the desired category doesnt exist. he can also **Add** certain items to the promotions section for publicity.

2.1.2 Use-Case for customers

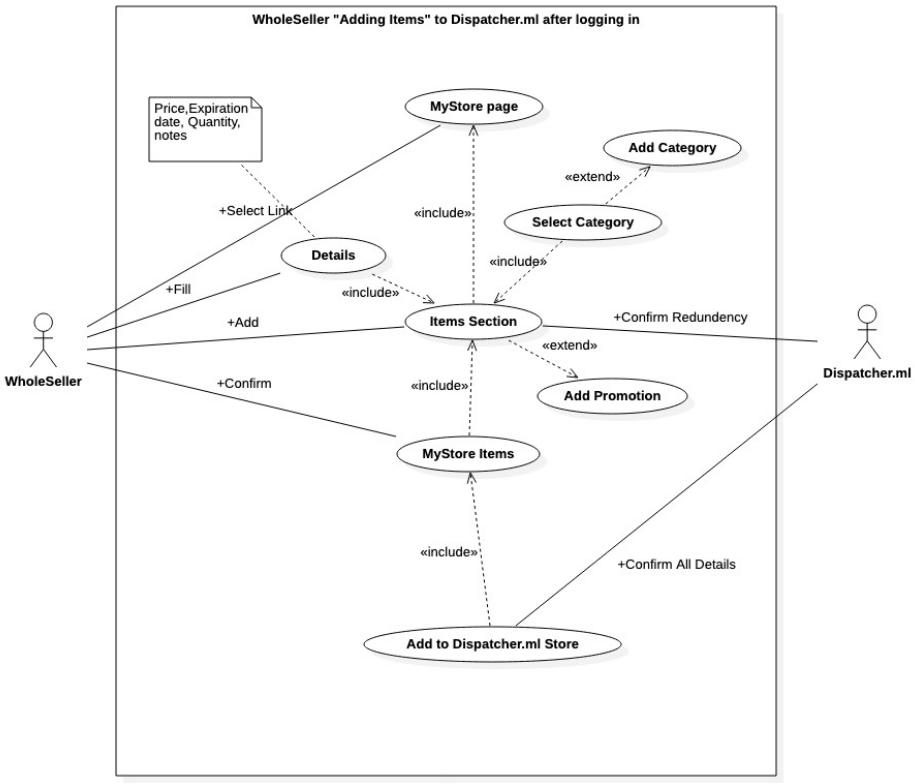


FIGURE 2.2: Customers Case Study

The customers use case diagram is probably the most important use case diagram as it showcases the most important use case and actions of the client Actor will trigger all other Actors in our web-app, here we are only showing the core use cases of the client as we can see he can interact with our web-application to **Browse** items in the website, he can **Add or Modify** items to the cart and he can **select** a delivery method then **confirms** his order and **select** delivery time to end his action by **Pay** through payment option.

Secondary Actors are Wholesaler and the Delivery team, both rely on actions triggered by the client, after the client confirms the order and delivery method the whole seller can **select** pick time to to end with the delivery team **accepting** the delivery. at the end of his job both him and the wholesaler can **collect** the payment.

It is noted that Items can include promotions **selected** by the wholesaler.

2.1.3 Use-Case for delivery system

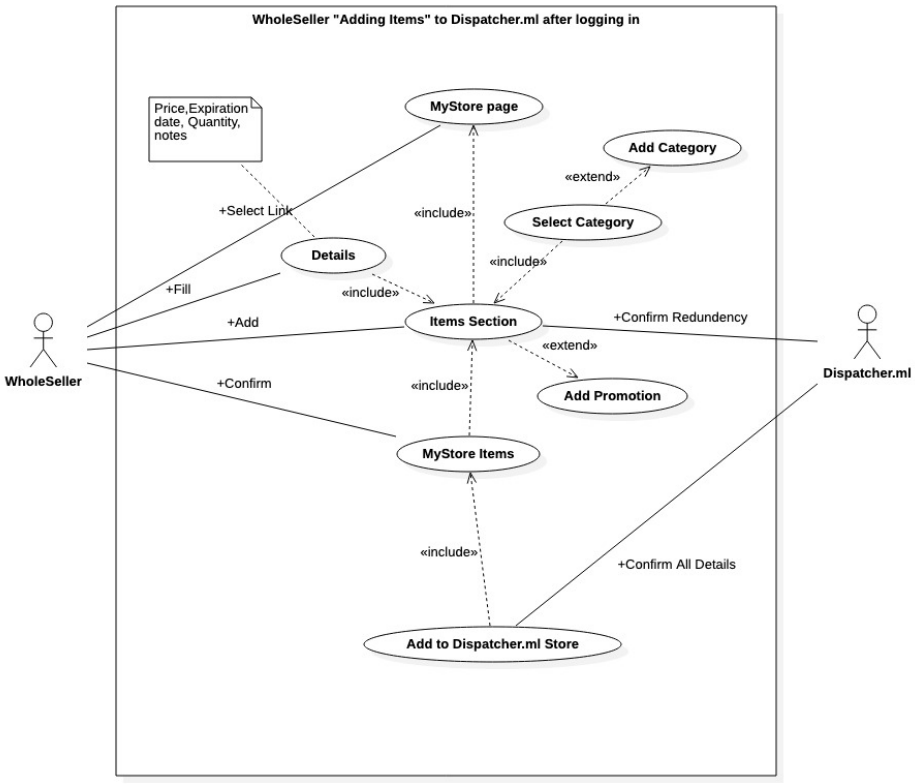


FIGURE 2.3: Delivery System Case Study

Despite that this use case is about the delivery team, since his actions can only triggered by other actors he can't be the main actor in our delivery use case diagram. as we can see after the client **confirms** the order the wholesaler and the delivery team **receive** notifications about the order where the delivery team can **confirm** collecting items and then the client should **pay** for the items, then after that the delivery team can **send** confirmation of payment from the client to wholesaler through Dispatcher.ml.

2.2 UML Sequence Diagram

A sequence diagram is an interaction diagram that emphasizes the time ordering of messages sent between objects. It shows a set of objects or roles and messages sent and received by them. Sequence diagram has two features that distinguish them from communication diagrams presence of lifeline and focus of control. The sequence diagram is included in UML specification since the first (1.1) version.[28]

Sequence diagrams have been popular ever since Jacobson introduced them as a means of documenting the behavior within use cases. In earlier uses, sequence diagrams were also called scenario diagrams, because they represented, pictorially, a scenario (or an instance) within a use case. Because of their practical ability to show what is happening "inside" a use case, sequence diagrams are popular with both business analysts and system designers. Each step within a use case appears on the sequence diagram as a note or a narration. Sequence diagrams represent the detailed interaction between actors and a system or between collaborating objects within a given time block. However, information as to what happened before the interaction

started and what happens after the time block stops is not shown in the sequence diagram. While messages shown in the sequence diagram can have preconditions and postconditions, these conditions are not directly visible in the diagram. Despite this limitation, the "time" appearing in the diagram is far more precise than in the activity diagram. Therefore, it is possible to show what happens between two messages and to ascertain what happens as time progresses. The sequence diagrams are thus considered dynamic-behavioral in nature.[34]

2.2.1 Sequence Diagram for the Client order

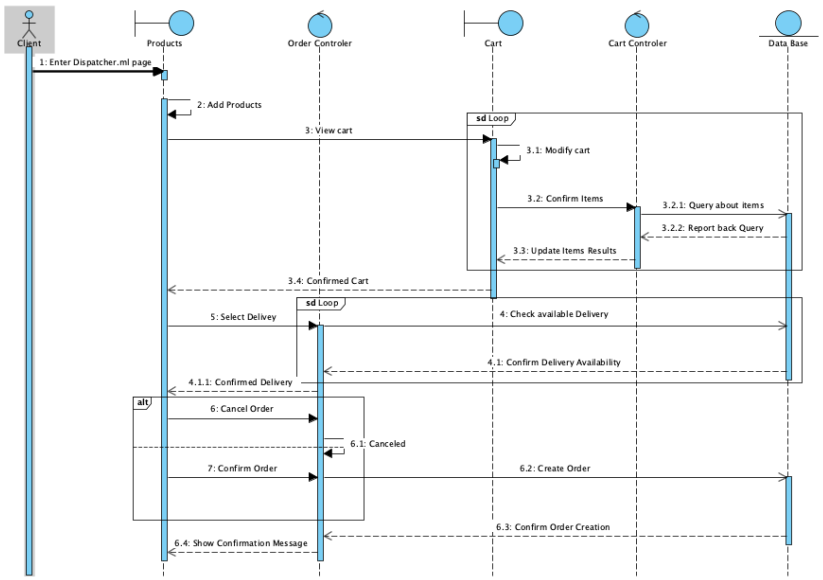


FIGURE 2.4: Sequence Diagram

In our sequence diagram we are showcasing the flow of the messages and the time for each action from its start to its finish. we have the Client as the actor, we have two interfaces the products interface and the cart interface which both represent web pages in our web application, the controller modules are also for the order and the cart, then we have our database as an entity object that responds to request or to be more precise **queries**. the steps are as follow :

1. **Add Items** during the browsing process he can add items to the cart.
2. **Cancel** This is a choice option that will clear the order.
3. **Confirm** This is a choice option as the client can confirm and using the order controller an new record for an order is added to the database. after a successful transaction the order controller will respond with a message of confirmation to the products interface.
4. **Select Delivery** The client now selects the delivery method and delivery time, using the order controllers it performs **4.1** a check on the database to see if the delivery options are available to the customer and if the time of delivery is possible to both the delivery team and customer the database responds **4.2** to the query with results which will be either accepted or rejected. this is a loop process so the client will be here until he is completely satisfied.

5. **Select the products page** here the user has just selected the products.php page which shows a list of products the client can choose from.

6. **View the cart** during this step he is presented with cart.php page that has other features like **3.1** Modify the cart or **3.2** confirms the Items. during this stage **3.2.1** a call to the database is done by the cart controller to check items availability and **3.2.2** report back to the controller which in it self will update results in the cart interface and products page, it is noted that this process can be repeated as long as the client sees necessary

2.3 Deployment Diagram

Deployment diagrams are structural and static in nature. Unlike all other diagrams in the UML, deployment diagrams are the only "hardware" diagrams in UML. They show the organization of processing nodes to enable deployment of the software system. They also show the components that are executed on these nodes. Being a hardware diagram, the deployment diagram provides a valuable foundation for communicating hardware-related decisions. Deployment diagrams enable discussion of the operational requirements of a system including the ability of the system to handle speed and volume, location and security of the nodes, and the manner in which the executables are deployed across the network.[34]

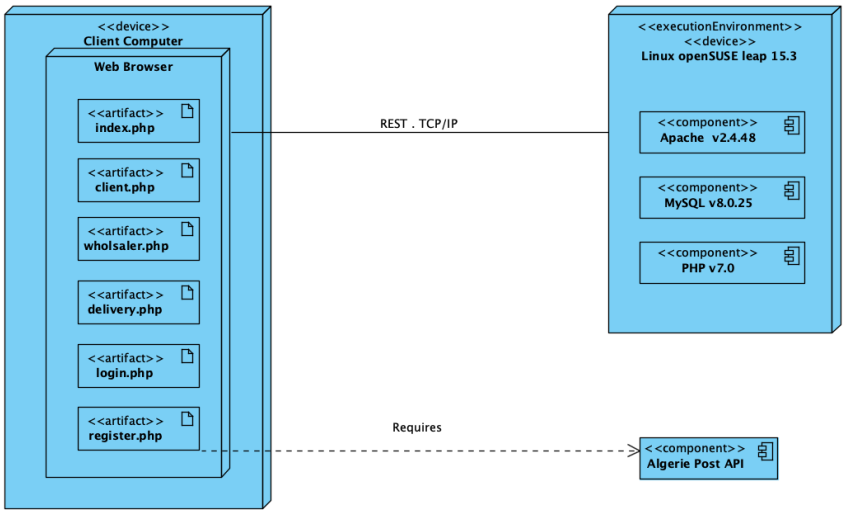


FIGURE 2.5: Deployment Diagram

Here we are showing the structure of our system, as we can see in figure, we are using a very simple client/server approach, the only addition is the Algerie Post API which we are using to verify the client during registration period.

2.4 Choosing the right design pattern

The rationale for the use of patterns is very simple. We face problems when we develop software. We might face problems on making decisions about which algorithms to use, what is the most suitable design, what techniques to use, and what modules to use. The chances that the same kind of problem has been encountered by some other software professionals are very high. If someone else has faced the

same problem before us, the chances of them having solved the problem is very high. If someone already solved the kind of problem that we are trying to solve, we are better off learning from that solution which is already available, rather than trying to reinvent the wheel.[1]
Every problem is unique. Therefore, every solution should also be unique. So how can we use someone else's solution to our problem? The rationale for the use of software patterns is not about picking up ready made solutions to problems. Rather, the patterns guide us on how to approach a problem. In the previous chapter, we discussed dividing and conquering a problem. When we divide a larger problem into smaller manageable pieces, those smaller problems can seem familiar. For example, we might need to sort a list of elements, before displaying them. We can use a sorting algorithm here, and there are plenty of implementations of sorting algorithms out there.[1]

The sorting algorithm is more of a ready made solution. When it comes to design, the patterns can guide us in the right direction in terms of what is to be done. For example, the MVC pattern helps us to figure out how to separate out presentation from business logic and data. That is just a guideline. We can use the guidelines and implement our solution based on that. Software patterns help us reuse known solutions to common problems, and then customize those to our needs.[1]

2.4.1 What is the problem with PHP ?

The problem is that PHP is just too easy. It tempts you to try out your ideas and flatters you with good results. You write much of your code straight into your web pages, because PHP is designed to support that. You add utility functions (such as database access code) to files that can be included from page to page, and before you know it, you have a working web application.[37]
PHP's phenomenal popularity meant that its boundaries were tested early and hard. As you will see in the next chapter, PHP started life as a set of macros for managing personal home pages. With the advent of PHP 3 and, to a greater extent, PHP 4, the language rapidly became the successful power behind large enterprise websites. In many ways, however, the legacy of PHP's beginnings carried through into script design and project management. In some quarters, PHP retained an unfair reputation as a hobbyist language, best suited for presentation tasks.[37]
Over the years, PHP 5 continued to evolve and improve, incorporating important new features such as namespaces and closures. During this time, it secured its reputation as the best choice for server-side web programming. PHP 7, released in December 2015, represented a continuation of this trend. In particular, it provided support for both parameter and return type declarations—two features that many developers (together with previous editions of this book) had been clamoring for over the years. There were many other features and improvements including anonymous classes, improved memory usage, and boosted speed. Over the years, the language grew steadily more robust, cleaner, and more fun to work with from the perspective of an object-oriented coder.[37]

2.4.2 What is MVC desigen pattern ?

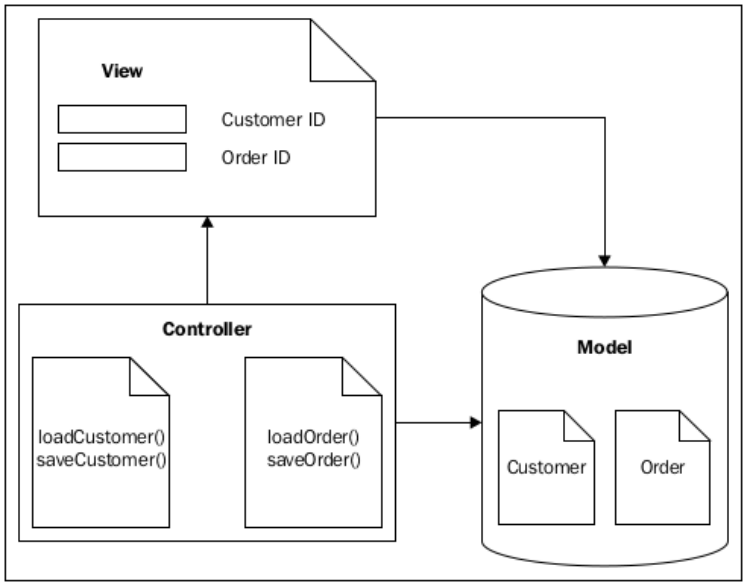


FIGURE 2.6: MVC pattern

Model: It represents the data on which the application operates.

The model represents the data on which the application operates. The model manages the behavior and data of the application domain. It responds to requests for information from the view and responds to instructions to change information from the controller. In the context of PHP, the model corresponds to the database schema. The database management system plays a key role in PHP applications when it comes to data persistence.[1]

View: It renders the data from the model into a form suitable for interaction, typically a user interface.

The view renders the data from the model into a form suitable for interaction, typically a user interface. In other words, the view manages the display of information. In the context of PHP applications, the view corresponds to the HTML-based presentation that is delivered to the user, to be displayed with a web browser.[1]

Controller: It responds to events, typically user interactions, and may invoke changes on the model

The controller responds to events, typically user interactions, and may invoke changes on the model or the view. In the context of PHP applications, the controller is the actual PHP code that deals with the business processing. It also couples with the HTTP logic—given that the PHP application's main delivery channel to the users is HTTP.[1]

For many PHP applications, MVC is the most useful pattern. This is because PHP is used for web-based applications. However, they are not just web pages, but are applications based on backend business logic. There are numerous other patterns such as observer/observable, iterator/for each, and handler chain, which can be of use in the design of the system.[1]

Applying a pattern just saves time and effort. We need to be aware of the problems and solutions that the patterns try to address in general, to make use of patterns effectively. A pattern is a description of a problem and the potential solution to it. There is no standard answer for all problems that we find when working on a software project. So it helps to be aware of the problems, and be capable of adapting

the solution suggested by the pattern to suite the problem at hand. Many software professionals have used patterns over the years and the patterns are proven to work. You might have used patterns without knowing that it is a pattern, but not knowing the principles would lead to creative chaos. It is always advisable to spend some time and learn the principles behind the software patterns for any developer. Having the knowledge about problems that can appear, and the potential solutions to those problems, adds to the experience of a good PHP developer. The effort spent on understanding the patters would help the PHP team members in the long run. It would even be a good idea to train the entire team on patterns with hands-on examples in the early stages of a project.[1]

2.4.3 How MVC can help ?

MVC is a very good example of separation of concerns. It helps us to break down a system into view (presentation layer), controller (business logic layer), and model (data layer). Let us look at a simple example. Assume that we want to store data about people in our system. We want to store the name and age of these people. Rather than the age, we will store the date of birth this makes the application more agile over the years. As people grow older, we will not need to change the database content. Suppose we want to list all of the teens in the system. For this, we query the database for the date of birth, compute the age of the person at business logic layer, and present the view to the user. If the user wants to sort every one by age to locate the youngest ones, the user should be able to do that on the view—click on the title of the age column, and the sorting will be done.[1]

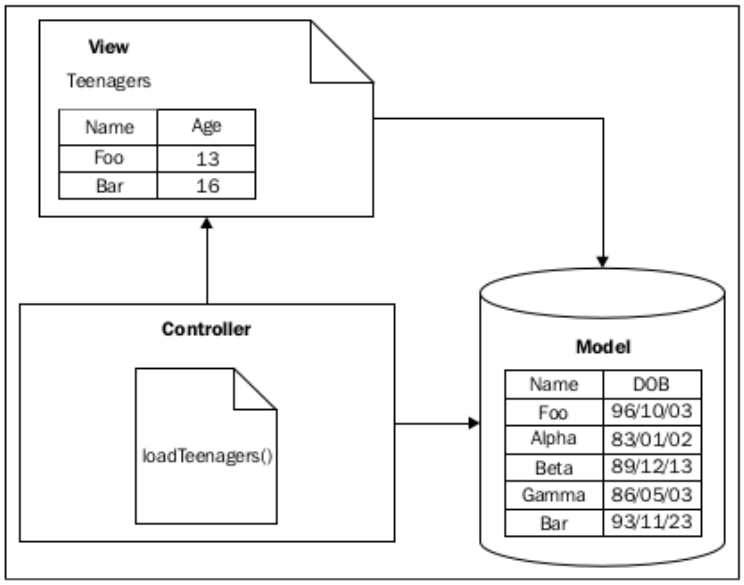


FIGURE 2.7: Model View Controller II

The preceding image shows the mapping of MVC into a real implementation. The model stores names and dates of birth for the people. The controller loads the teenagers form the model, based on the simple logic implemented in the loadTeenagers() function. The data set provided by the controller is then presented to the user. When the user is looking at the view, he might want to sort the list either by the name or by the age. So the user will click on the column header. Based on the

column clicked, the presentation layer, that is the browser would be able to come up with the correct sorting using some AJAX magic.[1]

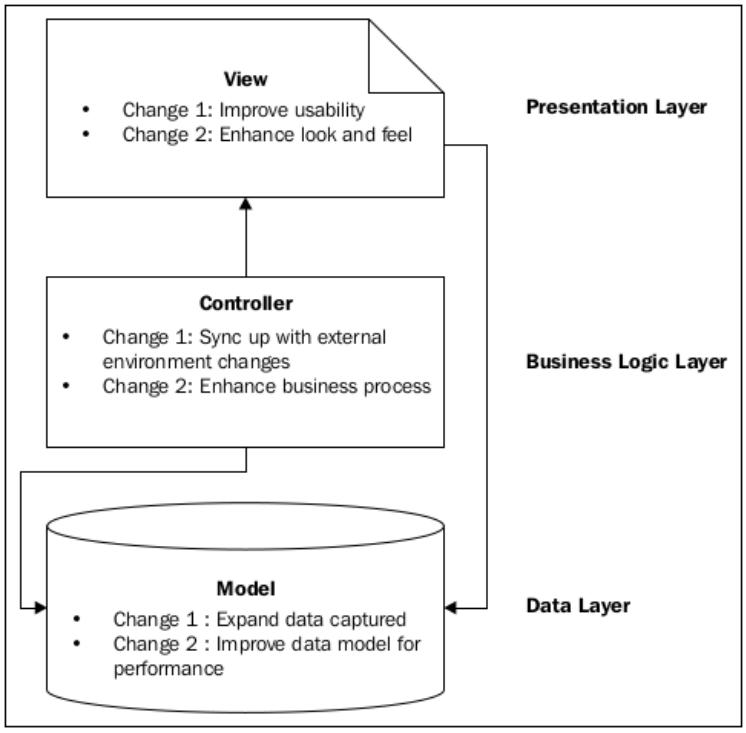


FIGURE 2.8: MVC Buisness Logic

Dealing with change is one of the key challenges in any software system. Therefore, PHP-based systems also need to face that reality. Changes can be of many forms and can happen at any layer of the software system. As we can see, the MVC pattern helps a great deal when it comes to changing, enhancing, and evolving the system over time.

The MVC patterns help us to easily identify the areas of change and facilitate the change requests from system users. This is very useful in any software system, irrespective of the scale of the system. In addition to helping us locate the piece of logic to be changed, an indirect advantage of MVC is that by isolating changes to a particular layer, it prevents regression issues in the system. One of the key challenges in changing a working system is the difficulty of making the change without breaking the existing functionality. If not for MVC, we might have a situation where we have mixed up the business logic, presentation, and data access. Therefore, a change done, say to improve presentation, cannot be guaranteed to not have affected the other areas. In most of the systems, it is the business logic (in other words, controller) and the presentation layer (also known as view) that are likely to change more than the data model. In comparison to business logic, presentation is far more likely to change on a regular basis.[1]

Chapter 3

Front End Development

Front-end web development is the practice of converting data to a graphical interface, through the use of HTML, CSS, and JavaScript, so that users can view and interact with that data.[10]

3.1 General website theme

Discovering requirements focuses on exploring the problem space and defining what will be developed. In the case of interaction design, this includes: understanding the target users and their capabilities; how a new product might support users in their daily lives; users' current tasks, goals, and contexts; constraints on the product's performance; and so on. This understanding forms the basis of the product's requirements and underpins design and construction.[30]

It may seem artificial to distinguish between requirements, design, and evaluation activities because they are so closely related, especially in an iterative development cycle like the one used for interaction design. In practice, they are all intertwined, with some design taking place while requirements are being discovered and the design evolving through a series of evaluation redesign cycles. With short, iterative development cycles, it's easy to confuse the purpose of different activities. However, each of them has a different emphasis and specific goals, and each of them is necessary to produce a quality product.[30]

The development of a website involves far more than just its design. There are a lot of pre-design activities concerned with establishing the purpose of the site, who it is aimed at and how it fits into the organization's overall digital strategy. In larger organizations there will be plenty of disagreement and arguments about all these issues and these internal politics often affect the final quality of the site. Many sites finish up as too large, trying to serve too many issues with the marketing people in charge; usability and engagement come a long way down the list of priorities. At the other end of the process the launch of the site has to be carefully managed and other infrastructure issues will need to be addressed, such as how, when and by whom the content is written and updated, who deals with e-mails and site maintenance, and so forth.[5]

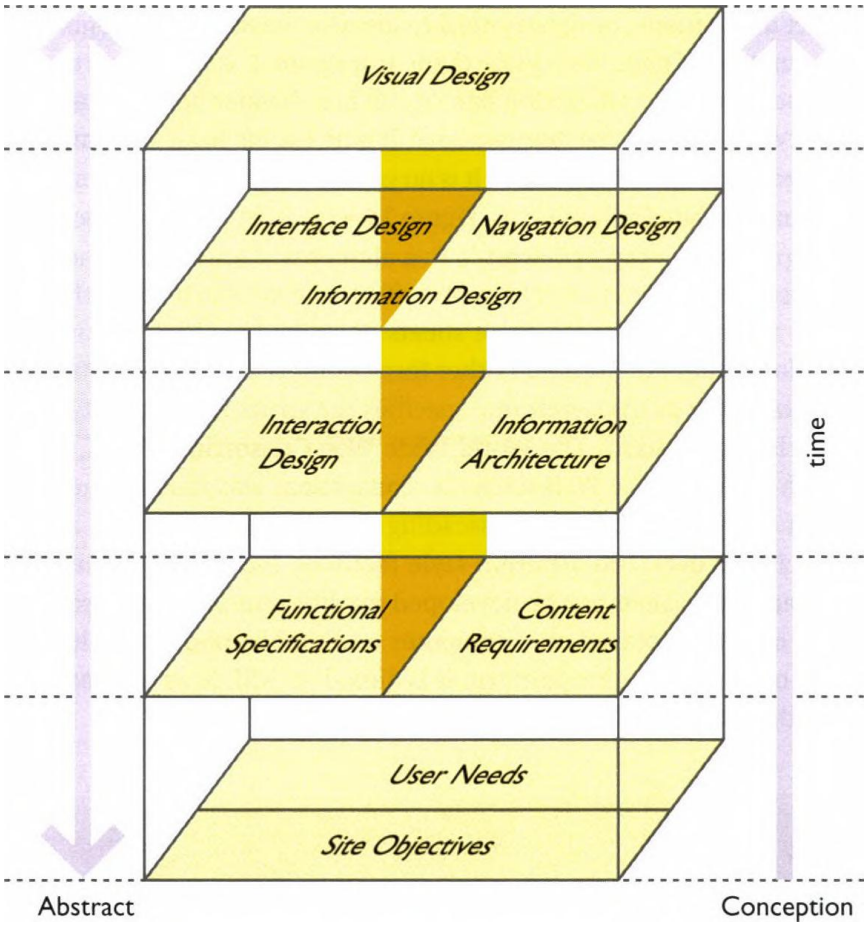


FIGURE 3.1: Elements of user experience

[5]

Vital to the success of a website, of course, is the content. In website design the designer has to acquire another skill of writing and organizing information content. In any organizations someone else might work with the designer to help. Many web sites are seriously overloaded with content and try to serve too many different types of customer. A university website will often try to cater for potential students, existing students, academic staff, administrative staff its own and from other universities, business partners and so on. Trying to accommodate all these different user groups results in an unruly and rambling site, making it difficult for any one of these groups to be satisfied. The same is true of large corporation and public service sites. A detailed PACT analysis and developing personas will help to identify the needs of different user groups.[5]

The design of websites should follow the principles of good interaction design. Designers need to know who is going to use the site and what they are going to use it for. Websites need to be well focused with clear objectives. They should develop personas of the people whom they expect to be visiting the site and understand clearly what goals they will have when using the site. The design phases of understanding, envisionment, design and evaluation need to be undertaken. Scenarios of use should be developed, prototyped and evaluated.[5]

Even if a site is well focused, it will soon get large and so issues of how to move around a website become important; navigation is a central concern here. Support to enable people to discover the structure and content of the site and to find their way

to a particular part of the site is the key issue. Information architecture is an area of study devoted to designing websites and helping people to answer questions such as: Where am I? Where can I go? Where have I been? What is nearby? Navigation bars at the top and down the side of the Web pages will help people develop a clear overall map of the site.[5]

It is also vital to pay attention to the design principles. Consistency is important and a clear design language should be developed, including interaction patterns for the main recurring interactions. If it is not desirable to use the standard blue underlined links then ensure that links are consistent so that people will quickly learn them. Many sites confuse people by not making links sufficiently visible and distinguishable from other text in the site.[5]

3.1.1 Colors

Choosing the right colors for a website or a logo can be a perplexing and time-consuming task, unless you have the right knowledge of colors. Colors play a pivotal role in the success of some businesses and can make a huge impact on their revenue. In this blog, we will learn how to pick color schemes that are not only pleasing to the eye but will also give users a psychological feeling they can experience when consuming your product.

The color pallet used in this project are: background: #191972 gradient colors: rgba(0,212,255,1), rgba(18,9,121,1), rgb(7, 3, 46), rgba(233,3,194,1) those fancy colors are one of the most favorite colors by users, those colors are attractive in the goal of make the website looks modern and stylish, this will help to bring users and give them a nice experience to.

Blue: It is the color of trust, reliability, and calmness. It is used by websites like Facebook and Intel to try and win the trust of its users. It is that calm feeling you get from the blue that makes you endlessly scroll through Facebook feeds. The negative part of using too much blue is that it can also show sadness, so designers make sure that color is balanced out.

Purple: Purple represents royalty, wealth, luxury, value, and bravery. Purple can be a difficult color to work with and can have a negative effect when the wrong shade is used.

3.1.2 Fonts

Long before the age of technology, fonts have been evolving. We've come a long way since scriptwriting, mainly because our works are no longer in scrolls but anywhere from screens to print ads. Fonts have changed with our times and the means of which content was presented to us, yet that doesn't mean all fonts have gone outdated. Some of them outlasted the test of time and are still used today! Finding that perfect font can often feel like locating a needle in a haystack, however, it's important to remember that before we dive into that haystack, we need to understand a few of the basics of font pairing.[6]

Typography often makes up about 90% of the design, and it is a communication art. When someone visits your website, they may not care much about the graphics, but surely they will read the content. Hence it is important to balance the graphics and text on the website. Many web developers and web designers ignore or never think about the typography it is necessary to understand that it's the psychological effect of conveying a message to an audience. Before reading the text, people judge

the seriousness of the text based on the typography. Ninety-six percent of the Internet comprises written information or text. This text is distributed into big and small chunks of information that are spread over different pages with different formats. The formatting of all this written information in a website's design is what is known as 'web typography'. Typography is the most important part of web design because it influences the way that a website communicates with readers. From larger headlines and bold blocks to a small body, the various uses of text on the web are the core communication method. So, it's natural that how a web designer decides to arrange type on a web-page is crucial to the effectiveness of that site. We see its usage everywhere, as all websites rely on typography to convey their message to the readers. However, some web designers tend to overlook the inevitable importance of typography and focus only on the visual or graphical part of the web design.[6]

3.1.3 Logo and animation sketches

These are some sketches for the services that are provided in the website. they were drawn by hand before digitizing them in tablet in SVG form for usage.

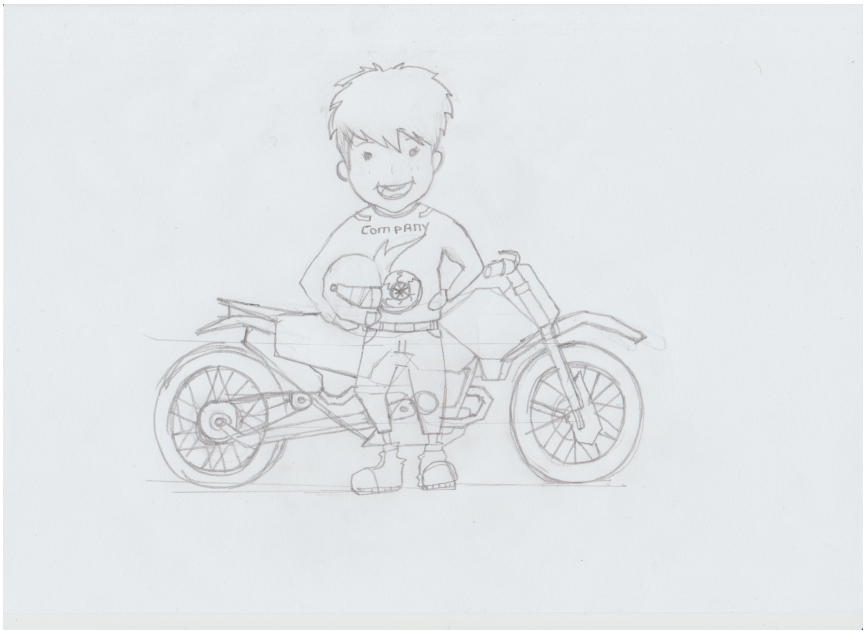


FIGURE 3.2: example of a sketch

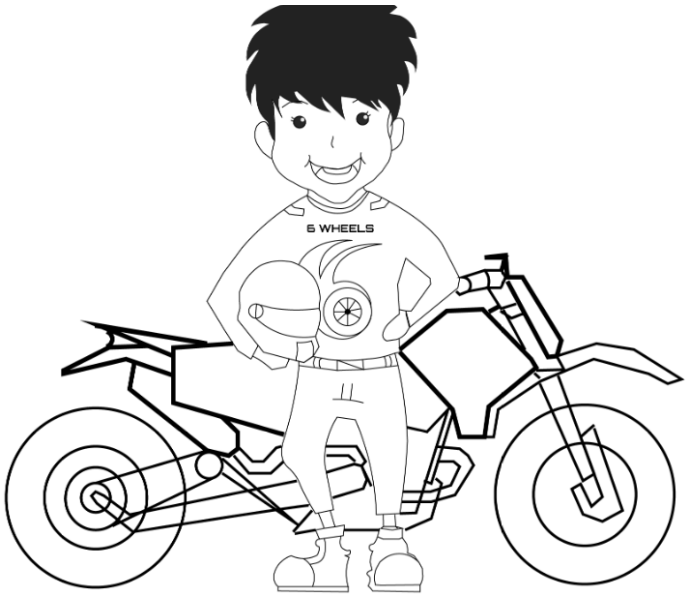


FIGURE 3.3: digitized sketch

3.2 Landing page

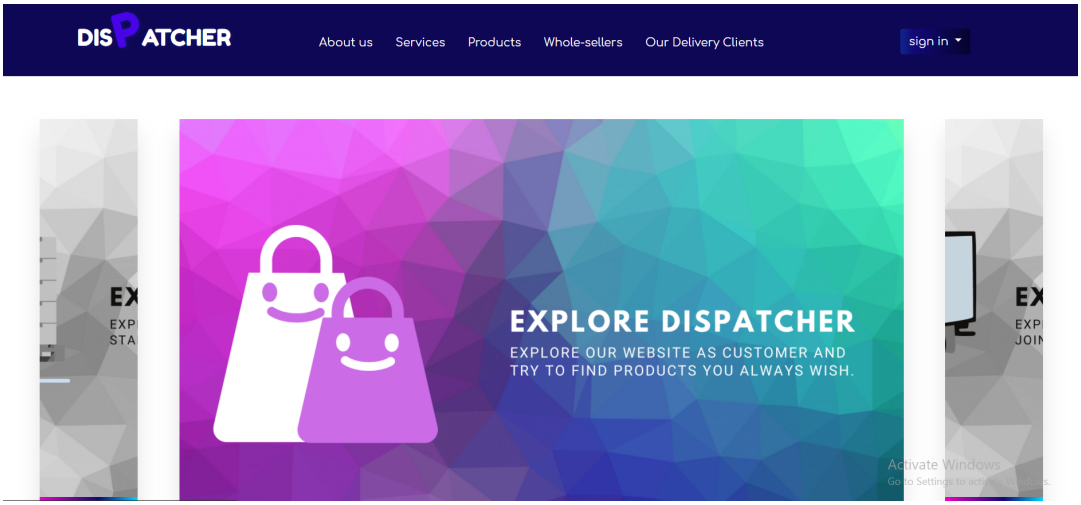


FIGURE 3.4: Landing Page

The landing page is one of the most important pages on any website, the landing page in general must contain all necessary stuff to introduce to the user all information needed to accomplish the services that the user is searching for.

The landing page is the entry point to the application that's why it must be manageable and be able to describe perfectly the web application.

The key to steal users Interest is a nice-looking website, this depends on real studies that have been made in the last years in human interaction technical and design creativity, we can say that front-end development is the key for any application to take the crown on the markets.

'DELIVERY' Landing page: It's a combination of Delivery and a way to say to customers that our service is here for Delivery everything you want.

The navigation bar is so important on the landing page because it's the way to make the website manageable for regular users, a Flex-box module and bootstrap are used to create a nice organized navigation bar, this navigation bar contains the brand on the left side, options of the website in the center and the drop-down button of log-ins option on the right side.

the way of choosing this method and not have only brand separate from other navigation bar element is that a good study is made for the most maintainable navigation method in 2018...2021 this made many websites update their navigation bar to focus on the same concept because users will by default focus in the center to see all option that a website can provide and at the same time if they will engage with the service they need a clear button separate from others to tell them the entry of user engagement.

why the brand name is set alone on the left side? it's the first element that users will see on the screen because in most cases website are made in English and English is left to right language, that's why it must be on left as the first separate element on the website this allows the user to recall the brand's name if necessary.

We used CSS to make a beautiful hover effect on buttons to help the users identifying their positions on the website.

3.3 Registration page

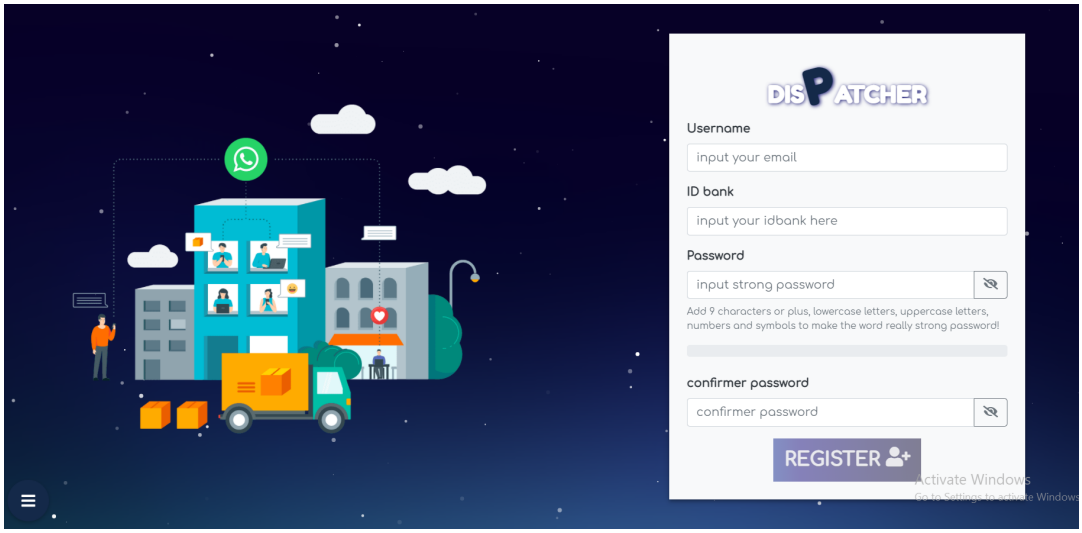


FIGURE 3.5: Registration Page

The registration page is containing only the important inputs fields that a customer need to fill out, this helps user to not be confused and engage with us in businesses, most users be afraid when they see a large amount of inputs and terms conditions, so a nice register form is the form with less inputs. so user can easily continue the process of fill-in all important information about his self later when he access his profile account.

3.3.1 Wholesalers Dashboard

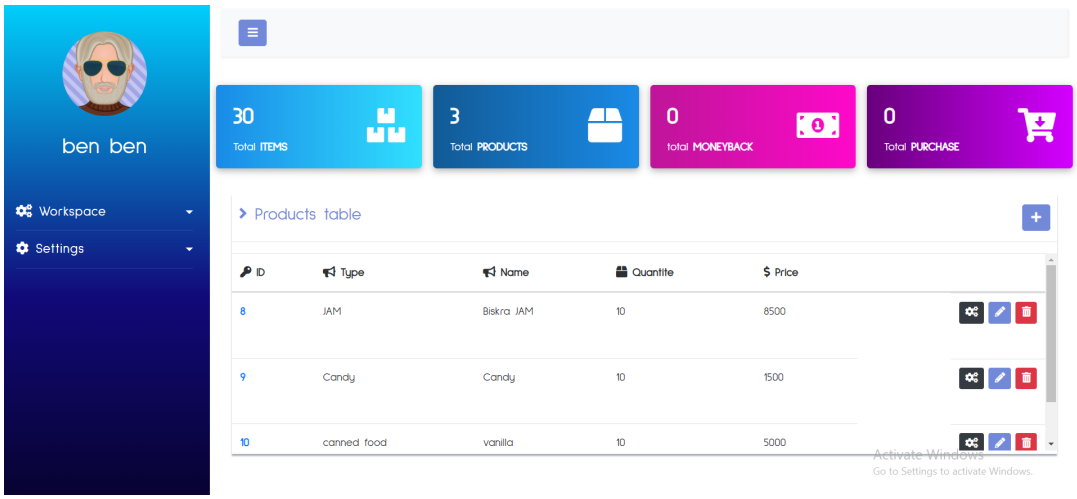


FIGURE 3.6: Wholesalers dashboard

A dashboard is a type of graphical user interface which often provides at-a-glance views of key performance indicators (KPIs) relevant to a particular objective or business process. In other usage, **dashboard** is another name for **progress report** or **report** and considered a form of data visualization.

FIGURE 3.7: Add-modify items

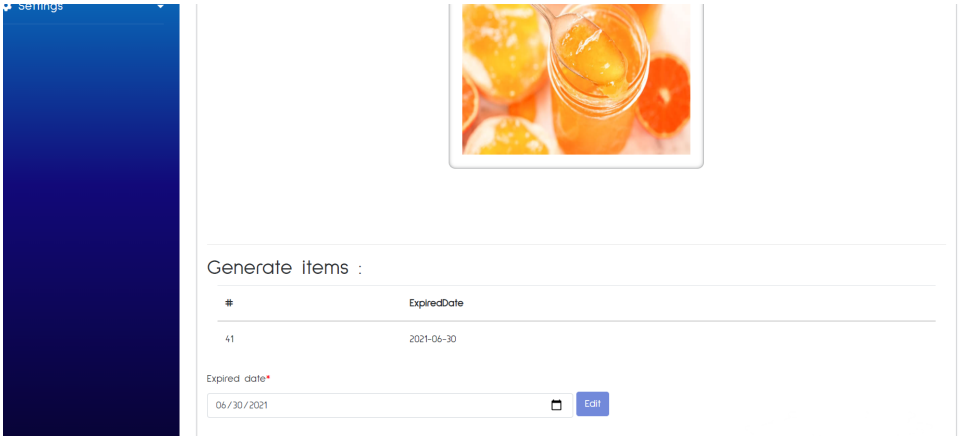


FIGURE 3.8: Add-modify items 2

Here at the client dashboard, the client can check his history of purchases, check the wholesalers he deals with and track his deliveries. he can also modify his orders update his orders or cancel them totally.

3.3.2 Form Validation

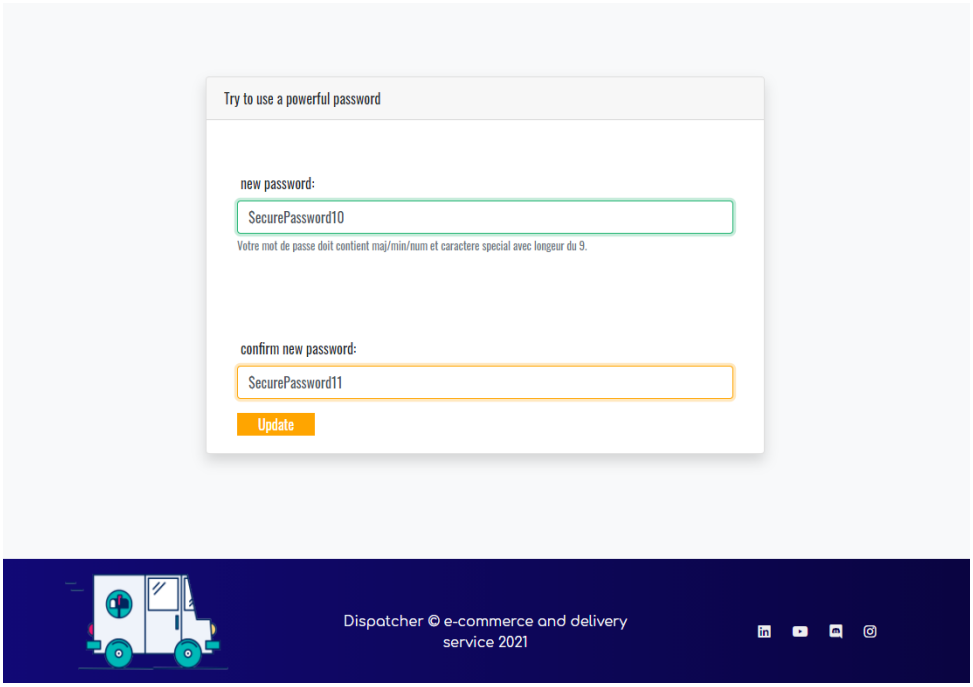


FIGURE 3.9: Form Validation 1

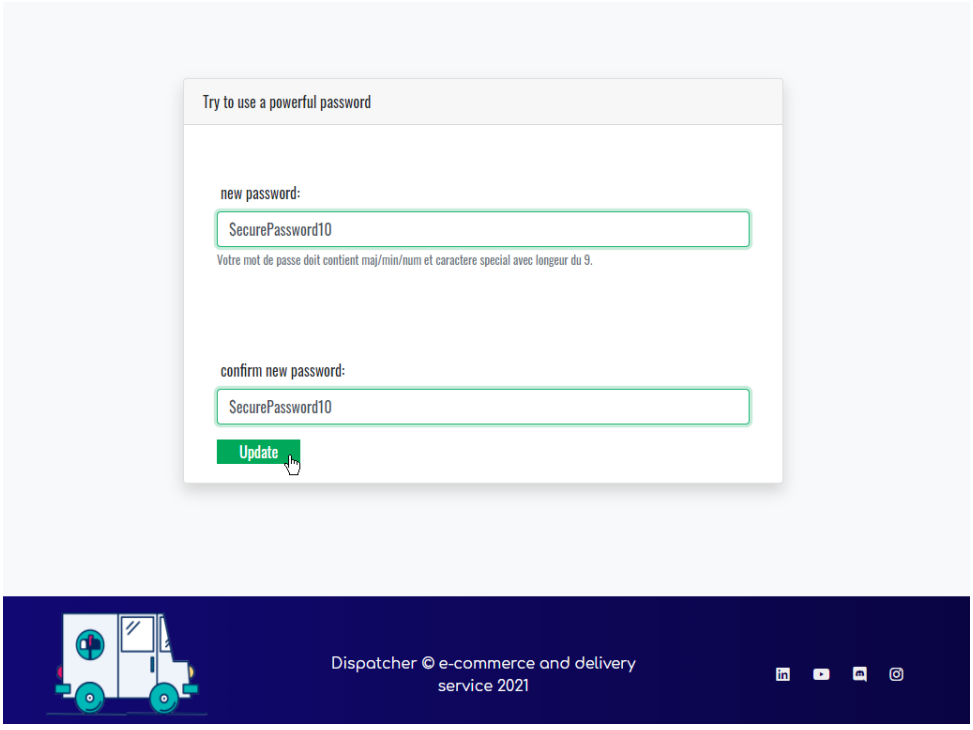


FIGURE 3.10: Form Validation 2

As seen on **Form Validation 2** figure. One of the most significant features of HTML5 form controls is the ability to validate most user data without relying on JavaScript. This is done by using validation attributes on form elements. for example **required** specifies whether a form field needs to be filled in before the form can be submitted. **minlength** and **maxlength** specifies the minimum and maximum length of textual data (strings). **min** and **max** specifies the minimum and maximum values of numerical input types, **type** specifies whether the data needs to be a number, an email address, or some other specific preset type. **pattern** specifies a regular expression that defines a pattern the entered data needs to follow. When an element is valid, the following things are true, The element matches the **:valid** CSS pseudo-class, which lets you apply a specific style to valid, The element matches the **:invalid** CSS pseudo-class, and sometimes other UI pseudo-classes (e.g., **:out-of-range**) depending on the error, which lets you apply a specific style to invalid elements.

We also used Validating forms with JavaScript as most browsers support the Constraint Validation API, which consists of a set of methods and properties available on the following form element DOM interfaces.

3.4 Editing Client/Wholesalers Info

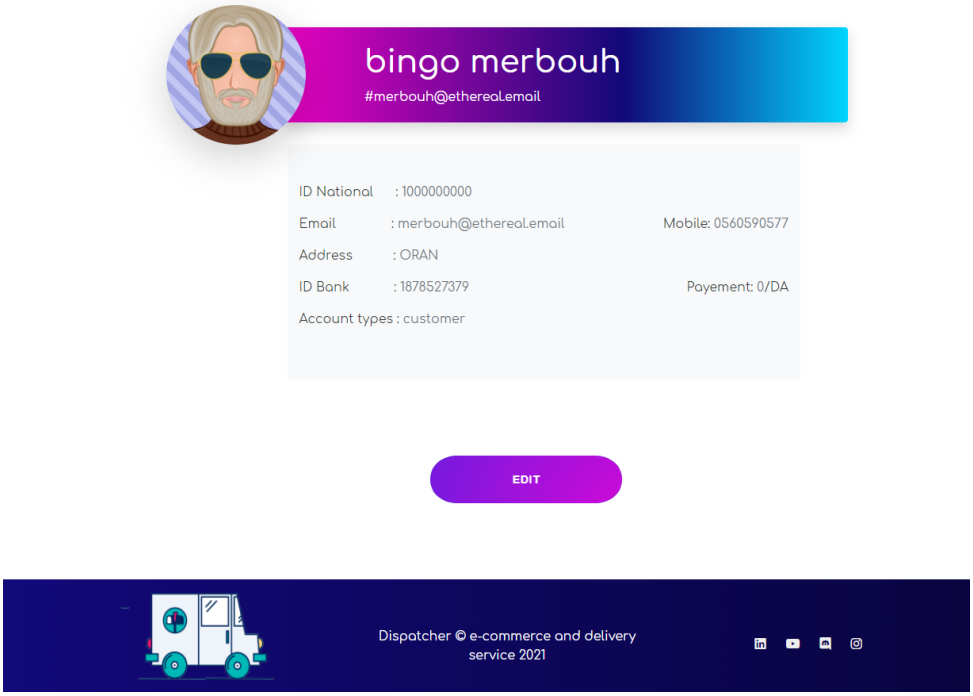


FIGURE 3.11: Edit profile info 1

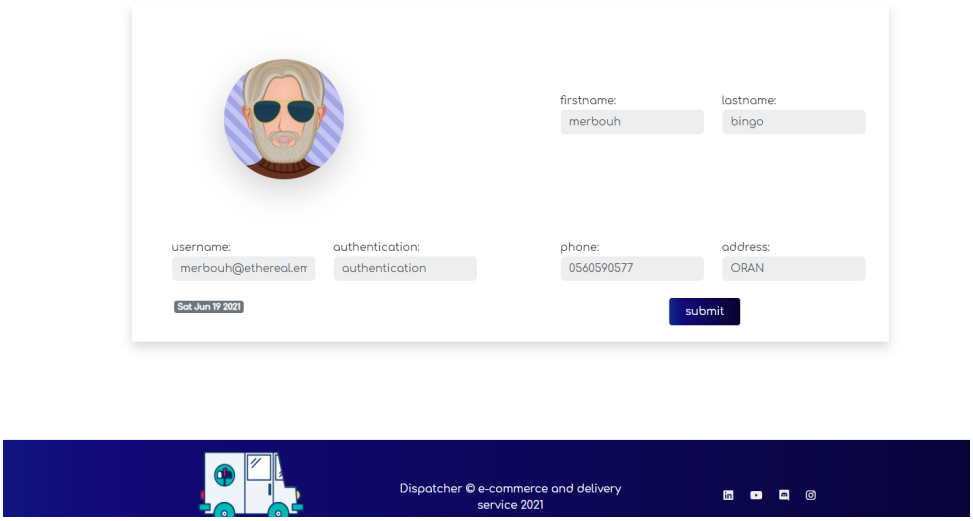


FIGURE 3.12: Edit profile info 2

Here the user either the customer or the wholesaler or the delivery team can modify their personal information.

3.4.1 Client Dashboard

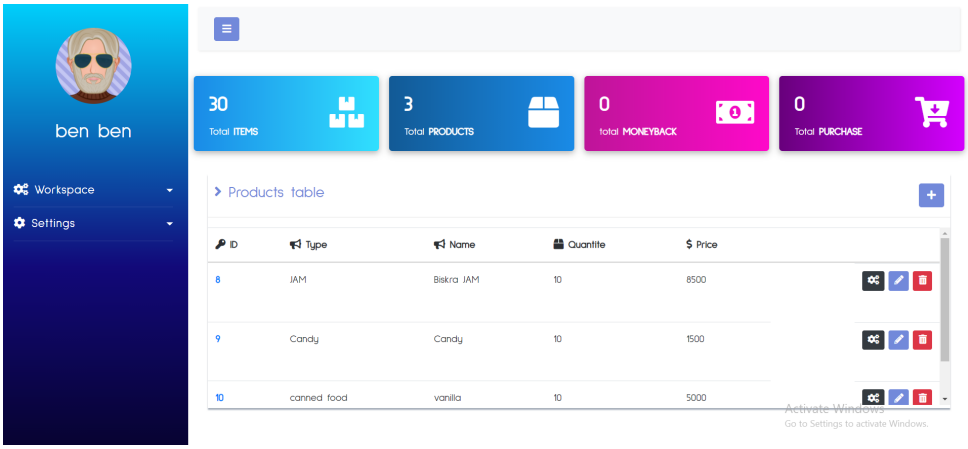


FIGURE 3.13: Client main dashboard

Here we see the most important part which is the dashboard in we see the client's general dashboard where he can see the important stuff like his balance, his orders, his history, and even contact the people that he makes business with like wholesalers and delivery team.

3.4.2 Notifications

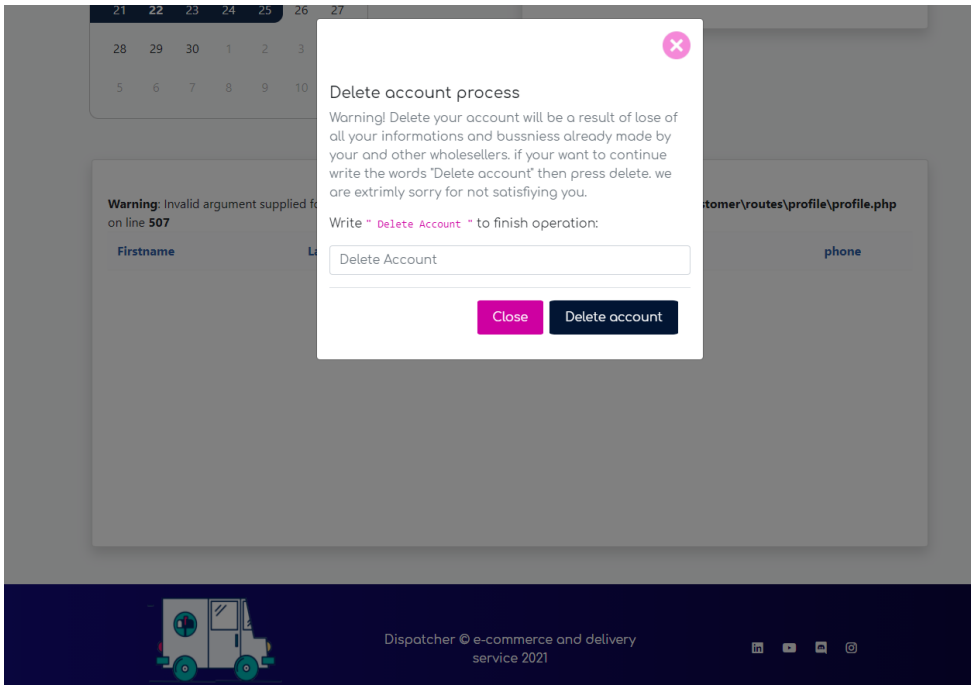


FIGURE 3.14: Notification 1

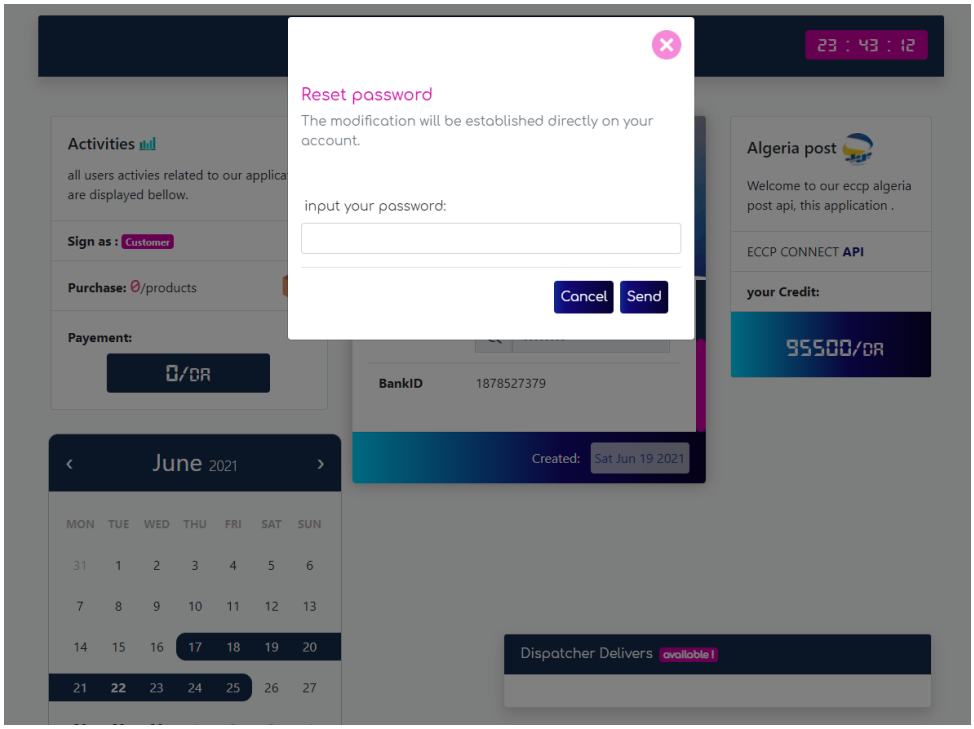


FIGURE 3.15: Notification 2

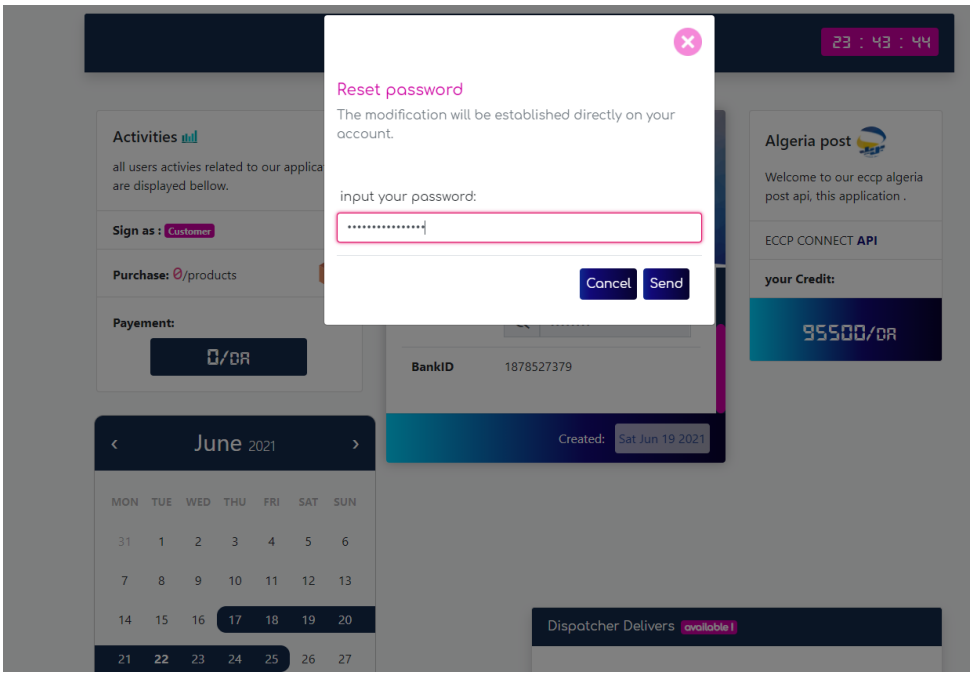


FIGURE 3.16: Notification 3

These are samples of how notifications in our web app are displayed to the user, we can notice how colors help make it easier to the user to understand what are the problems and guide him to the right decision.

Chapter 4

Back End Development

Back-end is the server-side of the website. It stores and arranges data, and also makes sure everything on the client-side of the website works fine. It is the part of the website that you cannot see and interact with. It is the portion of software that does not come in direct contact with the users. The parts and characteristics developed by back-end designers are indirectly accessed by users through a front-end application. Activities, like writing API's, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the back-end.[31]

4.1 Database (relational databases)

A relational database organizes data into tables which can be linked or related based on data common to each. This capability enables you to retrieve an entirely new table from data in one or more tables with a single query. It also allows you and your business to better understand the relationships among all available data and gain new insights for making better decisions or identifying new opportunities. For example, imagine your company maintains a customer table that contains company data about each customer account and one or more transaction tables that contain data describing individual transactions. The columns (or fields) for the customer table might be Customer ID, Company Name, Company Address, etc.; the columns for a transaction table might be Transaction Date, Customer ID, Transaction Amount, Payment Method, etc. The tables can be related based on the common Customer ID field. You can, therefore, query the table to produce valuable reports, such as a consolidated customer statement. Report generators take these queries and run them on demand to create formal reports. Many of the documents businesses run to track inventory, sales, finance, or even perform financial projections come from a relational database operating behind the scenes.[15]

You can communicate with relational databases using Structured Query Language (SQL), the standard language for interacting with management systems. SQL allows the joining of tables using a few lines of code, with a structure most nontechnical employees can learn quickly. With SQL, analysts do not need to know where the order table resides on disk, how to perform the lookup to find a specific order, or how to connect the order and customer tables. The database compiles the query and figures out the correct data points.[15]

The primary benefit of the relational database approach is the ability to create meaningful information by joining the tables. Joining tables allows you to understand the relationships between the data, or how the tables connect. SQL includes

the ability to count, add, group, and also combine queries. SQL can perform basic math and subtotal functions and logical transformations. Analysts can order the results by date, name, or any column. Those features make the relational approach the single most popular query tool in business today.[15]

Relational databases have several advantages compared to other database formats:

Flexibility

SQL has its a built-in language for creating tables called Data Definition Language (DDL). DDL allows you to add new columns, add new tables, rename relations, and make other changes even while the database is running and while queries are happening. This allows you to change the schema or how you model data on the fly.[15]

Reduced redundancy

Relational databases eliminate data redundancy. The information for a single customer appears in one place a single entry in the customer table. The order table only needs to store a link to the customer table. The practice of separating the data to avoid redundancy is called normalization. Progressional database designers make sure the tables normalize during the design process.[15]

Ease of backup and disaster recovery

Relational databases are transactional they guarantee the state of the entire system is consistent at any moment. Most relational databases offer easy export and import options, making backup and restore trivial. These exports can happen even while the database is running, making restore on failure easy. Modern, cloud-based relational databases can do continuous mirroring, making the loss of data on restore measured in seconds or less. Most cloud-managed services allow you to create Read Replicas, like in IBM Cloud Databases for PostgreSQL. These Read Replicas enable you to store a read-only copy of your data in a cloud data center. Replicas can be promoted to Read/Write instances for disaster recovery as well.[15]

4.2 Client/Server Systems

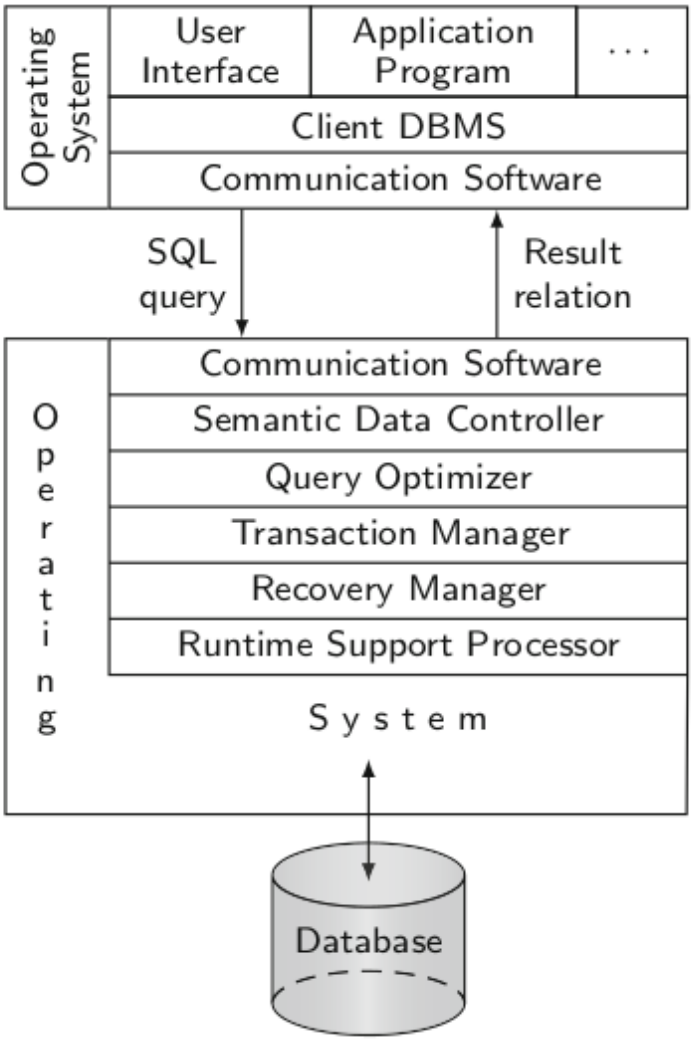


FIGURE 4.1: Client/server reference architecture

Client/server entered the computing scene at the beginning of 1990s and has made a significant impact on the DBMS technology. The general idea is very simple and elegant: distinguish the functionality that needs to be provided on a server machine from those that need to be provided on a client. This provides a two-level architecture which makes it easier to manage the complexity of modern DBMS's and the complexity of distribution.[38]

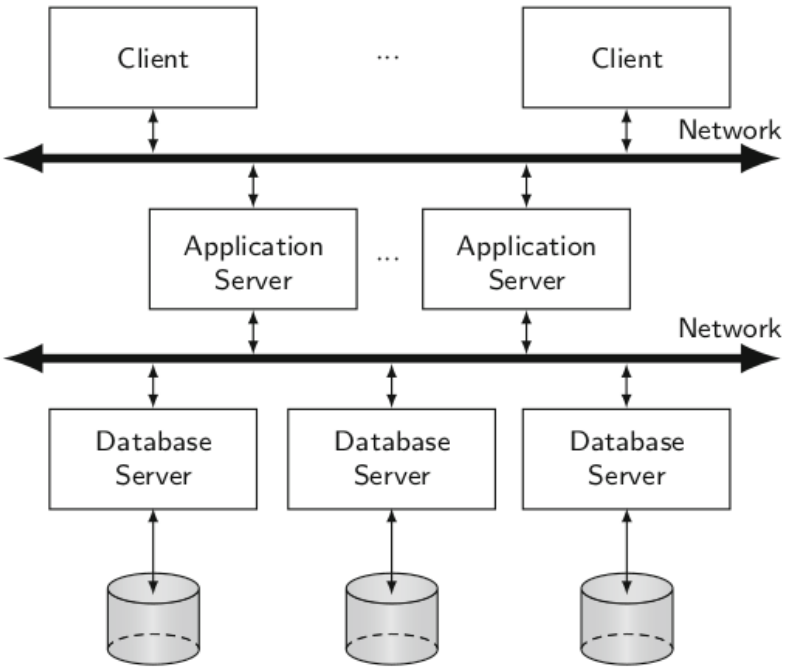


FIGURE 4.2: Distributed database servers

In relational client/server DBMS's, the server does most of the data management work. This means that all of query processing and optimization, transaction management, and storage management are done at the server. The client, in addition to the application and the user interface, has a DBMS client module that is responsible for managing the data that is cached to the client and sometimes managing the transaction locks that may have been cached as well. It is also possible to place consistency checking of user queries at the client side, but this is not common since it requires the replication of the system catalog at the client machines. This architecture, dep, is quite common in relational systems where the communication between the clients and the server(s) is at the level of SQL statements. In other words, the client passes SQL queries to the server without trying to understand or optimize them. The server does most of the work and returns the result relation to the client.[38]

There are a number of different realizations of the client/server architecture. The simplest is the case where there is only one server which is accessed by multiple clients. We call this multiple client/single server. From a data management perspective, this is not much different from centralized databases since the database is stored on only one machine the server that also hosts the software to manage it. However, there are important differences from centralized systems in the way transactions are executed and caches are managed since data is cached at the client, it is necessary to deploy cache coherence protocols. A more sophisticated client/server architecture is one where there are multiple servers in the system the so-called multiple client/multiple server approach. In this case, two alternative management strategies are possible: either each client manages its own connection to the appropriate server or each client knows of only its home server which then communicates with other servers as required. The former approach simplifies server code, but loads the client machines with additional responsibilities. This leads to what has been called heavy client systems. The latter approach, on the other hand, concentrates the data management functionality at the servers. Thus, the transparency of data access is provided at the server interface, leading to light clients.[38]

4.3 Over View Of the database

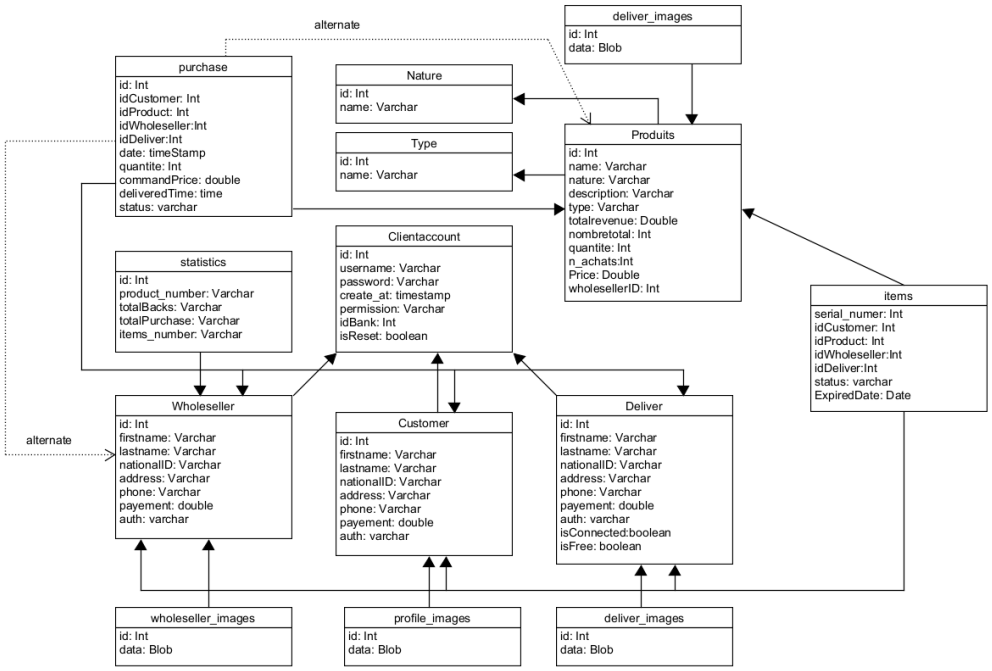


FIGURE 4.3: Dispatcher.ml database

We have designed the database with as simple as possible just to get a first draft working prototype version of our web application ready for testing, even with that in mind it took us a lot of effort to design a database with everything in our mind being applied. therefore the database shown in 4.3 figure. the working version we are using has even less tables as we didn't want to get over ourselves from the size of our web application.

While designing we used techniques we have learned during our study of Relational Databases and tried to use it to our favor. for example we have normalized our database to NF3 to make it dynamic and reducing the size of table when the application starts to grow with more customers and more data.

Due to the complexity of the relations between the entities of our application there is a very intensive use of foreign keys to maintain the flow of information between entities, this is also an indirect result if Normalizing our database.

4.4 Routing pages

Navigating through different sections of the page is a simple as a **href** HTML tag, we tried to keep it as simple as possible and use the simplest for of routing through links to other pages or other sections of the same page, Routing to back end services is also as simple as **GET/POST** type of requests with PHP and nothing fancy as debugging these are far more easier than using Express.js in NodeJS or other more complex types of routing. we know that as we keep developing and growing our web-app the time will come to move to more sophisticated method of routing to handle the growing complexity of the project. but for now we can settle for the simple methods to at least get a prototype going on for us.

4.5 Registration verification with Algerie Poste API

FIGURE 4.4: Algerie Poste API 2

Here we decided to use a free API service from Algerie Post which enables us to verify the CCP number of the client during the registration process, this will help us in two ways first it will ensure there will be no fake accounts in our database and second no duplicate accounts as each account can have a single unique CCP account linked to the client database.

4.6 Hosting

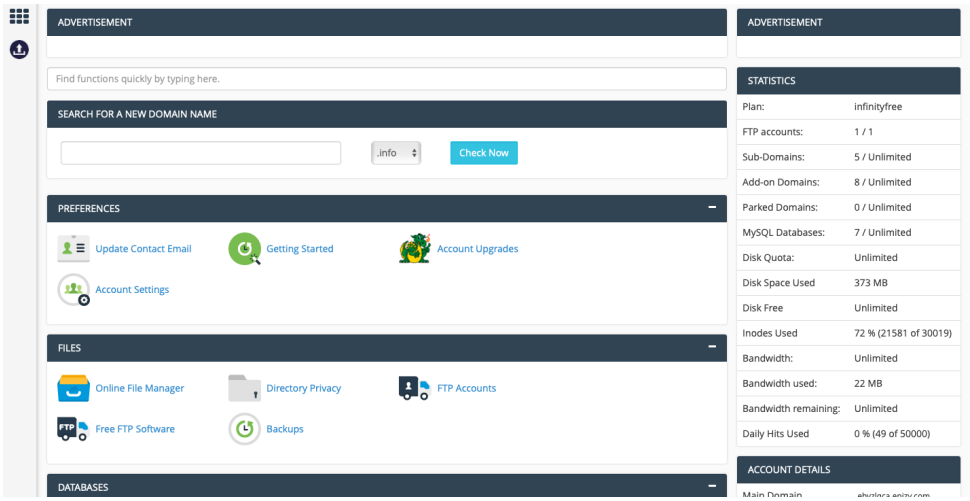


FIGURE 4.5: Infinityfree Hosting service

For web-hosting we opted for the free service of InfinityFree which is more than enough for demonstrating our PFE project and for general prototyping, it offers free space of 5GB of data. on a 24/7 Apache server running PHP 7.4 and MySQL, we can use any number we need of databases and the size of the database is unlimited. another benefit as we will see in next section **Domain Name** we can use our own domain names and not the ones provided by InfinityFree.

4.7 Domain Name

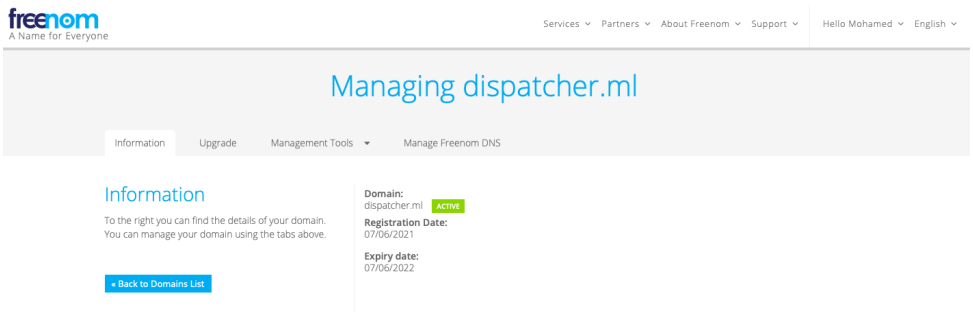


FIGURE 4.6: Domain name registration

Here we are using the free domain names services of freenom.com which offers free domain names with alias of (.ml , .ga , .tk) and many others. we opted for free one year service and reserved the following domain for our PFE project **dispatcher.ml**.

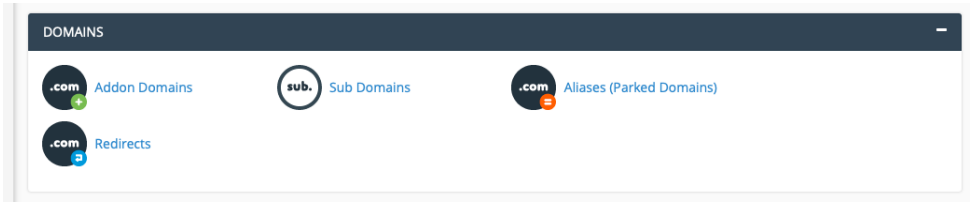


FIGURE 4.7: Domain name linking 1

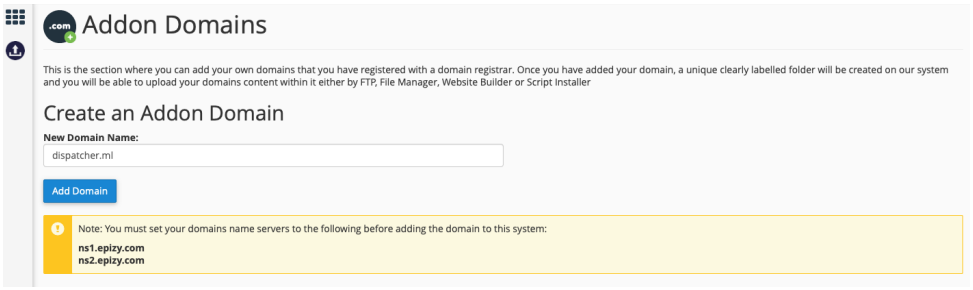


FIGURE 4.8: Domain name linking 2

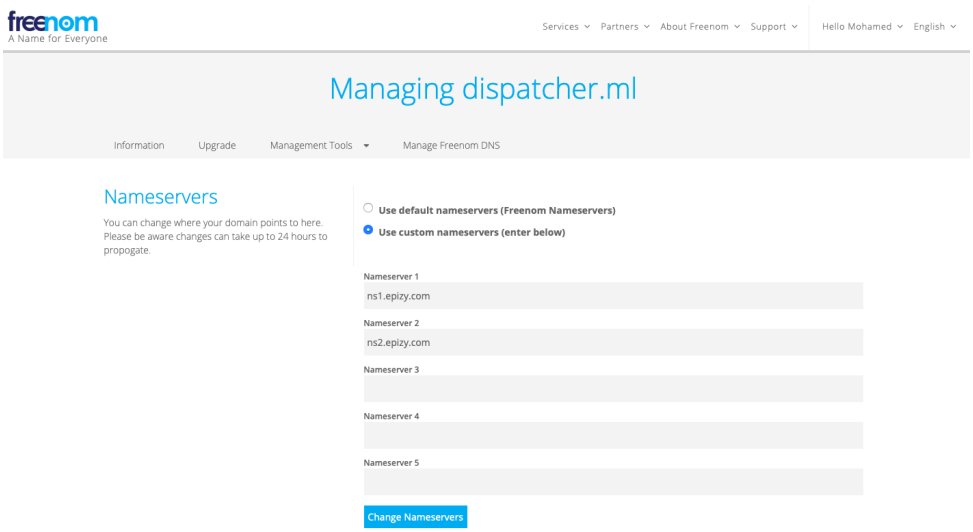


FIGURE 4.9: Domain name linking 3

In order to attach our domain name all we have to do is add the domain name to our hosting service and link it to our website, the next step is to update the DNS nameservers to link it to the hosting website.

4.8 Unauthorized Access (ERROR 403)

Web server security is important for any organization that has a physical or virtual Web server connected to the Internet. It requires a layered defense and is especially important for organizations with customer-facing websites.

We are using the Apache HTTP Server which has a good record for security and a developer community highly concerned about security issues. But it is inevitable that some problems small or large will be discovered in software after it is released. For this reason, it is crucial to take mesure to our hand by doing simple things like **Restricting access to web server directories** , **Restricting access to visitors from Algeria only**. we will also be taking performance into account as we will be **Restricting access media files** to prevent unauthorised download of media files from our server.

4.8.1 Restricting access to web server directories

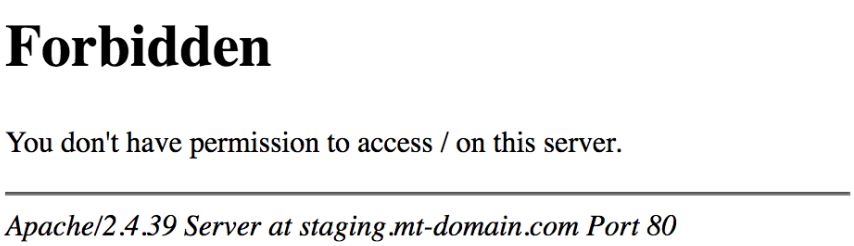


FIGURE 4.10: Page 403 ERROR

One of the most important things is to secure access to our web server and allow visitors to only gain access to the pages they can visit. there many ways to protect

access to the website's dirictories and even some file types, the best and easiest way is to use .htaccess file to give our Apache servers some commands. these are very similar to linux access restriction rules.

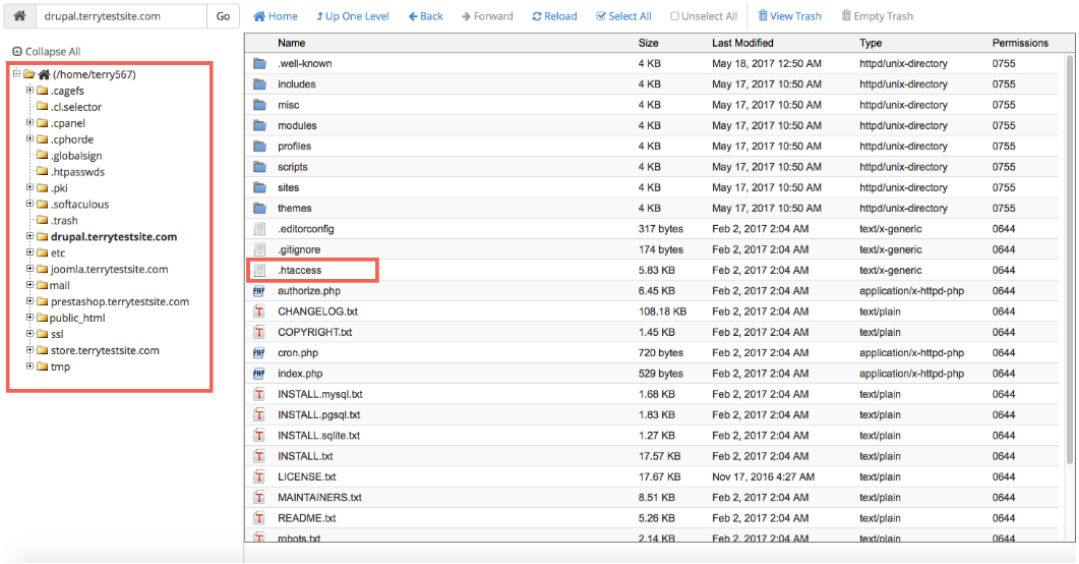


FIGURE 4.11: Locating .htaccess file

Steps to activate it Apache server

1. **Access to .htaccess File** We connect to our website with Filezilla and search for .htaccess usually it is in the root folder.

2. **Backup of the .htaccess File** Now we make a copy of the file before modifying it and keep it in a safe place.

3. **Deleting Caches** The last thing to do is to delete the caches on the Apache server to force the new rule.

4. **Modifing .htaccess File** It is important to use a simple text editor, we used Geany for editing.

5. **Stopping Indexing** now we add at the end of the file this line "Options -Indexes" and save it.

4.8.2 Restricting access to visitors from Algeria only

Since our web application is targeting algerian customers only, it would be wise to reduce trafic to our server to algerians only, this will reduce overhead on the server and save us from unwanted trafic, this is doable inside Apache web server by also modifying the .htaccess file.

Finding the IP range

A simple google search will yeild a list of IP addresses that we can work with. to be save we will take the entire range in from 41.200.0.0 to 41.201.255.255 , this way we make sure we don't exclude anyone.

Algeria IP Address Ranges

Begin IP Address	End IP Address	Total Count
41.200.0.0	41.201.255.255	131,072
41.220.144.0	41.220.159.255	4,096
41.221.16.0	41.221.31.255	4,096
41.223.236.0	41.223.239.255	1,024

FIGURE 4.12: IP range of Algeria

Now all we have to do is modify our .htaccess and add these 2 lines "**Deny from all**" this will block everyone now we add this line to leave access to algerian clients "**allow from 41.200.0.0 to 41.201.255.255**". and that's all we have to do, now anyone outside that IP range will get the 403 ERROR page.

4.8.3 Restricting access media files

```
# Simple File List Access Restricter

RewriteEngine On

# 1) If NOT the current host
RewriteCond %{HTTP_HOST}@@%{HTTP_REFERER} !^([^@]*)@@https?://\1/.*

# 2) Deny access to these types
RewriteRule \.
(gif|jpg|jpeg|png|tif|pdf|wav|wmv|wma|avi|mov|mp4|m4v|mp3|zip?)$ - [F]
```

FIGURE 4.13: Disabling media access

Another very useful thing is to disable access to media files as allowing access to them may result in lowering bandwidth to our server by allowing clients to download media files inside our servers directly.

To do this we can also modify .htaccess file and add the **RewriteRule** rule with media types we want to restrict.

4.9 SSL/TLS Encryption

Standard HTTP sends unencrypted data over the Internet making it easy to intercept. In contrast, Hypertext Transfer Protocol Secure (HTTPS) encryption prevents wiretapping, stolen credit card numbers, and other interceptions. HTTPS secures Internet traffic through encryption. HTTPS is a combination of the standard HTTP protocol and a security protocol called SSL/TLS.**??**

4.9.1 What is SSL/TLS encryption?

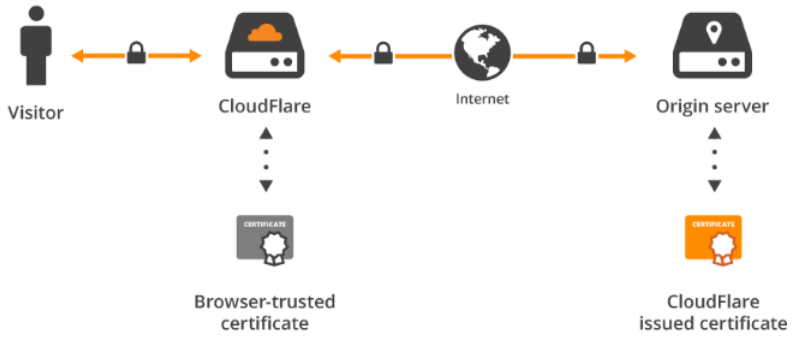


FIGURE 4.14: How SSL/TLS works

Transport Layer Security (TLS) is a protocol for encrypting data that is sent over the Internet. TLS grew out of Secure Sockets Layer (SSL), the first widely-adopted web encryption protocol, in order to fix most of the earlier protocol's security flaws. The industry still uses the terms somewhat interchangeably for historical reasons. Any web site that you visit starting with **https://** rather than **http://** is using TLS/SSL for communication between a browser and a server.**??**

Proper encryption practices are a necessity in order to prevent attackers from accessing important data. Because the Internet is designed in such a way that data is transferred across many locations, it is possible to intercept packets of important information as they move across the globe. Through the utilization of a cryptography protocol, only the intended recipient is able to decode and read the information and intermediaries are prevented from decoding the contents of the transferred data.**??**

The TLS protocol is designed to provide 3 components:

1. **Authentication** The ability to verify the validity of the provided identifications.
2. **Encryption** The ability to obfuscate information sent from one host to another.
3. **Integrity** The ability to detect forgery and tampering.

4.9.2 End-to-end HTTPS with Cloudflare

FIGURE 4.15: Changing Namerservers

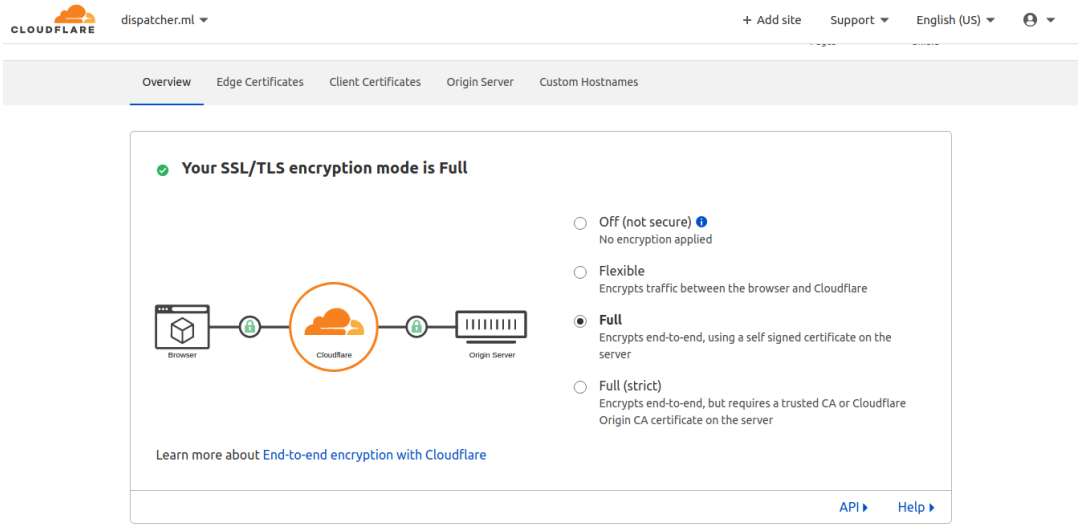


FIGURE 4.16: Activating SSL/TLS on Cloadflare

Using this service was very easy as all we had to do is register to cloadflare with our domain-name distpatcher.ml, then we had to change DNS records from our hosting DNS nameservers to the ones provided to us by cloudflare, this will make requests to our domain-name get handled by cloudflare's servers instead. then we had to enable the SSL/TLS certifications option.

4.10 Gzip Compression

Compression is a simple solution and effective way to save bandwidth and speed up our website.

——————————— SUBSECTION 1 ———————————-

4.10.1 Web application size

Well, the system works, but it's not that efficient. 100KB is a lot of text, and frankly, HTML is redundant. Every <html>, <table> and tag has a closing tag that's almost the same. Words are repeated throughout the document. Any way you slice it, HTML (and its beefy cousin, XML) is not lean.

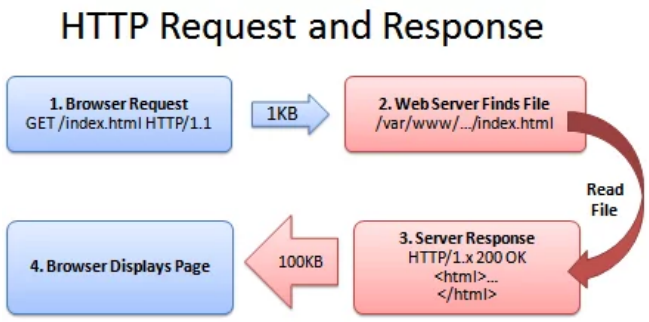


FIGURE 4.17: Uncompressed landing page

If we could send a .zip file to the browser (index.php.zip) instead of plain old index.php, we'd save on bandwidth and download time. The browser could download the zipped file, extract it, and then show it to user, who's in a good mood

because the page loaded quickly. The browser-server conversation might look like this.

Another thing that slows pages are photos and media in general as they actually consume the most space and even thought we opimize them by using SVG's on logos and webp on pictures. zipping them reduces thier size even further.

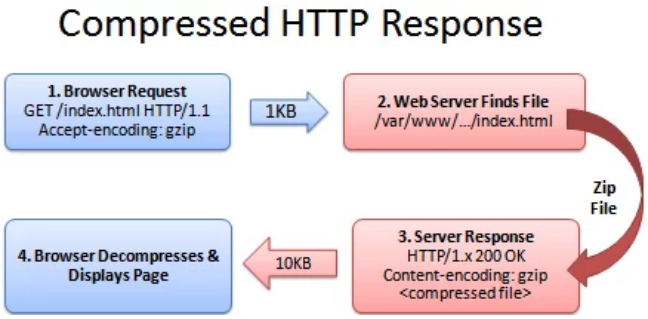


FIGURE 4.18: Compressed Landing page

4.10.2 Preparing backend in PHP

We add this line of code to our index.php to indicate to the Apache web server to gzip our pages in case .htaccess fails for some reason.

```
<?php if (substr_count($_SERVER['HTTP_ACCEPT_ENCODING'], 'gzip'))
ob_start("ob_gzhandler"); else ob_start(); ?>
```

FIGURE 4.19: PHP code for index.php

We check the "Accept-encoding" header and return a gzipped version of the file (otherwise the regular version). This is almost like building your own web-server (what fun!). But really, try to use Apache to compress your output if you can help it. You don't want to monkey with your files.

4.10.3 Setting up the server

The Apache server has this feature built in inside it. so the only thing to do is to set it properly.

For setting up the server to compress the website before sending it to web browsers, we write these scripts inside the .htaccess file inside Apache.

```
# compress text, html, javascript, css, xml:
AddOutputFilterByType DEFLATE text/plain
AddOutputFilterByType DEFLATE text/html
AddOutputFilterByType DEFLATE text/xml
AddOutputFilterByType DEFLATE text/css
AddOutputFilterByType DEFLATE application/xml
AddOutputFilterByType DEFLATE application/xhtml+xml
AddOutputFilterByType DEFLATE application/rss+xml
AddOutputFilterByType DEFLATE application/javascript
AddOutputFilterByType DEFLATE application/x-javascript

# Or, compress certain file types by extension:
<files *.html>
SetOutputFilter DEFLATE
</files>
```

FIGURE 4.20: Code inside htaccess file

Apache actually has two compression options:

1. **mod deflate :** is easier to set up and is standard.
2. **mod gzip :** more powerful as you can pre compress content.

Deflate is quick and works, so I use it; use mod gzip if that floats your boat. In either case, Apache checks if the browser sent the "Accept encoding" header and returns the compressed or regular version of the file. However, some older browsers may have trouble and there are special directives you can add to correct this.

4.10.4 Verifying Improvements

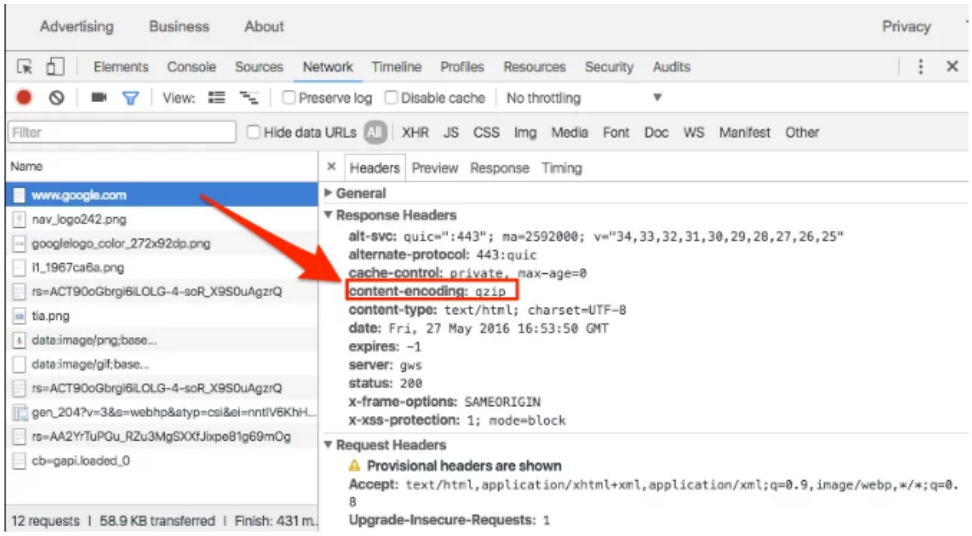


FIGURE 4.21: Network Tab in Chrome

it is possible to verify the size using the web browser for example in Chrome, we open the Developer Tools then Network Tab.

Now we need to verify the results by checking in our browser, after checking our landing page before using gzip web compression and after it we see a huge 72% reduction in size from 790KB to 221KB which improved loading times dramatically.

4.11 Code Optimization

To optimize our source code, we did to simple tricks one is quite long but very effective and the second is very simple and also very effective.

First we went through our code and did proof reading on it and removed any unnecessary code or any redundant code. we used some PHP OOP features in some parts of our code to simplify the coding process and reduce the number of lines. then we tried to separate redundant code in separate .php files and only include them late when needed.

Second after finishing with the proof reading part we went to using the minifying feature in visual studio code to convert our code to a single line of code which will dramatically improve loading times of our pages since it will be faster for web browser engines to read and compile.

SECTION 11 ——————————————————————————————-

4.12 SQL Injection Protection

SQL injection is one of the most common web attack mechanisms utilized by attackers to steal sensitive data from organizations. While SQL Injection can affect any data-driven application that uses a SQL database, it is most often used to attack web sites. SQL Injection is a code injection technique that hackers can use to insert malicious SQL statements into input fields for execution by the underlying SQL database. This technique is made possible because of improper coding of vulnerable web applications. These flaws arise because entry fields made available for user input unexpectedly allow SQL statements to go through and query the database directly.[19]

We wanted to provide the minimum of security features available, and one of the most common attacks are SQL-Injection attacks, so we took some measures to protect our database from these types of attacks.

4.12.1 prepared statements and parameterized queries

1. Using PDO (for any supported database driver):

```
$stmt = $pdo->prepare('SELECT * FROM employees WHERE name = :name');
$stmt->execute(array('name' => $name));
foreach ($stmt as $row) {
    // do something with $row
}
```

1. Using MySQLi (for MySQL):

```
$stmt = $dbConnection->prepare('SELECT * FROM employees WHERE name = ?');
$stmt->bind_param('s', $name);
$stmt->execute();
$result = $stmt->get_result();
while ($row = $result->fetch_assoc()) {
    // do something with $row
}
```

FIGURE 4.22: prepared statements examples in PHP

Use prepared statements and parameterized queries. These are SQL statements that are sent to and parsed by the database server separately from any parameters. This way it is impossible for an attacker to inject malicious SQL.[33]

4.12.2 DBMS Privileges

To minimize the potential damage of a successful SQL injection attack, you should minimize the privileges assigned to every database account in your environment. Do not assign DBA or admin type access rights to your application accounts. We understand that this is easy, and everything just 'works' when you do it this way, but it is very dangerous. Start from the ground up to determine what access rights your application accounts require, rather than trying to figure out what access rights you need to take away. Make sure that accounts that only need read access are only granted read access to the tables they need access to. If an account only needs access to portions of a table, consider creating a view that limits access to that portion of the data and assigning the account access to the view instead, rather than the underlying table. Rarely, if ever, grant create or delete access to database accounts. If you adopt a policy where you use stored procedures everywhere, and don't allow application accounts to directly execute their own queries, then restrict those accounts to only be able to execute the stored procedures they need. Don't grant them any rights directly to the tables in the database. SQL injection is not the only threat to your database data. Attackers can simply change the parameter values from one of the legal values they are presented with, to a value that is unauthorized for them, but the application itself might be authorized to access. As such, minimizing the privileges granted to your application will reduce the likelihood of such unauthorized access attempts, even when an attacker is not trying to use SQL injection as part of their exploit. While you are at it, you should minimize the privileges of the operating system account that the DBMS runs under. Don't run your DBMS as root or system! Most DBMSs run out of the box with a very powerful system account. For example, MySQL runs as system on Windows by default! Change the DBMS's OS account to something more appropriate, with restricted privileges.[33]

Chapter 5

Results and Benchmarks

The goal is to make analyzing and testing complex software after each change practical, which allows developers and testers to diagnose and remove faults as soon as they occur, thus saving testing and development effort while increasing software reliability. We introduce new analysis and testing techniques that improve the state of the art in the validation of evolving software by combining static and dynamic analysis. A special emphasis is put on compatibility of different versions of software in a network, as upgrades are usually done gradually, and hence nodes with different versions need to coexist and provide the required functionality. The current practices of software project teams for handling software changes and upgrades are unsatisfactory: each change usually requires an expensive revalidation of the whole system or is simply not checked thoroughly, thus potentially introducing new errors into the system design. When we raised this concern in an informal discussion with representatives of a major European systems vendor, we got the following answer: "Normally [we] understand the impact of changes by difference between code releases or simply based on the experience of the engineer managing the code". The root cause of this issue is that state-of-the-art testing and validation tools are not optimized to validate system changes and upgrades, but instead focus on a single program version only.[7]

5.1 Verification and Validation

Software validation is always linked with verification and it is most of the time labeled as "**Verification and validation (V & V)**", Verification is static, low level and hap- pens before and during the development stage and it consists of human based (developers) checking of documents and files, It has no code execution at this stage and uses walkthroughs, inspections and reviews, during our PFE we used verification for example to verify our HTML and CSS that our code has no bugs and it matches the standards written by w3.org. these 2 official tools were used for HTML http://validator.w3.org and for CSS http://jigsaw.w3.org/css-validator/.

[7]

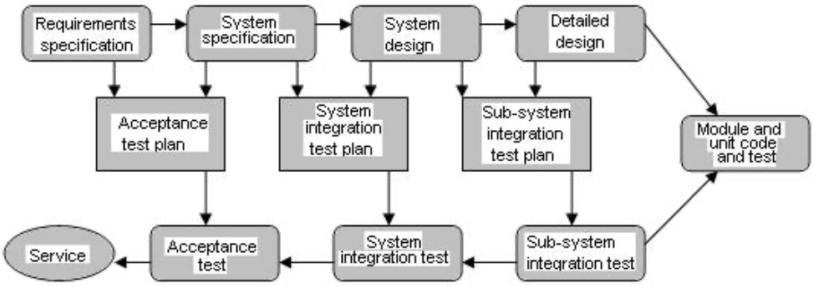


FIGURE 5.1: Software validation

Now to the Validating process which is dynamic and can involve humans (Testing teams) from inside or outside the development team and even customers /clients dur- ing last validation stages to insure the software is meeting the client demands and requirements, however there is also **Design Validation**, to validate our design we used the mathematical Formal method and TLA+/TLC to write all the scenarios and see if there is any unwanted outcomes in our design. Next step is to validate the actual software were it consists of the following **Development or component testing**, **System testing** and **User Acceptance testing (UAT).**

The main parts of System testing are :

1. **Execution of the application, Code execution in real time** .
2. **Real inputs to simulate user data and get actual outputs** .
3. **Using a standard for testing like BENCHMARK method** .
4. **White, Gray and Black box testing** .

Every "BENCHMARK" is deferment as every application targets different clients and may be execution on deferment platform, First as our PFE application is a web-app so we have created a benchmark that suits a website that has to run on multiple web browsers (Chrome, Opera, Firefox, Safari....) and different platforms like Desktops, Tablets and mobile phones. Even different operating systems add to the challenge as the same browser like Chrome will act different on Windows, Linux and MAC, so the more variables that can play means that our benchmark must contain all these variables. Second the "BENCHMARK" method requires **performance data** from competition or similar web-application to **compare** or web-app to what is in the market. So here we had to collect a list of web-app's that provide similar services and actions to our own and run benchmarks on these web-app's as will as our own and compare the results. Third our benchmark also has UX (user experience) part, because numbers can only tell one side of the story, so at this stage we ask potential customers and customers of the competition to try our application and provide feedback about positive experiences as well as negative experiences. We also gave them a table to compare our system to others and rate their experience and what they would change.

The software validation process appears in our PFE report in two parts of the **IMRaD** structure, first in the Methodology part as we lay down the structure of development including validation process we used, and secondly in the Results part as we show the results from the benchmarks we did.

Chapter 6

Conclusion

6.1 Final thoughts

This project was very interesting and informative to work on, we have learned so much from it from all sorts of topics. The project is still requires more work on the web site and more testing, many features we wanted to implement wasn't implemented because of the short time we had in hand. however we are very happy of the outcome and the learning experience we gained from it.

Working with the project from scratch meant we had to deal with all sorts of troubles and problems during the conception and programming phases. rewriting the entire project from scratch multiple times just to get to this early alpha stage teaches us that those programs that we take for granted and use them everyday, took from talented developers very long hours of coding and brainstorming to fix bugs they see everyday (some really really weird bugs that can make you quit) .

6.2 Future updates

As mentioned above the project is still missing many features that we want to implement like implementing geo-localization with either google maps API or openstreet API. we want to rebuild the back end in NodeJS to implement some security features like reverse proxy, and client routing services.

Another thing is missing is a mobile app, to get easy access for the clients and implement better notifications and tracking system for shipments, and live update of sales and deliveries to the wholesalers.

A Desktop app built with Electron is also considered as future addition to provide to our wholesalers an offline option for their business to keep working in case of internet loss.

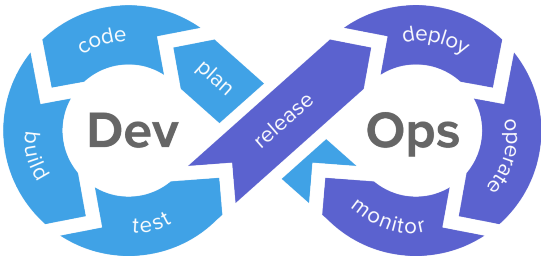
Appendix A

Technologies and Terminologies

A.1 Development Method

Software development tools are basically computer programs, and they usually run on personal computers, helping the programmer or system developer to create and/or modify or test applications programs.

A.1.1 DevOps



What is devops? Some might define it as a software development method, while others might think that it is a set of tools and technologies such as configuration management and continuous delivery. We would argue instead that devops is a cultural movement that seeks to improve both software development and the professional lives of the people involved in the field. In order to fully understand what we mean when we're talking about devops, it is necessary to understand not only what the concept means and how it is used today, but also the history of how it came to be.[12]

Pre-1880, walking was the primary mode of transportation. Cities were compact with residences and workplaces intermingled. Streets were narrow and inconsistently paved. As automobiles were introduced, cities made decisions about the organization and infrastructure either in favor of automobiles or existing pedestrians. These days, some cities forgo infrastructure planning that factors in pedestrians at all creating busy roadways with no safe way to travel by foot. Walking shaped the city and how we worked, and the arrival of new technology changed the landscape accordingly. Devops is part of the cultural weave that shapes how we work and why. While devops does involve certain tools and technologies, an equally important part of our culture is our values, norms, and knowledge. Examining how people work, the technologies that we use, how technology influeces how we work, and how people influence technology can help us make intentional decisions about the landscape of our industry.[12]

DevOps is a software development practice that brings people, process, and technology together to deliver continuous value. The approach is divided into planning and tracking, development, build and test, delivery, and monitoring and operations.

DevOps is unique in that development, IT operations, quality engineering, and security teams work together to create efficiency across all tasks involved in launching a new product, release, or update.[13]

Rooted in stability, consistency, and planning, the DevOps culture seeks to identify new ways to improve and streamline processes. As a result, DevOps focuses on maximizing efficiency, identifying programmable processes, and increasing automation.[13]

DevOps represents the intersections of development, operations, and quality assurance. Cross-disciplinary teams unite and collaborate in the development and delivery of software.[13]

A.2 Development Tools

Software development tools are basically computer programs, and they usually run on personal computers, helping the programmer or system developer to create and/or modify or test applications programs.

A.2.1 Visual Studio Code

Visual Studio Code

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages such as C++, C sharp, Java, Python, PHP, Go, and runtimes such as .NET and Unity. Begin your journey with VS Code with these introductory videos.[14]

A.2.2 Beekeeper Studio

Beekeeper Studio

Open Source SQL Editor and Database Manager.Beekeeper Studio is built with Vue.js.Beekeeper Studio is built and maintained by Matthew Rathbone and Gregory Garden. Broadly, Matthew does the backend platform stuff, and Gregory does the UI and UX stuff.[4]

A.2.3 StarUML

StarUML is a sophisticated software modeler aimed to support agile and concise modeling, Modeling a software system requires describing multiple models because it is not enough to describe the system with a single perspective, so we typically make multiple models such as Use-Case Model, Design Model, Component Model, Deployment Model, or others in a Project. Typically Project is organized as a set of UMLModels, UMLPackages, or UMLSubsystems. If you want to know more about UML Elements, please refer to OMG UML Specification.[23]

A.2.4 AMPPS

AMPPS is a WAMP MAMP Stack. It includes Apache, PHP, PERL, Python mod wsgi, MySQL, MongoDB, phpMyAdmin, RockMongo, FTP Server FileZilla Server Windows and Pure-FTPd Mac. AMPPS enables you to focus more on using applications rather than maintaining them. AMPPS has a Control Center which controls the Servers like Apache, MySQL, FTP, MongoDB and allows the users to edit the Configuration files of the respective servers and check the Log Files in case of an error.[2]

A.2.5 GitHub

GitHub is a cloud-based hosting service that provides a user-friendly UI experience to the users for git versioning. Users can also create unlimited repositories on the platform where they can store code Dles, images, and other essentials required to run the project[32]

In simple words, GitHub Actions enables the user to create custom Software Development Life Cycle (SDLC) workMows in their GitHub repositories. It gives a privilege to the repository owner to write individual actions and then combine them to create a custom workMow of their choice for their project. GitHub Actions also provides the facility to build end-to- end Continuous Integration (CI) and Continuous Deployment (CD) capabilities directly in the repository. However, the owner does not need to write the actions by themselves. GitHub has an inbuilt market place

where people can Dnd the actions created by other people, and reuse them if it Dts their needs. Since GitHub actions is enabled on all repositories and organizations by default, extra e:orts are not needed for getting started with it.[32]

A.2.6 Filezilla

FileZilla is a free and open-source, cross-platform FTP application, consisting of FileZilla Client and FileZilla Server. Clients are available for Windows, Linux, and macOS, servers are available for Windows only. Both server and client support FTP and FTPS (FTP over SSL/TLS), while the client can in addition connect to SFTP servers.[16]

FileZilla's source code is hosted on SourceForge and the project was featured as Project of the Month in November 2003. However, there have been criticisms that SourceForge bundles malicious software with the application.[16]

A.2.7 Geany

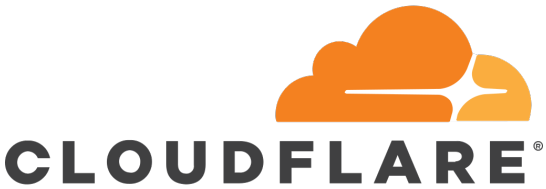
Geany is a lightweight GUI text editor using Scintilla and GTK, including basic IDE features. It is designed to have short load times, with limited dependency on separate packages or external libraries on Linux. It has been ported to a wide range of operating systems, such as BSD, Linux, macOS, Solaris and Windows. The Windows port lacks an embedded terminal window; also missing from the Windows version are the external development tools present under Unix, unless installed separately by the user. Among the supported programming languages and markup languages are C, C++, C#, Java, JavaScript, PHP, HTML, LaTeX, CSS, Python, Perl, Ruby, Pascal, Haskell, Erlang, Vala and many others.[17]

A.2.8 LaTeX

LaTeX, which is pronounced «Lah-tech» or «Lay-tech» (to rhyme with «blech» or «Bertolt Brecht»), is a document preparation system for high-quality typesetting. It is most often used for medium-to-large technical or scientific documents but it can be used for almost any form of publishing.[26]

It is the main tool we used during the creation of this project paper, it helped us alot with bibliography organization and documentation.

A.2.9 Cloudflare



Cloudflare is one of the world's largest networks. Today, businesses, non-profits, bloggers, and anyone with an Internet presence boast faster, more secure websites and apps thanks to Cloudflare. Approximately 25 million Internet properties are on Cloudflare, and our network is growing by tens of thousands each day. Cloudflare powers Internet requests for 17% of the Fortune 1,000 and serves 25 million HTTP requests per second on average. Cloudflare also provides security by protecting Internet properties from malicious activity like DDoS attacks, malicious bots, and other nefarious intrusions. [36]

A.3 Front End Side

A.3.1 HTML

HTML (the Hypertext Markup Language) and CSS (Cascading Style Sheets) are two of the core technologies for building Web pages. HTML provides the structure of the page, CSS the (visual and aural) layout, for a variety of devices. Along with graphics and scripting, HTML and CSS are the basis of building Web pages and Web Applications.[20]

HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behavior (JavaScript).[21]

"Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.[21]

HTML is the World Wide Web's core markup language. Originally, HTML was primarily designed as a language for semantically describing scientific documents. Its general design, however, has enabled it to be adapted, over the subsequent years, to describe a number of other types of documents and even applications.[22]

HTML, its supporting DOM APIs, as well as many of its supporting technologies, have been developed over a period of several decades by a wide array of people with different priorities who, in many cases, did not know of each other's existence.[22]

Features have thus arisen from many sources, and have not always been designed in especially consistent ways. Furthermore, because of the unique characteristics of the web, implementation bugs have often become de-facto, and now de-jure, standards, as content is often unintentionally written in ways that rely on them before they can be fixed.[22]

A.3.2 CSS

CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts. It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based markup language. The separation of HTML from CSS makes it easier to maintain sites, share style sheets across pages, and tailor pages to different environments. This is referred to as the separation of structure (or: content) from presentation.[20]

Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media.[11]

CSS is among the core languages of the open web and is standardized across Web browsers according to W3C specifications. Previously, development of various parts of CSS specification was done synchronously, which allowed versioning of the latest recommendations. You might have heard about CSS1, CSS2.1, CSS3. However, CSS4 has never become an official version.[11]

A.3.3 Java Script

JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js,

Apache CouchDB and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles. Read more about JavaScript.[24]

This section is dedicated to the JavaScript language itself, and not the parts that are specific to Web pages or other host environments. For information about API specifics to Web pages, please see Web APIs and DOM.[24]

The standards for JavaScript are the ECMAScript Language Specification (ECMA-262) and the ECMAScript Internationalization API specification (ECMA-402). The JavaScript documentation throughout MDN is based on the latest draft versions of ECMA-262 and ECMA-402. And in cases where some proposals for new ECMAScript features have already been implemented in browsers, documentation and examples in MDN articles may use some of those new features.[24]

A.3.4 Bootstrap

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components. As of April 2021, Bootstrap is the tenth most starred project on GitHub, with more than 150,000 stars, behind freeCodeCamp (almost 312,000 stars), Vue.js framework, React library, TensorFlow and others.[9]
Bootstrap was originally created by Twitter developers for internal use, but after a while, it was laid out in open access and became a convenient set of tools for developing user interfaces of any complexity. So, now Bootstrap is an open and free HTML, CSS and JS toolkit that is used by web developers to create responsive website designs quickly and effectively. You can find great examples of the bootstrap web design on specialized sites like Awwward. The Bootstrap framework is used not only by independent developers but also by entire companies. The main area of its application is the development of front-end component sites and admin interfaces. Among similar systems (Foundation, UIkit, Semantic UI, InK, etc.), the Bootstrap framework is the most popular.[35]

A.4 Back End Side

A.4.1 PHP

PHP (recursive acronym for PHP: Hypertext Preprocessor) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.[29]

PHP is mainly focused on server-side scripting, so you can do anything any other CGI program can do, such as collect form data, generate dynamic page content, or send and receive cookies. But PHP can do much more.[29]

PHP can be used on all major operating systems, including Linux, many Unix variants (including HP-UX, Solaris and OpenBSD), Microsoft Windows, macOS, RISC OS, and probably others. PHP also has support for most of the web servers today. This includes Apache, IIS, and many others. And this includes any web server that can utilize the FastCGI PHP binary, like lighttpd and nginx. PHP works as either a module, or as a CGI processor.[29]

So with PHP, you have the freedom of choosing an operating system and a web server. Furthermore, you also have the choice of using procedural programming or object oriented programming (OOP), or a mixture of them both.[29]

With PHP you are not limited to output HTML. PHP's abilities includes outputting images, PDF files and even Flash movies (using libswf and Ming) generated on the fly. You can also output easily any text, such as XHTML and any other XML file. PHP can autogenerate these files, and save them in the file system, instead of printing it out, forming a server-side cache for your dynamic content.[29]

One of the strongest and most significant features in PHP is its support for a wide range of databases. Writing a database-enabled web page is incredibly simple using one of the database specific extensions (e.g., for mysql), or using an abstraction layer like PDO, or connect to any database supporting the Open Database Connection standard via the ODBC extension. Other databases may utilize cURL or sockets, like CouchDB.[29]

PHP also has support for talking to other services using protocols such as LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (on Windows) and countless others. You can also open raw network sockets and interact using any other protocol. PHP has support for the WDDX complex data exchange between virtually all Web programming languages. Talking about interconnection, PHP has support for instantiation of Java objects and using them transparently as PHP objects.[29]

A.4.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications. f that is what you are looking for,

you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together. MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.[27]

A.4.3 Apache web server

The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards. The Apache HTTP Server ("httpd") was launched in 1995 and it has been the most popular web server on the Internet since April 1996. It has celebrated its 25th birthday as a project in February 2020. The Apache HTTP Server is a project of The Apache Software Foundation.[3]

A.4.4 Domain Names with freenom

Freenom is the world's first and only free domain provider. Our mission is to bring people online and help countries develop their digital economy. By breaking barriers and integrating free domains with the latest website building and hosting technologies, Freenom makes it easy for any business or individual to build websites and content, at no cost. Using the latest AnyCast Cloud technology, Freenom guarantees the stability and performance of all the domains it manages. And with strategic trusted security partners such as Twitter, Internet Identity, Kaspersky and more than 40 others, we can deliver state of the art anti-abuse technology to keep free domains safe and secured for all internet users.[25]

A.4.5 Web Hosting with Infinityfree

InfinityFree is an independent free hosting initiative, dedicated to providing reliable free hosting services.[18]

Appendix B

Data Base Tables

1 achats

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM
Last check: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No		auto_increment			
idClient	int(11)		No					
idProduct	int(11)		No					
date	varchar(500)		No					
quantite	int(11)		No					
prixCommand	double		No					

FIGURE B.1: Client account Table

2 clientaccount

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No		auto_increment			
username	varchar(50)		No					
password	varchar(500)		No					
created_at	varchar(50)		No					
permission	varchar(50)		No					
idBank	int(11)		No					
isReset	varchar(250)		No					

FIGURE B.2: Achets Table

3 customer

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No					
firstname	varchar(250)		No					
lastname	varchar(250)		No					
nationalID	varchar(250)		No					
address	varchar(250)		No					
phone	varchar(250)		No					
paiement	double		No					
auth	varchar(250)		No					

FIGURE B.3: Customer Table

4 deliver

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM
Last check: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No					
firstname	varchar(250)		No					
lastname	varchar(250)		No					
nationalID	varchar(250)		No					
address	varchar(250)		No					
phone	varchar(250)		No					
paiement	double		No					
auth	varchar(250)		No					

FIGURE B.4: Delivery Table

5 deliver_images

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Id	int(11)		No					
imageType	varchar(255)		No					
imageData	longblob		No					

FIGURE B.5: Delivery images Table

PDF export page

6 items

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
serial	int(11)		No		auto_increment			
customerID	int(11)		No					
DeliverID	int(11)		No					
WholeSellerID	int(11)		No					
ProductID	int(11)		No					
Status	varchar(250)		No					
ExpiredDate	varchar(250)		No					

FIGURE B.6: Items Table

7 nature

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No		auto_increment			
name	varchar(250)		No					

FIGURE B.7: Category Table

8 output_images

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Id	int(11)		No					
imageType	varchar(255)		No					
imageData	longblob		No					

FIGURE B.8: Output Images Table

9 produits

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No		auto_increment			
name	varchar(50)		No					
nature	varchar(200)		No					
description	varchar(200)		No					
type	varchar(50)		No					
totalreveue	double		No					
nombretotal	int(11)		No					
quantite	int(11)		No					
n_achats	int(11)		No					
price	double		No					
wholesellerID	int(11)		No					

FIGURE B.9: Products Table

10 profile_images

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Id	int(11)		No					
imageType	varchar(255)		No					
imageData	longblob		No					

FIGURE B.10: Profile Images Table

11 statistique

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No					
nombreProduit	int(11)		No					
revenueTotal	double		No					
total_purchase	int(11)		No					
products	int(11)		No					

FIGURE B.11: Statistics Table

12 types

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No		auto_increment			
name	varchar(250)		No					

FIGURE B.12: Types Table

13 wholeseller

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No					
firstname	varchar(250)		No					
lastname	varchar(250)		No					
nationalID	varchar(250)		No					
address	varchar(250)		No					
phone	varchar(250)		No					
paiement	double		No					
auth	varchar(250)		No					

FIGURE B.13: Wholesalers Table

14 wholeseller_images

Creation: Jun 24, 2021 at 01:27 PM
Last update: Jun 24, 2021 at 01:27 PM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
Id	int(11)		No					
imageType	varchar(255)		No					
imageData	longblob		No					

FIGURE B.14: Wholesalers Images Table

Bibliography

[1] Samisa Abeysinghe and Inc Ebrary. *PHP team development : easy and effective team work using MVC, agile development, source control, testing, bug tracking, and more*. Packt Publishing, 2009.

[2] *AMPPS Docs*. http://www.ampps.com/wiki/.

[3] *Apache HTTP Server Version 2.4 Documentation - Apache HTTP Server Version 2.4*. Apache.org, 2019. URL: https://httpd.apache.org/docs/2.4/.

[4] *Beekeeper Studio*. docs.beekeeperstudio.io. URL: https://docs.beekeeperstudio.io/ (visited on 06/08/2021).

[5] David Benyon. *Designing interactive systems. a comprehensive guide to HCI and interaction design*. Pearson, 2014.

[6] Jeff Bullas. *The Importance of Fonts in Web Design - Do You Have The Right One?* Jeffbullas's Blog, Aug. 2019. URL: https://www.jeffbullas.com/the-importance-of-fonts-in-web-design/.

[7] Hana Chockler and Natasha Sharygina. *Validation of evolving software*. Springer, 2015.

[8] Jim Conallen. *Building Web applications with UML*. Addison-Wesley, 2003.

[9] Wikipedia Contributors. *Bootstrap (front-end framework)*. Wikipedia, Apr. 2019. URL: https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework).

[10] Wikipedia Contributors. *Front-end web development*. Wikipedia, Dec. 2019. URL: https://en.wikipedia.org/wiki/Front-end_web_development (visited on 01/06/2020).

[11] *CSS: Cascading Style Sheets*. MDN Web Docs, June 2019. URL: https://developer.mozilla.org/en-US/docs/Web/CSS.

[12] Jennifer Davis and Ryn Daniels. *Effective devOps building a culture of collaboration, affinity, and tooling at scale*. O'reilly, 2016.

[13] *DevOps vs. Agile | Microsoft Azure*. Microsoft.com, 2019. URL: https://azure.microsoft.com/en-us/overview/devops-vs-agile/ (visited on 11/05/2019).

[14] *Documentation for Visual Studio Code*. code.visualstudio.com. URL: https://code.visualstudio.com/docs.

[15] IBM Cloud Education. *relational-databases*. Ibm.com, Aug. 2019. URL: https://www.ibm.com/cloud/learn/relational-databases.

[16] *FileZilla*. Wikipedia, Sept. 2020. URL: https://en.wikipedia.org/wiki/FileZilla.

[17] *Geany*. Wikipedia, Apr. 2021. URL: https://en.wikipedia.org/wiki/Geany (visited on 06/17/2021).

[18] *Getting started with a new web hosting account*. InfinityFree Knowledge Base. URL: https://support.infinityfree.net/getting-started/getting-started-with-a-new-account/ (visited on 06/08/2021).

[19] *How to protect against SQL injection attacks*. https://security.berkeley.edu/education-awareness/how-protect-against-sql-injection-attacks. Accessed: 2021-6-24.

[20] *HTML and CSS W3C*. W3.org, 2014. URL: https://www.w3.org/standards/webdesign/htmlcss.html.

[21] *HTML HyperText Markup Language*. MDN Web Docs, Dec. 2018. URL: https://developer.mozilla.org/en-US/docs/Web/HTML.

[22] *HTML Standard*. html.spec.whatwg.org. URL: https://html.spec.whatwg.org/multipage/introduction.html (visited on 06/08/2021).

[23] *Introduction*. StarUML documentation, 2020. URL: https://docs.staruml.io/.

[24] *JavaScript*. MDN Web Docs, July 2019. URL: https://developer.mozilla.org/en-US/docs/Web/javascript.

[25] *Knowledgebase Freenom*. my.freenom.com. URL: https://my.freenom.com/knowledgebase.php (visited on 06/08/2021).

[26] *LaTeX Documentation*. www.latex-project.org. URL: https://www.latex-project.org/help/documentation/ (visited on 06/08/2021).

[27] *MySQL 8.0 Reference Manual*. Mysql.com, 2019. URL: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html.

[28] Janis Osis. *Topological UML modeling : an improved approach for domain modeling and software development*. Elsevier, 2017.

[29] Php.net. *What is PHP Manual*. Php.net, 2019. URL: https://www.php.net/manual/en/intro-whatis.php.

[30] Yvonne Rogers, Jenny Preece, and Helen Sharp. *Interaction design*. Wiley ; Chichester, 2011.

[31] sabyasachi.samadder. *Frontend vs Backend - GeeksforGeeks*. GeeksforGeeks, July 2019. URL: https://www.geeksforgeeks.org/frontend-vs-backend/.

[32] Sanchit. *What Is GitHub Action and How Do They Help*. Medium, Oct. 2020. URL: https://medium.com/swlh/what-is-github-action-and-how-do-they-help-c8b254118fa5 (visited on 06/08/2021).

[33] *SQL injection prevention - OWASP cheat sheet series*. https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html. Accessed: 2021-6-24.

[34] Bhuvan Unhelkar. *Software engineering with UML*. Crc Press, 2018.

[35] *What is Bootstrap and How to Use it in Web Development?* F5 Studio. URL: https://f5-studio.com/articles/what-is-bootstrap-and-how-to-use-it-in-web-development/.

[36] "What is Cloudflare". In: *Cloudflare* (). URL: https://www.cloudflare.com/learning/what-is-cloudflare/.

[37] Matt Zandstra. *PHP 8 OBJECTS, PATTERNS, AND PRACTICE : mastering oo enhancements, design patterns, and essential... development tools.* Apress, 2021.

[38] M Tamer Zsu. *Principles Of Distributed Database Systems*. Springer, 2019.