

Table of Contents

CHAPTER 1	1
1.1 Background	1
1.2 Scope	2
1.2.1 Scope of the Problem and Solution	3
1.2.2 Scope of Work	3
1.3 Objective	4
1.3.1 Aim	4
1.3.2 Vision and Mission	5
1.4 Structure	5
Chapter 1: Introduction	5
Chapter 2: Theoretical Foundation	6
Chapter 3: System Design	6
Chapter 4: Solution Design	6
Chapter 5: Conclusion	6
CHAPTER 2	7
2.1 Software Development Life Cycle	7
2.2 Testing	12
2.3 Open Source Software	15
2.4 Website Application	20
2.5 Frontend Development	24
2.5.1 Human-Computer Interaction	28
2.5.2 8 Golden Rules	30
2.5.3 Material	31
2.6 Virtual Data Room	32
CHAPTER 3	35
3.1 Unified Modeling Language Diagram	35
3.2 Flowchart	38

	2
3.3 User Interface / User Experience Design	41
3.4 Test Plan	44
CHAPTER 4	46
4.1 Final User Interface / User Experience Design	46
4.2 Routing	49
4.3 State	50
4.4 Communication with Identity Provider	50
4.5 Communication with Backend	50
CHAPTER 5	51
REFERENCES	52
APPENDICES	58

List of Figures

Figure 2.1 Software Development Life Cycle Agile Methodology	9
Figure 2.2 Software Taxonomy	16
Figure 2.3 Open Source Delivery	17
Figure 2.4 Model View Controller	22
Figure 2.5 Communication between the user and a web browser with Single Page Application	26
Figure 2.6 Model View View-Model Architecture Diagram of Vue	27
Figure 3.1 Use Case Diagram of a Client/User	37
Figure 3.2 Use Case Diagram of a Premium Client	37
Figure 3.3 Use Case Diagram of an Admin	38
Figure 3.4 User Flowchart	39
Figure 3.5 Administrator Flowchart	40
Figure 3.6 Sketch Design For Landing Page	41
Figure 3.7 Sketch Design for Map Page	42
Figure 3.8 Sketch Design for Production Page	42
Figure 3.9 Sketch Design for Upload File Page	43
Figure 3.10 Sketch Design for List of Clients	43
Figure 3.11 Sketch Design for Registering New Client	44
Figure 4.1 Final Design Home Landing Page	47
Figure 4.2 Final Design Map Page	47
Figure 4.3 Final Design Production Page	48
Figure 4.4 Final Design File Management Page	48

Figure 4.5 Final Design List of Clients	49
Figure 4.6 Final Design Registering New Client	49
Figure A. 1 Predefined Processors part 1	58
Figure A. 2 Predefined Processors part 2	58
Figure A. 3 Predefined Processors part 3	59
Figure A. 4 Sketch Design (a) Login Page (b) View Client Detail/Profile	59
Figure A. 5 Sketch Design Map Overlay Detail	60
Figure A. 6 Final Design (a) Login Page (b) View Client Profile Detail	60
Figure A. 7 Final Design Map Overlay Detail	61
Figure A. 8 Final Design Production Page Result Page	62

List of Tables

Table 1.1 Scope of Work of the Team

4

CHAPTER 1

INTRODUCTION

This chapter clarifies the project that the author worked on, including the background story and problem that inspired the start of the project. Moreover, scopes of work, objectives, vision and mission of this project are discussed in this chapter.

1.1 Background

In the 19th century, the premodern era, oil and gas were mainly used as a source of lubricant, medical use, source of light, and energy source [1]. At present, oil and gas have been the center of modern industrial society as well as the major geopolitical objective and majority of energy sources for nations [2] for centuries. Both crude oil and natural gas are used as the source and material of extensive types of chemicals and materials, they are the majority and dominant source of food production, pesticides, and transportation fuels [1].

Several countries, including Indonesia, are greatly reliant on the use of oil and gas for the growth of the national economy as well as the industry [3]. It is one of Indonesia's economic pillars, contributing 21.09% to Indonesia's gross domestic product (GDP) from oil exports alone in 2004 according to the Organization of Petroleum Exporting Countries (OPEC) [2]. In Indonesia, the state-owned oil and gas corporation is Pertamina. Pertamina is involved in the government oil and gas exploration and

production activities as well as help improving the Indonesian GDP by contributing in more than 60% of Indonesia's GDP [3]. Furthermore, they also provide more quality products to meet their consumer demands [3].

Oil and gas has a lot of importance and impact on Indonesia's economy. As a result, there is an enormous amount of existing dashboard software applications to help the oil and gas industry with similar features to that of the Virtual Data Room (VDR) website application. These applications include Lynx and INTviewer. INTviewer is a platform that offers users the ability to check their seismic data, geospatial integrity as well as process their datasets which could cost up to \$60,000 a year [4]. On the other hand, Lynx is a platform that offers access to geophysical and geographical information systems (GIS) while also offering the 2D and 3D seismic viewer that cost at least €250 a year for each user [5]. However, while this software is useful and benefits the oil and gas industries such as Pertamina, the expenses needed to be able to use the software are expensive. Therefore, the main goal of this project is to develop a VDR website application that has similar features at a lower cost.

1.2 Scope

The scope of the problem, especially the solution that the author team thought of, as well as the scope of work and responsibility that the author had and the task the author team had as a group is described in this chapter.

1.2.1 Scope of the Problem and Solution

The main problem of this project is the high cost of the software with preferable and useful features. To solve the problem, which was to develop a VDR website application with various features at a lower cost, the solution that the author discussed with their team was to make use of open-source libraries along with hand-picking essential features from the existing software stated previously to be used.

The features that the VDR website application uses are based on similar existing software applications, Schlumberger's ProSource Front Office [6]. The website application referenced the data visualization support of ProSource Front Office in the form of lines, bars, bubbles, and pie charts, as well as their integration with ArcGIS Arcmap to perform their geospatial analysis [7].

1.2.2 Scope of Work

The author's responsibility in this VDR website application project was the frontend of the website, the design and user interfaces (UI) / user experience (UX), as well as the business logic of the frontend development. In addition, being in charge of the making of one of the visualizations, the charts, for the oil, gas, and water. To ensure that the website application is running smoothly and as intended, testing is also required during the creation of the website application.

Meanwhile, the responsibility as a team for this project was to create the different kinds of visualizations, as well as implement the various features and functions the author's team had made to the VDR website application. The members of the team had different

responsibilities and importance to successfully finish this project as presented in table

1.1.

Table 1.1 Scope of Work of the Team

Name	Role
Vicky Vanessa	Designing the UI/UX of the front end of the website application, visualizing the data of oil, gas, and water, as well as doing the testing.
Elizabeth Chan	Designing the UI/UX, developing the visualization of information-bearing heat map using GIS, as well as the testing.
Kotrakona Harinatha Sreeya Reddy	Collecting and processing data, and using them to develop predictive models, visualizing them through diagrams such as charts while doing the data analytics

1.3 Objective

This section describes the author team's aim for this project, likewise, the vision and mission the team had set.

1.3.1 Aim

Apart from solving the problem stated in the previous section, the primary aim in developing this VDR website application is to help the oil and gas industry, through visualizing the volume of oil, gas, and water over time along with the visualization of the

reserve resources. In other words, helping the oil and gas industries to discover more profitable areas of resources. Additionally, this VDR website application can be used to obtain or derive latent information from abundant and meaningful production data. Moreover, this VDR website application can make a predictive model through the production data.

1.3.2 Vision and Mission

The vision of the author team is to make the local companies in Indonesia to have their own VDR with a lower cost as well a customizable as well as flexible features to match the company or the client needs.

Along with the vision, the mission is to develop a high-quality yet affordable website application by using an open-source library while choosing the essential features that are important in the oil and gas industries. Apart from providing advantages for the oil and gas industries, the application is also useful to aid the engineers in comprehending sophisticated data and to gain a better understanding of how the data could be used.

1.4 Structure

This thesis consists of 5 chapters as defined further in this section

Chapter 1: Introduction

In chapter 1, the topic is introduced along with the description of the author's scope, objective, and aim for this project.

Chapter 2: Theoretical Foundation

In chapter 2, the project theories, why it is carried out in a certain way, how the project will be made, as well as what it is the author used to develop the project is described and explained.

Chapter 3: System Design

In chapter 3, how the project is designed, the initial phase of the project, along with the flow of the project development is specified.

Chapter 4: Solution Design

In chapter 4, more of how each flow and interaction in the project design is explained in detail.

Chapter 5: Conclusion

In chapter 5, the final chapter, the conclusion of this thesis, and the project are stated.

CHAPTER 2

THEORETICAL FOUNDATION

This chapter discusses the theories that the author uses in order to develop and design this project. How the author and the team design the project together, what theory and concept they use to aid in building the project. Aside from that, this chapter also describes the reasoning behind theory thoroughly throughout the chapter, specifically on the approach, sources, limitation, along with the frameworks of the frontend and the development that the author is using to build the project.

2.1 Software Development Life Cycle

Software Development Life Cycle or SDLC is a framework or a model that describes a continuous process of how the project started the deployment of the product, a plan on how to develop the product, it is a cycle that will help arrange the activities to be executed during the process of development [8]. SDLC approach to complete the development is by using a step by step approach, the main phases are

- Specification
- Design
- Validation
- Evolution

The specification phases are focused on understanding and comprehending the problem as well as what is required by the customer or the user in the application [8]. Designing

is involved in planning out a solution for the problem described in the specification and designing how the application will be executed and flow [8]. The validation phase is in which the application is evaluated and tested whether or not it meets the customer's expectations [8]. Meanwhile, evolution is the latter stages of the project, how it will be maintained [8]. These phases are described and utilized differently according to the SDLC models and purposes. Throughout the cycle, it will enhance the understanding of the product, the problem, along with the solution for the problem, programming, as well as how to test and maintain the product developed [8].

A team developed application is complicated to work on without thorough planning and discussion, without a framework to structure the stages of development, developers will find it challenging to develop a product they do not have sufficient understanding of, thus, SDLC is an advantageous cycle to have in development [8]. Aside from that, SDLC aids in developing a timeline for the project, the responsibilities of each team member, as well as to keep track of the progress made during the development [8].

Among the SDLC models, there are several models that repeat the stages during revision and several with straightforward one-way models. Models such as iterative model, agile model, and V-shaped model are examples of models which have repeated stages during revision. Meanwhile, the straightforward one-way model is a traditional SDLC methodology, most known as the waterfall model.

Methodology like the agile model is great in order to reduce the amount of error later in the deployment stage as revision could be made over and over again during the

implementation stage, though it does not have proper documentation and the team might have a chance to get sidetracked while changing in between phases frequently, it offers a faster project completion time [8]. Agile methodology is a modern project management method in which the project is made and adjusted based on customer or a user requirement for the project and allows quick revision and modification to be made instantaneously [9]. The agility in completing the project is acquired by including only the process to the project and excluding all types of processes and activities that are irrelevant and will consume more time than necessary for the project [8]. Each iteration of processes involved in the agile SDLC is as shown in figure 2.1, it consists of small and manageable which could be completed within a couple of weeks.



Figure 2.1 Software Development Life Cycle Agile Methodology

Each agile project regularly involve a customer or a stakeholder representative on the team at the end of each iteration they will have to re-evaluate the progress and made some adjustment if necessary, thus, emphasizing that enhancing their communication skill is of importance for them [8] [10]. Moreover, agile also relies more on the implementation of the software rather than the documentation of the project as more requests to change the requirement may be made by the customer and should be efficiently incorporated into the program [8].

There are 12 principles behind agile software development according to the agile manifesto which are:

- Satisfy customers throughout the delivery of the software
- Accept any changes in requirements throughout the development process
- Frequently deliver working software in a short timescale
- Customer or stakeholder representatives and developers should work together throughout the project
- Provide developers the environment, support, and trust to help them finish their job
- Relay information to and from the development through face-to-face conversation
- The main assessment of progress is a working software
- Promote sustainable development
- Enhancing agility by continuous attention to outstanding excellent and design

- Simplicity is essential
- Good architecture, requirements, and design comes from good self-organizing teams
- Adjust and reflect periodically [11].

The main advantage of agile methodology as briefly described previously would be that the programmers are recommended to work in pairs as it would increase the efficiency during the development of the program as it would result in fewer errors and bugs than working alone [8]. Moreover, this would reduce the overall development and building time of the entire project, as well as customers and representatives would be able to get updated often after each iteration, making feedback, revision, and adjustment would be able to get implemented as soon as possible [8]. These advantages are what made agile methodology better than other SDLC methodologies.

In agile methodology, there exists a scrum framework to help increase the agility and flexibility in developing a project [12]. It is developed as an additional framework for the purpose of becoming a development as well as a management methodology for software projects [12]. In scrum, there are 3 roles, a product owner (PO), scrum master (SM), as well as the team [12]. A PO usually consists of a project manager (PM) and a business analyst (BA) is responsible in defining and registering the requirements and the specification of the project [12]. The team consists of developers, testers, and other relevant roles which will complete the backlog provided with the product owner [12]. Meanwhile, the scrum master is the individual who will set the scrum process in the

project and make sure that everyone involved in the project implement the scrum method provided [12]. The scrum process is implemented as the following:

1. Determining the backlog
2. Sprint planning
3. Daily stand up meeting
4. Sprint review
5. Sprint retrospective [12]

The PO will be the one to prepare the backlog for the team to follow, which the SM will arrange the priority of the task [12]. The whole team needs to participate in a meeting to evaluate the backlog and assign the task or feature that the developers will need to finish by the end of the sprint for the next sprint [12]. To supervise the progress, there should be a meeting to determine what each member of the team has accomplished prior to the meeting [12]. At the end of every sprint, the task will be demonstrated to the customer or the stakeholder representative in order to show the features or progress that have been completed [12]. Thereafter, the whole team would discuss the problem they encountered while doing their task that would need to be discontinued in the next sprint [12]. By using the Scrum framework, the quality and progress of the project could be seen clearly and done more efficiently [12].

2.2 Testing

Testing is a part of the SDLC to ensure delivery quality, it is a significant process for quality assurance, finding a defect in the program before the program is delivered to the user or the customer [13]. Testing is planned in the early stages of SDLC, which is in the requirement gathering stage and continued to be refined as the program is being

developed [14]. Not to mention, testing requires a lot of costs, in terms of finding a suitable testing case, as well as the time it took to run those tests as well as the time to fix the found errors [14]. The technique involved to evaluate the quality of the system could be categorized into 2, static and dynamic techniques [14]. The static technique is a technique that does not involve any code execution, instead, it is used to check any defects in the code, whether or not it adheres to the specifications specified on the requirement or the project documentation [14]. This technique could be used during the whole development process. On the contrary to the static technique, the dynamic technique involves the execution of the code as well as analyzing the responses and behavior to determine the existence of an error [14]. It obtains information about the program by observing the behaviour during the code execution, the methods could involve a simulation, time analysis, and prototyping [14].

There are several objectives for the testing aside from finding defects in the system, such as

- Acceptance testing: confirming that the system follows the customer requirement, commonly, tested by the customer themselves [14].
- Installation testing: to validate that the system is running in the specified environment [14].
- Alpha testing: tested by an internal user, letting them explore the system themselves without any test plan involved [14].
- Beta testing: similar to alpha testing, but require an external user to test the system instead [14].

- Functional testing: confirming whether or not the functionality is working as specified in the requirement [14].
- Non-functional testing: testing the system performance, reliability, functionality, and security [14].
- Regression testing: this is a test to verify that any modification in the code does not affect other functionality that adheres to the requirements [14].

The purpose of testing is not to show the absence of defects in the system, rather, it is to reveal the defects that are present in the system [14].

Throughout the process of SDLC, testing is utilized at different levels which could involve a part or the whole system [14]. Based on the SDLC methodology used, the testing could be applied on different phases, each phase referring to specific requirements which are subjective to different parts of the system [14]. Thus, no matter which SDLC methods are used, the team should be able to distinguish between unit, integration, system, and regression test [14]. The unit test refers to the test of a small piece of a program, usually created by a single programmer [14]. The purpose is to confirm that the feature and functionality in that unit satisfy the requirement, work, and are implemented as expected and intended [14]. Aside from that, it can also be applied to test for the interface and data structure [14]. Regarding integration testing, the purpose of this test is to find any errors that are present during the process of assimilating program components into one [14]. Although there might not be an error as an individual component, problems could arise when the components begin to interact with each other [14]. System testing is to test whether a system is working properly as required when it

is implemented into the intended environment [14]. Any errors revealed in this test could not be detected during a unit or integration testing, rather it is more to the behaviour of the components when a user interacts with it [14]. Moreover, this testing also includes the test of performance, security, reliability, and recovery of the system [14]. As mentioned beforehand, regression testing is about retesting a component or a unit after the code in the system has been modified to make sure there are no further complications after the modifications [14]. As the interest and requirement for a great program increase, so does the complexity of the website system, testing would be a major part of SDLC and quality assurance [15] [14].

2.3 Open Source Software

Open Source Software (OSS) is an up and coming feature of the software world, a user-driven, self-organizing team of contributors formed through online interaction, developing libraries and codes that are available under a license that could be reused, modified, improved by the communities [16] [17] [18]. Over the emanating development over the last decades, OSS has been becoming important as it is valuable to major information technology (IT) companies, for instance, Google, who creates a platform to host OSS projects for the community publicly in 2005 named Google Code . Figure 2.2 presents the difference between a proprietary software with open source code and a closed source code, while the corporate or individual could choose to publicize the code, according to the figure, software with open source could offer more flexibility and adaptability as it is in development by the community as it could be modified and fixed repeatedly [16].



Figure 2.2 Software Taxonomy

Both users and developers perchance licensed to use and modify the code, as well as distribute the modification and improvements they made at their discretion, however, according to Open Source Initiative (OSI) and Free Software Foundation (FSF), the free source of software in figure 2.2 does not entail that it is free of cost, but it does imply the freedom that the users and developers could use the code or program for

- Executing the program for one own's purpose
- Study how the program operates and modify it to one own's need
- Redistributing copies of the original or the modified, improvised program
- Improving the program and publicized it for the community [16]

According to Open Source Definition (OSD), a document published by OSI, there are 10 criteria to decide whether a license of software is open-source which is shown in figure 2.3 [16] [19].

1. **Free redistribution:** the software to be available for distribution without payment.
2. **Source code:** the soft to be distributed with the source or well-publicized access to it.
3. **Derived works:** the license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.
4. **Integrity of the author's source code:** distribution of "patch files" used to recreate derived works to be permitted.
5. **No discrimination against persons or groups:** the license must not discriminate against any person or group of persons.
6. **No discrimination against fields of endeavour:** for example, it may not restrict the program from being used in a business, or from being used for genetic research.
7. **Distribution of license:** the rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.
8. **License must not be specific to a product:** license rights must not depend on the software being distributed with the other specific software.
9. **License must not restrict other software:** the license must not place restrictions on other software that is distributed along with the licensed software.
10. **License must be technology-neutral:** No provision of the license may be predicated on any individual technology or style of interface.

Figure 2.3 Open Source Delivery

There are 3 key benefits amongst the numerous benefits that could be found along with the emergence of OSS [17]. The first benefit is the fact that it reduces the cost for an IT company exponentially, the cost of development, including the innovation cost as it has the efforts and contribution from proficient developers in the community throughout the world [17]. Second of all, OSS can assist organizations, specifically the small-medium enterprise (SMEs) and public institutes, to implement IT systems into their business process, as SMEs and public institutes may not be able to implement the IT systems due to the IT system's high cost of implementation and their limited budget [17]. Last but not the least, the information and knowledge from getting involved with the OSS project could be distributed, which would be valuable to those who wish to increase their proficiency in programing language as they could learn through practice [17].

An abundant amount of individuals has involved themselves with OSS projects for their own purpose and goal, which leads to the finding of 2 key motivations as of why they are willingly contributing to the OSS projects, which are regarded as intrinsic motivation and extrinsic motivation [17]. Intrinsic motivation is driven by an individual interest or enjoyment of the project or task instead of relying on external pressure or rewards at the end of the project or task [17]. The research found that over half of the individuals indicate that the enjoyment and satisfaction that they have during the programming and involvement in OSS projects, as well as their creativity and ideas that are implemented in the OSS projects, conferred a great sense of accomplishment motivate them to contribute to it [17]. Contrary to intrinsic motivation, extrinsic motivation is driven by an outcome such as profits and reputation, strictly speaking, the motivation will disappear once those rewards are withdrawn [17]. Some large OSS projects and contributors have been analyzed that the contributions have been involved with the pursuit of rewards and reputations [17]. Aside from those, research also found that a sense of reciprocity and job opportunities are the other factors of extrinsic motivation, some individuals would aid the OSS community by responding to the question in the community for them to be able to obtain assistance from others when require in the future, moreover, some IT companies would like to recruit capable programmers from the OSS communities which encourage them to be involved in the OSS projects [17].

The success of OSS could be measured to a limited extent as the factors researched to measure it are relatives to conditions and factors of the users. The main methods in the research are to assess it through the number of downloads and Concurrent Version System (CVS) commits [17]. However, the number of users downloading and

subscribing to the OSS project could not be correct representative of the OSS success, users could download the software as a trial and deleted it afterward, in the end, those numbers could only signify the level of exposure the OSS software has in the community [17]. To counter the defectiveness in the assessment, some use the frequency of CVS commits instead to assess the success of OSS, nonetheless, it could imply another defect in the project, which could imply that the contributors often found bugs and constantly revise and fix the distributed version [17]. Thus, to certainly assess the success of OSS projects, scholars would use the frequency of how often the code in the OSS project would be reused in other OSS projects.

All software, including OSS, is subjected to copyright law, it is how the creators or developers are able to maintain their control of work, which is the exclusive legal right to control the rule for copying, modifying, and distribution of the software [19]. The copyright holder is an individual or organization that has control over the work, they could give permission to other individuals to use, modify, and redistribute their software according to certain provisions through licensing [19]. OSS license is different than the general commercial license as it grants more access to the individual, allowing access to the source code and the right to modify it [19]. Copyleft, enforced by copyright law, is a term established by the free software community for a license condition to ensure that all versions of modification made to the source code can be copied, modified, and redistributed similar to how it is with the original [19]. There are frequently used open-source licenses for instance the Berkeley System Distribution (BSD) license, the Massachusetts Institute of Technology (MIT) license, and Apache. The BSD license is one of the least restricted as well as most acknowledged open-source licenses, it allows

redistribution of source code, modified or not for as long as the work has its copyright notice regarding disclaimer and limitation on liability found in the license [19]. MIT license is quite identical with BSD license as it authorizes the reuse of open source code if only MIT license includes proprietary software in their term, moreover, it also allows the use of the copyright holder's name the promotion of the software [19]. Apache is similar to BSD and MIT licenses, as it grants the use of software without the obligation to redistribute the code of any version modification of the software, aside from that, it supports the clause of patent licensing and termination [19]. Understanding the difference between each licensing is important in order to use OSS.

2.4 Website Application

In the beginning, World Wide Web (WWW) was introduced in order assist to achieve access of information from any source in a consistent and convenient method in the 1990s by the Conseil Européen pour la Recherche Nucléaire (CERN) in order to accommodate the enormous amount of data and documents required to be shared with other scientists [20]. The method adopted is by using HyperText Transfer Protocol (HTTP) which was designed to allow one computer to request data and documents from another computer which is from the server computer, this helps users to access the documents available through the computer, WWW was perceived as a large archive of information that allows access to numerous amount of users [20]. Website application refers to the program that could return the page of data from the server to return its result to the client, some described it as a combination of hypermedia and information system [20]. At present, website application is a powerful platform that supports individuals,

SMEs, communities of users, as well as large corporate businesses [20]. In 1999, Fratenali stated the following as a requirement of a website application:

- The need to handle both structured and non-structured data, referring to the database record and multimedia items respectively
- Navigational interface in order to assist exploratory access
- Impressive graphical quality
- Customizable and the dynamic adaptive content structure, navigation, as well as presentation style [20]

Website, a combination of connected and related website pages which occupied on a single server are available for a variety of categories and usages [20], continues to rapidly evolve though they still maintain the basic concepts of WWW which are Uniform Resource Application (URL), HyperText Markup Language (HTML), and HTTP [20]. URL is a naming system that prescribes the way objects could be identified and located as well as searchable based on their attributes or names [20]. HTML is a document on the web in which the content could be displayed and linked to other documents to achieve enhanced visual presentation [20]. Meanwhile, HTTP is a communication of a request and reply protocol that contains the most used basic operations of GET, PUSH, PUT, DELETE which could be requested to the URL [20].

Website applications use a similar concept to that of software, they have a UI, user interactable, may contain and manage a vast amount of data on the server, as well as the Model View Controller (MVC) design pattern, in fact, several frameworks are developed in support of the design and development of MVC website application [20]. The model

in MVC represents the object-oriented class that interacts with the database or the data storage of the application, the view is the HTML pages that represent what people will see in the website application, the controller would contain the logic in the application [20]. As presented in figure 2.4, the controller manipulates the data that would be updated on the view as well as respond to the queries from the view that the user made [20].

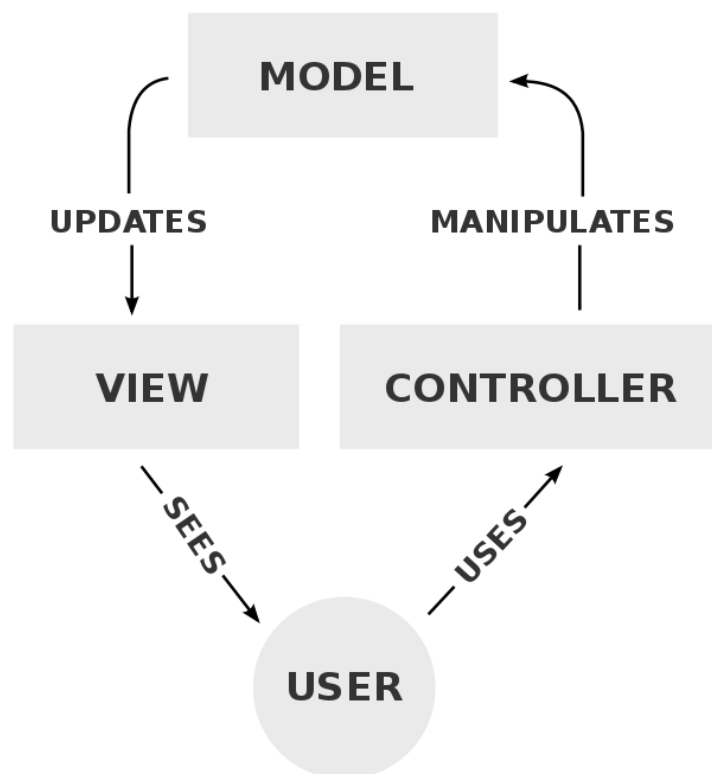


Figure 2.4 Model View Controller

Aside from that, research on modern website applications pointed out that there are several common features in website applications which are

- Search
- Tagging

- User participation
- UI and collaboration

Searching and tagging have been a mechanism to assist users to find the information that they need amongst the abundance amount of information on a website [20]. User participation is to create and design the structure of the data [20]. While UI and collaboration mostly exist in a messaging system, for instance, Google Mail and Skype, which would notify the user in real-time when someone in their contact is online [20].

As technology advanced, it becomes mandatory for a corporation, big or small, to have a website to provide their service to users at a global level, thus, developers need to commit a lot of effort to the design of the website so it could be integrated with different services [21]. However, there is some websites design that fails to achieve the desirable expectation on the website across the browsers for the reason that browser compatibility becoming an important issue that affects functionalities and UI components [21] [22]. The majority of website designers believe that they should be able to support as many browsers as they could, but as a matter of fact, it is nigh impossible to support every browser [21]. At present, there are hundreds of web browsers with the top major browsers being Chrome, Safari, Internet Explorer (IE), Firefox, and Opera, on top of that, there are multiple versions of each browser, furthermore, each browser process a web code in a different way, and it is strenuous for a website designer to adjust and modify the code to allow the website to show the same behaviour in every browser [21]. There are several web pages component that effects the cross-browser compatibility of a website, for instance, blinking, active X control, image format, video tag, and ajax, these features are supported at a different degree [21]. Aside from the component and browser

differences, there are also features that interfere with the compatibility as well, computer types, screen sizes, font size, add ons, as well as third parties [21].

Website developers or designers could not assume that the website application they developed will work perfectly fine with all the expected functions and UI/UX across all browsers without doing cross-browser testing [21]. Organizations and developers will need to put a lot of effort in order to maintaining the functionality of the website across browsers to avoid loss of reputation in businesses [21]. It is also recommended for website developers to follow the standard suggested by the World Wide Web Consortium (W3C) [21].

2.5 Frontend Development

Website technology has expanded over the years along with the development of the internet, especially the HTML that leads the frontend development [23]. Frontend development has evolved simultaneously with the demand of business needs, the traditional model which used JQuery was not enough to fully develop the website frontend that meets the interest of the current era of website development [24]. JQuery is an open-source javascript library to help the handling of HTML Document Object Model (DOM), ajax, and Cascading Style Sheet (CSS), although it is still widely used in website development, more complex libraries that fit the business needs, for instance, for Single Page Application (SPA) and Progressive Web Application (PWA) [24].

Developers frequently and especially use front-end frameworks when developing a large and sophisticated website application, it allows them to reuse repetitive code, saving time in designing and developing the application [25]. There are numerous javascript

frameworks and libraries for SPA, among those, the most popular front-end frameworks are Angular, React, and Vue respectively [23] [26]. SPA is a website interface consisting of individual components that can be easily updated or replaced individually, thus, preventing a reload of the page for every action the user made while loading dynamically [25] [26]. In accordance with the modern website technology and business needs, especially as a developer, the author team decided on using SPA for the website application.

The initial request, from the user web browser, regularly sends a request through HTTP, the server replied to it through javascript libraries or dependencies used in the website as presented in figure 2.5 [25]. Meanwhile, SPA act as an intermediary between the user input and the backend server as it handles the user input processes within the SPA, sending any data or request related to the input done through JavaScript Object Notation (JSON) to the backend server which is made asynchronously, the server will send back JSON data in responding to the SPA, the DOM component will re-render based on the responses, which is on the sixth step of the communication process based on figure 2.5, furthermore, this communication process will be repeated in a loop until the user decided to end or exit the SPA website [25].

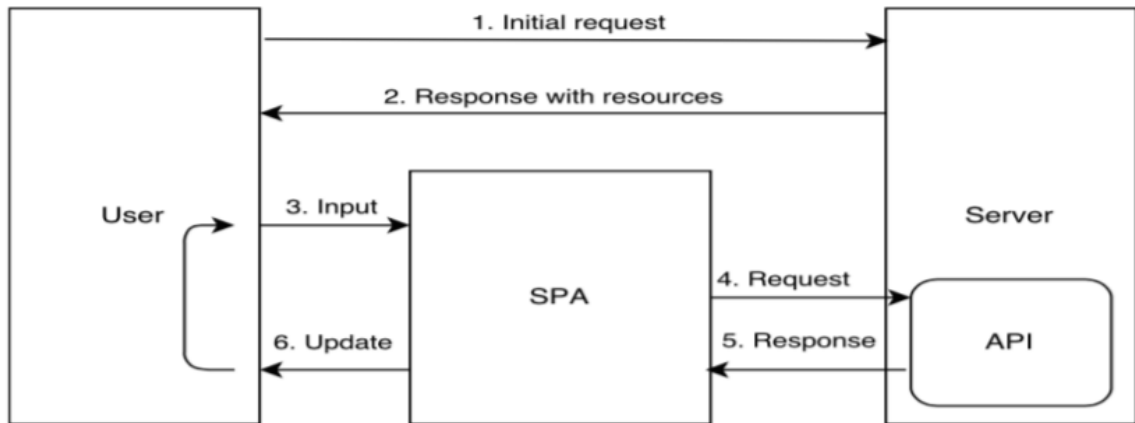


Figure 2.5 Communication between the user and a web browser with Single Page Application

Moreover, as a result of the communication process described, SPA has a quick response/render time for the user, while for the developer, it is advantageous for them as the presentation side and logic are separated, allowing them to be able to work and update on them separately [25]. Aside from that, the data transmission, sending, and receiving to and from the server are faster [25].

As mentioned previously, Vue is one of the popular open-source Javascript ES6 frontend frameworks that create a SPA website with comprehensive and high-performance in developing an interactive interface [25] through Model View View-Model (MVVM) design pattern that supports its data-driving and component-based development [24], which the author team has decided to use as the frontend framework of the website application.

MVVM architecture is unique from the conventional operation of DOM nodes, as shown in figure 2.6, Vue is more focused on the view-model aspect of MVVM, being bonded to both directions responsively, from the DOM listener to the data operation, and from the data to the view DOM [25]. The bindings made Vue respond asynchronously to updates from the data of the user in order for the DOM to be synchronized with the updates [25] [27].

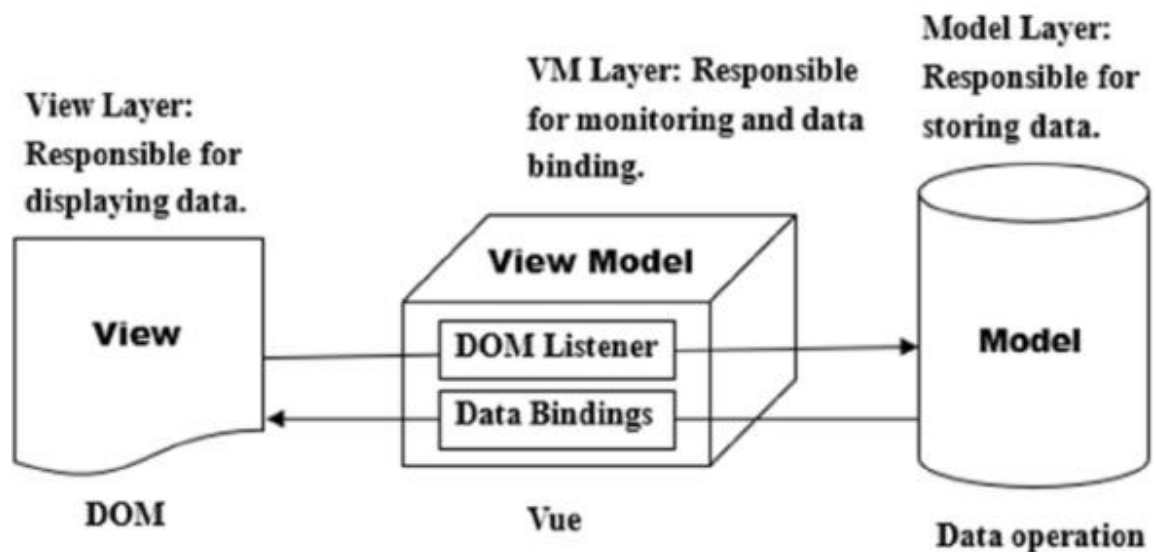


Figure 2.6 Model View View-Model Architecture Diagram of Vue

Vue is a great choice as a frontend framework for the reason that it has great advantages over other frontend frameworks such as;

- Greater efficiency, usability, user experience, fast and light.
- Rendering broad modular models without additional time.
- Great reactivity between HTML and Javascript code
- DOM modified accordingly to any updates from user [26]

All these advantages are what makes it great as a SPA, furthermore, Vue is adaptable to the user, usable for users with poor connection, as well as great for computers with insufficient resources [26].

2.5.1 Human-Computer Interaction

Human-Computer Interaction (HCI) is a term for the interaction between a human user and a computer system that accomplished something, to ensure that the interaction between the 2 parties is effective, efficient, and satisfies the user [28]. HCI term emerged in 1980 and has been a successful technological and scientific undertaking by achieving an effective integration of software engineering with human factors or ergonomics of computing systems by using the concept of cognitive science [29] [30]. Human factors are science as well as a field of engineering related to human capabilities, limitation, and performance as well as system design that are efficient, comfortable, and pleasant for the human who uses the system [30]. HCI is quite broad of a subject as it interests expertise from cognitive psychology, sociology, linguistic, cognitive science, as well as computer science [30].

HCI in relation to computer science is connected with the Graphical User Interface (GUI) of a software or an application [30]. GUI entered the mainstream in 1984 to replace the text-based Command Line Interfaces (CLI) to be more user-friendly and enable users to be able to control what and how they see [30] [31]. The UI principles in HCI research according to the Human Factors Engineering by Christopher D. Wickens in 1998 introduced 13 principles that have been perceived as an outstanding guide for designers in the industry which could be categorized into 4 different principle categories

which are perceptual principles, mental model principles, principle-based on attention, as well as memory principles [32] as shown as followed

Perceptual principles:

- Legible display
- Avoiding absolute judgment limits
- Top-down processing
- Redundancy gain
- Avoiding similarities

In this category of principles described that objects in the design should be clear to see and easy to read, signals to the users should be according to the UX, signals that could be understood by presenting in an alternative form, as well as removing similar features [32].

Mental model principles:

- Pictorial realism
- Moving part

This category explains that a character needs to represent what it seems, for instance, a higher bar in a bar graph indicates a higher value, following that, moving objects should need to reflect the change in value [32].

Principles-based on attention:

- Minimizing interaction cost
- Compatibility principle
- Multiple resources

In this category, the context means that regularly used components should be located efficiently to minimize the user's time and effort, the visual presentation should have

consistency, along with using information that would help the user process signals easily such as using red colour to present a warning to the user [32].

Memory principles:

- Replacing memory with visual information
- Predictive aiding
- Consistency

This category describes that a good design should not have the user remember important information in their memory, using perceptual than cognitive to present the user's progress, as well as using the same consistency between versions [32].

2.5.2 8 Golden Rules

According to Ben Shneiderman, there are 8 golden rules that are relevant in the majority of an interactive system as explained in the following:

1. Strive for consistency

Consistency in terms of the design of the interfaces, the whole system should have the same menu, colour, layouts, and fonts [33].

2. Seek universal usability

Considering the different and diverse users that will use the system, the design of the system should analyze the target user, international variation, as well as the differences in technology [33].

3. Offer informative feedback

For every action the user made in the system, there should be a response from the interface, a change in visual presentation in the object they interacted with would conveniently show the changes explicitly to the user [33].

4. Design dialogs to yield closure

A sequence or a set of processes and actions that have the beginning and an end as an indicator to organize and prepare for the next sequence of action [33].

5. Prevent errors

Creating a design that would help prevent users to make errors in the interface as well as providing a simple step of recovery in the event that the users made an error [33].

6. Permit easy reversal of actions

Users should have access to reverse their actions to offer them in exploring the available options [33].

7. Keep users in control

Users would want and hope to be the one in charge of the interface and for the interface to be able to respond to them, not wishing for any unexpected actions from the interface [33].

8. Reduce short-term memory load

There are limits to how much humans could process information in short-term memory, the system of the design should not require the user to memorize the information that would be needed in another display in the system [33].

These 8 rules could be further interpreted and modified in different environments, however, it provides a good start for designers [33].

2.5.3 Material

Material is an adaptable system consisting of guidelines, components, as well as tools to support a good practice of UI design made by Google [34]. Material components are

available for Android, iOS, web, and Flutter to help developers develop their UI in accordance with the guidelines provided by Material with customizable UI components, enabling developers to build an appealing and functional UX [34] [35]. It is inspired by the real physical world and its material and textures, for instance, how light is reflected and cast their shadow, specifically using paper and ink as the medium [34]. Material components cover a range of the essentials of UI, for instance:

- Display
- Navigation
- Actions
- Input
- Communication [34]

Material Design in the web is available on SPA frameworks, such as Angular Material for Angular, MUI for React, as well as Vuetify for Vue.

Vuetify is a UI framework built for Vuejs, and unlike Material Design on other frameworks, Vuetify is designed from scratch to make it easy for learning [34]. It has active development as it is being patched frequently, seeking out issues in the community, having more bug fixes, and got enhanced often [34]. Vuetify has a massive community to support developers and collaborate with them as well [34].

2.6 Virtual Data Room

Data room is a promising valuable asset of intelligence about oil and gas resources that are available for sale, undoubtedly used by the oil and gas companies who wish to distribute their share of assets [36]. Interested buyers' Mergers and Acquisitions (M&A)

team would visit the data room to analyze and inspect the data placed and arranged by the seller then decide whether or not the assets are worth purchasing [36]. Data rooms, with an abundant amount of confidential sources of oil and gas information, have different forms, the Physical Data Room (PDR), VDR, or both combined [36]. Visiting a data room depends on the seller, whether or not they allow it, most often than not, they will not allow a second look once they decided to close it [36]. Thus, both sellers and buyers must be conscientious assure everything is going well [36].

PDR is a closely monitored room to ensure secrecy of the data placed by the seller or their representatives, it has a lot of disadvantages as it is expensive, time-consuming, as well as an inconvenient geographical location [36]. VDR is built as an alternative to a cumbersome way of PDR, it also has replaced the majority of PDR [36]. VDR is a controlled-access website that allows loads of documents to be uploaded for a limitless amount of clients to analyze the documents [36]. VDR is available and accessible continuously non-stop and in some circumstances should be easy and quick to set up. Though it would require a lot of time on the clients to ensure that the documents uploaded to the VDR are appropriate. Any information on the VDR could be updated, modified, or deleted at any time [36].

VDR has a lot of benefits over the traditional PDR for the clients [36]. VDR offers information and documents at a faster speed, lower fees, as well as better efficiency [36]. Moreover, since the platform is online, it allows access no matter the current location and region of the clients [36]. Aside from that, by using VDR, clients do not have to go their way to send their teams to an inconvenient location of a PDR [36]. At times, it

could be complicated for the clients to send the right team to the PDR, but the existence of VDR saves that complication [36]. Through VDR, the right team could analyze the data and information [36]. Based on the Schlumberger, there are 3 possible features to VDR which are

- Full series of petrotechnical solution
- Generate and download reports as well as other documentations
- Available to access from anywhere around the world regardless of Operating System (OS) and devices [37].

CHAPTER 3

SYSTEM DESIGN

This chapter describes and presents the visualization of how the website application works, how it interacts with the user and the server, the flow of the interaction of the user with the website application, the UI/UX initial prototype design, as well as how the test phase is going to be conducted. It focuses on visualizing the design of the website application at the initial phase of the SDLC.

3.1 Unified Modeling Language Diagram

Unified Modeling Language (UML) diagram is used as a visual language to interpret and document a system as well as to present how the user could use the system and the behaviour [38] [39]. There are several types of UML diagrams that could be used by developers, such as class diagrams, activity diagrams, use case diagrams, and component diagrams [39]. Each type of UML diagram has a different perspective for the customer and developer in a different scale of perception [38].

Use case diagram is used to aid the developers to illustrate the functionality of the system. This includes the function requirements, the relationship between the ‘actors’ and the functions as well as the expected process of the system [38]. The actor is the individual who is interacting with the system [38].

Figure 3.1 represents the UML use case diagram, the main actor in this figure is the premium client and the regular client, which falls under the same category of a client/user. Meanwhile, the secondary actor is the identity provider as well as the backend. The flow of the use case diagram is read from left to right, implying the start of the flow is starting from the client/user. Everything inside the rectangle container represents the functions or the process that the client/user can go through. In figure 3.1, there are a total of 6 processes that the client/user can go through in the VDR website application. Every single process will go through the identity provider to confirm that the client/user has access to the process that they are accessing. While every process related to showing data and modifying it will be sent to the backend for the request.

Whilst figure 3.1 show every process that a client could access, figure 3.2 present the process that is exclusive for a premium client. The process that is available to the premium client is accessing the predictive model that the author's team decided to develop and integrate into the VDR website application.

Moreover, the process of the administrator in the website application could be seen in figure 3.3. As administrators, they could access 6 processes which are all related to managing and modification of the client/user. As shown in figure 3.1, there is no option for the user to register, a client/user would need to contact the administrator to get their account registered to be able to access the website application

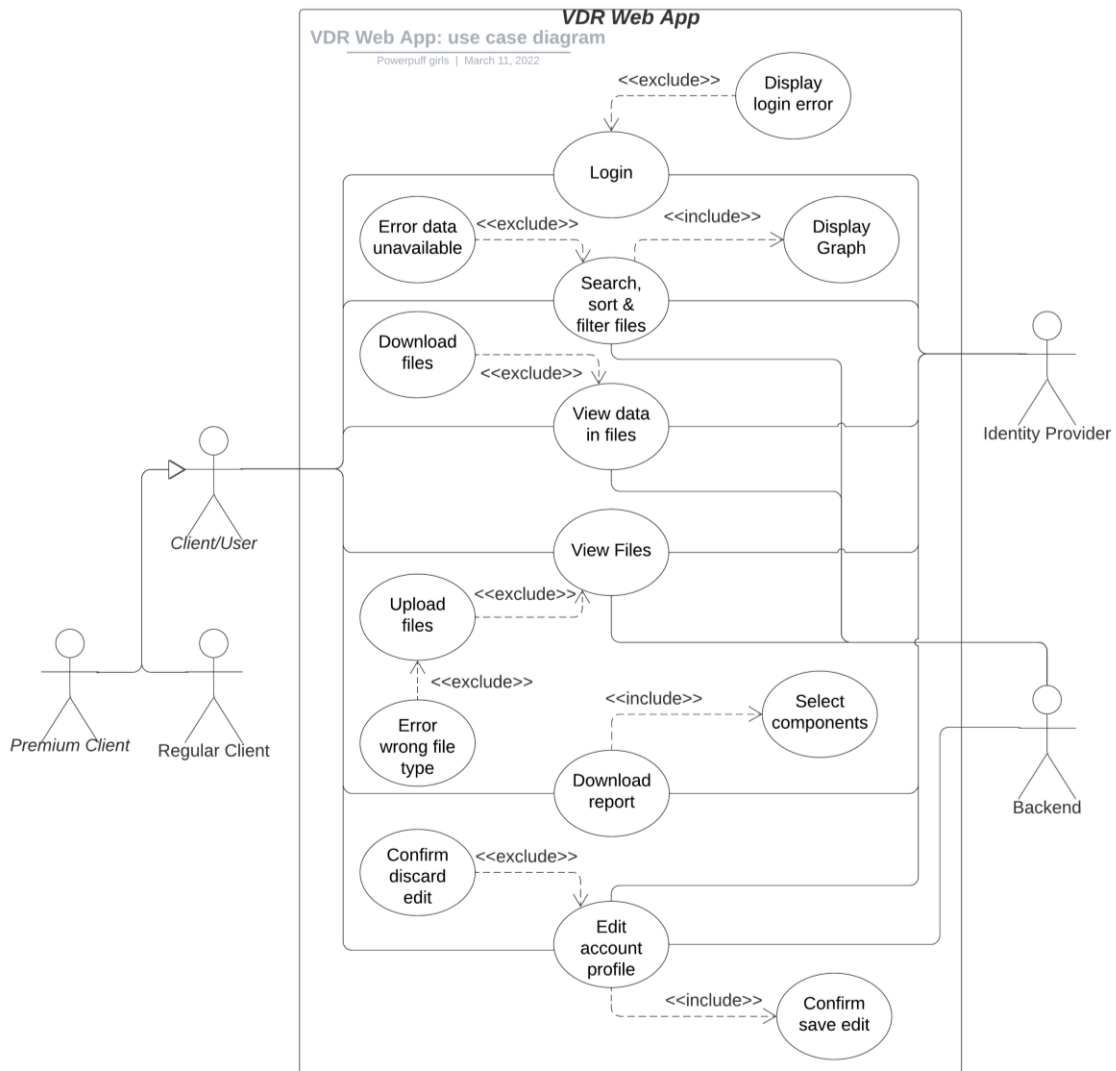


Figure 3.1 Use Case Diagram of a Client/User

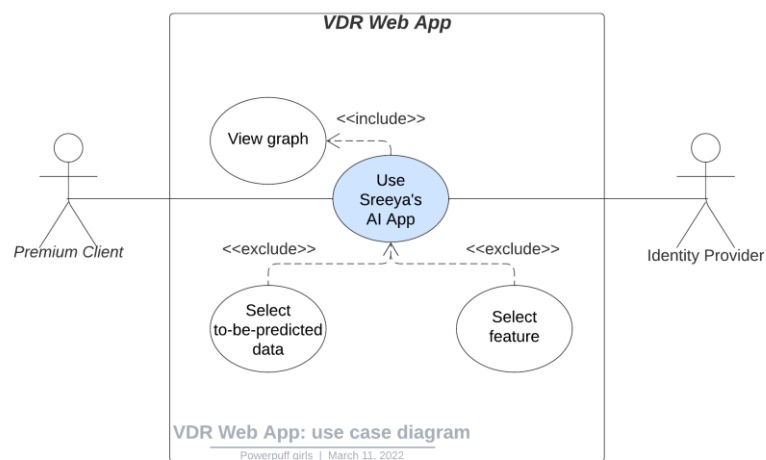


Figure 3.2 Use Case Diagram of a Premium Client

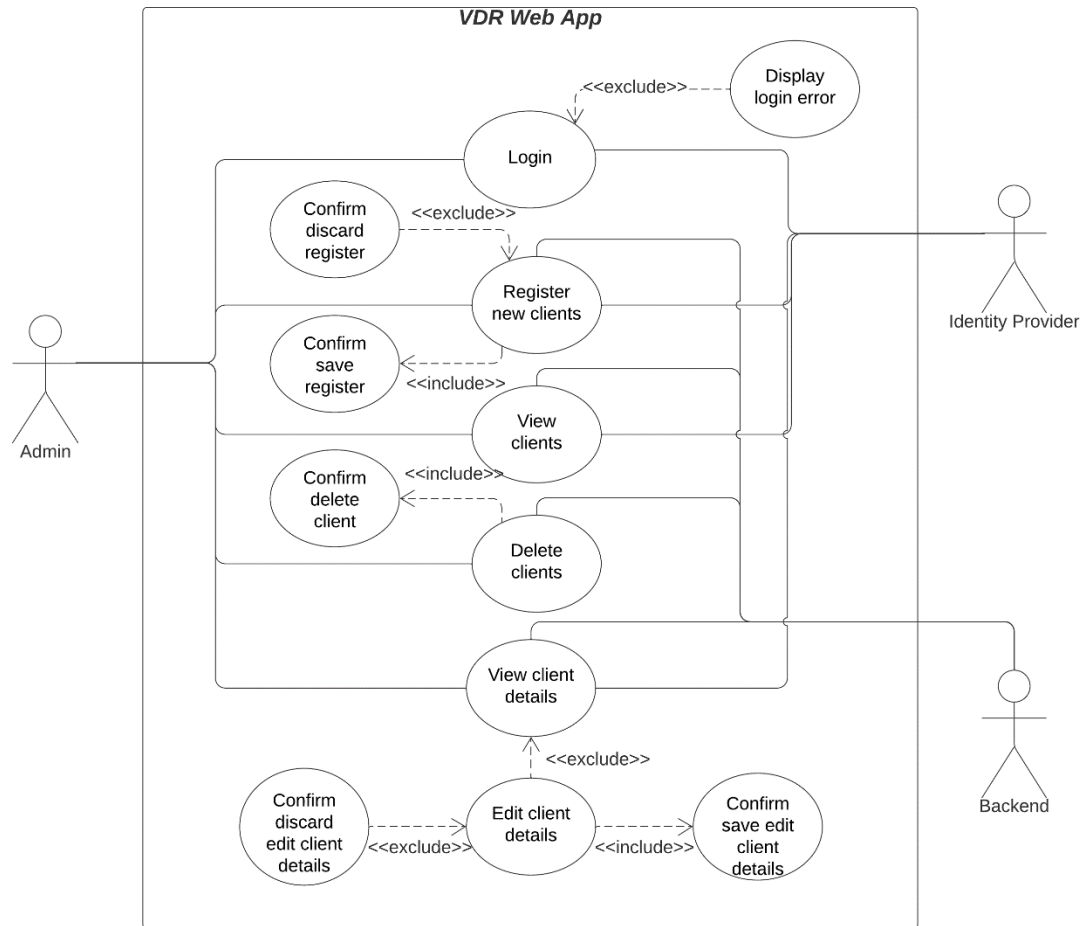


Figure 3.3 Use Case Diagram of an Admin

3.2 Flowchart

A flowchart is a diagram that portrays a process, system, or computer algorithm [40].

This diagram is commonly used in a variety of fields, from documents, study plans, to complex processes in a comprehensible diagram [40]. A flowchart uses several shapes to define its functionalities and processes, for instance, rectangles, diamonds, and ovals which are connected to one another [40]. Moreover, since flowchart could be used in a variety of forms, it is one of the most frequently used diagrams, used for both technical and non-technical fields [40].

Figure 3.4 presents the flowchart of a user, the process that the user could go through.\

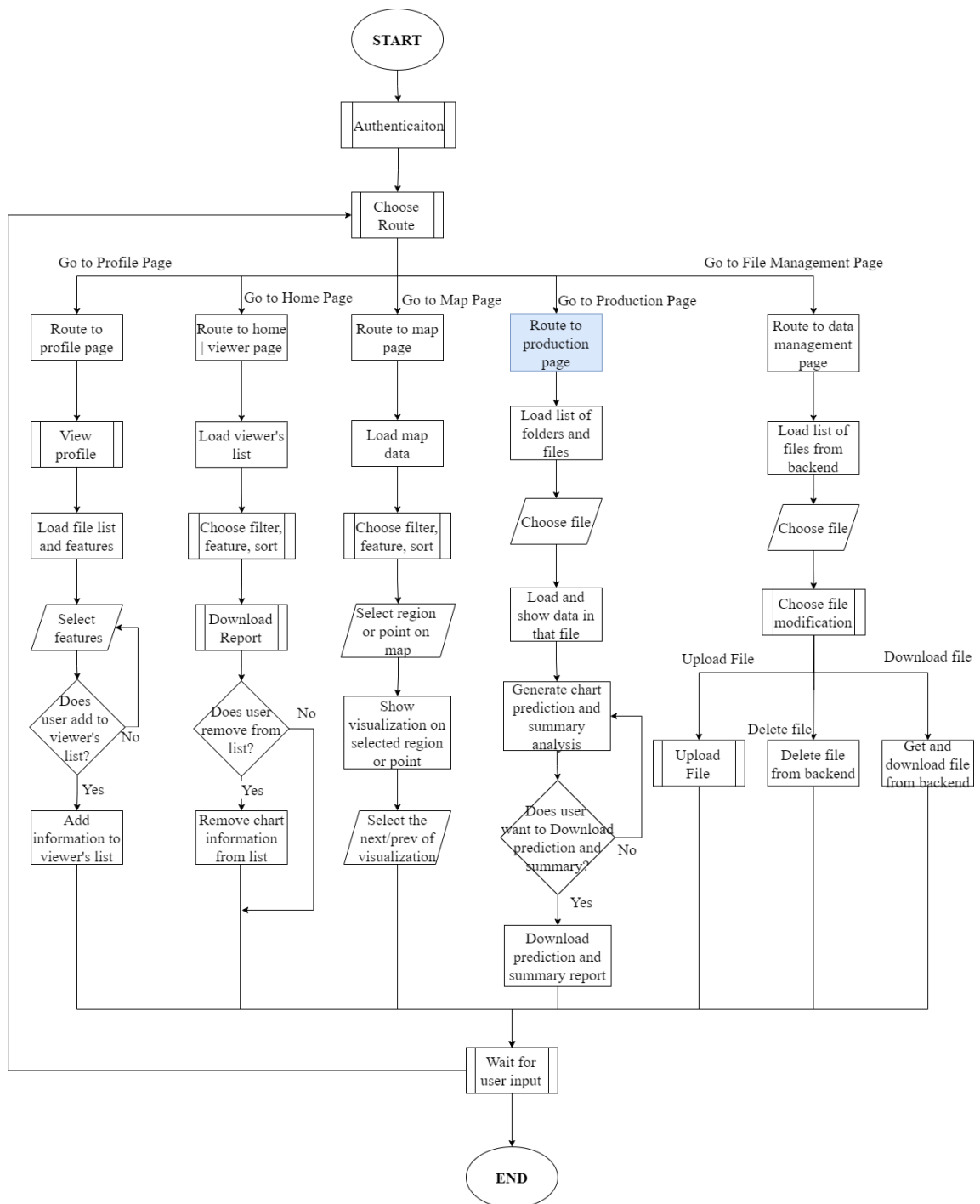


Figure 3.4 User Flowchart

Inside figure 3.4 there are rectangles with a vertical line on the left and right side of the rectangle, this shape signifies predefined processors which are presented in figure A.1 and figure A.2. Meanwhile, figure 3.5 show the flowchart of the administrator, with their predefined processor defined in figure A.3.

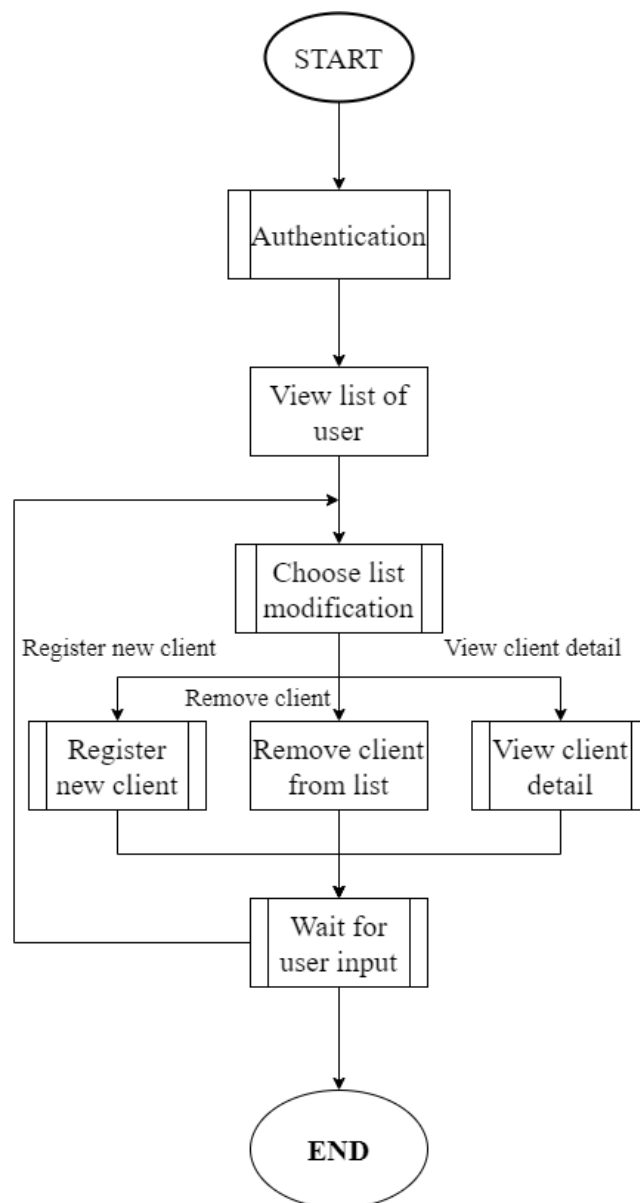


Figure 3.5 Administrator Flowchart

3.3 User Interface / User Experience Design

The design of the UI/UX that the author's team planning on using for the website application is illustrated through Figma. Figma is a powerful collaborative interface design tool that allows the team to create a design for the project together. The initial sketch of the UI/UX design for the user can be seen in figure 3.6, figure 3.7, figure 3.8, and figure 3.9 for the landing page, map page, production page, and upload file page respectively.

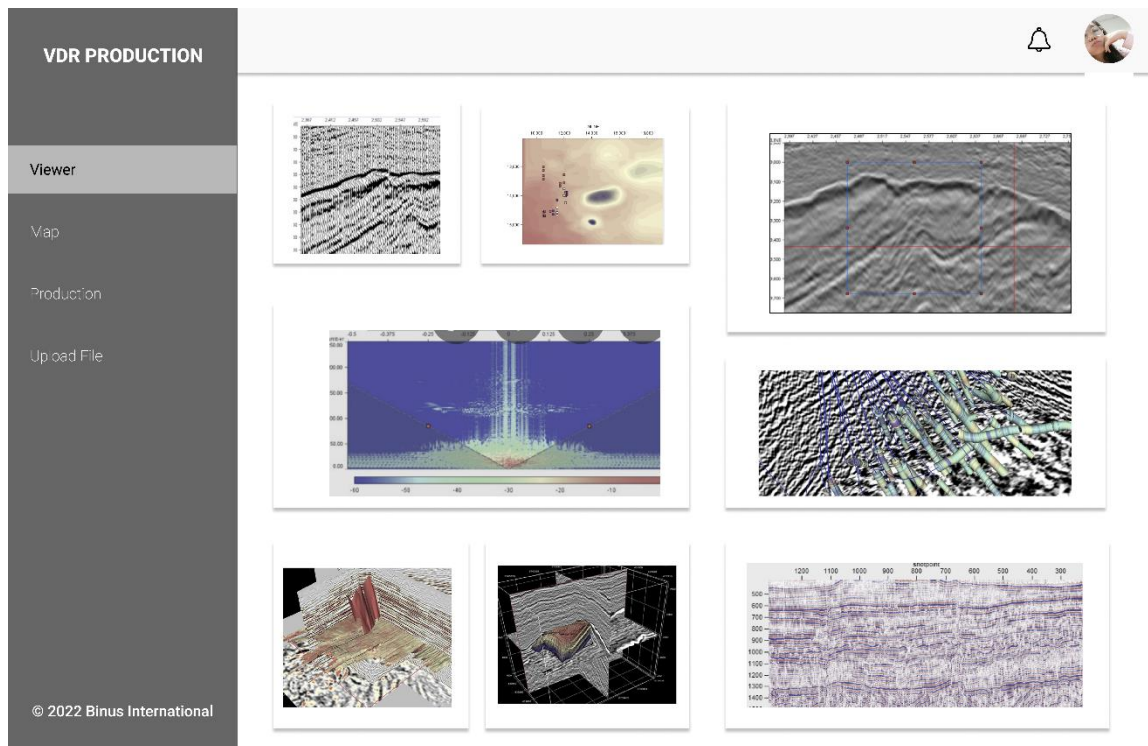


Figure 3.6 Sketch Design For Landing Page

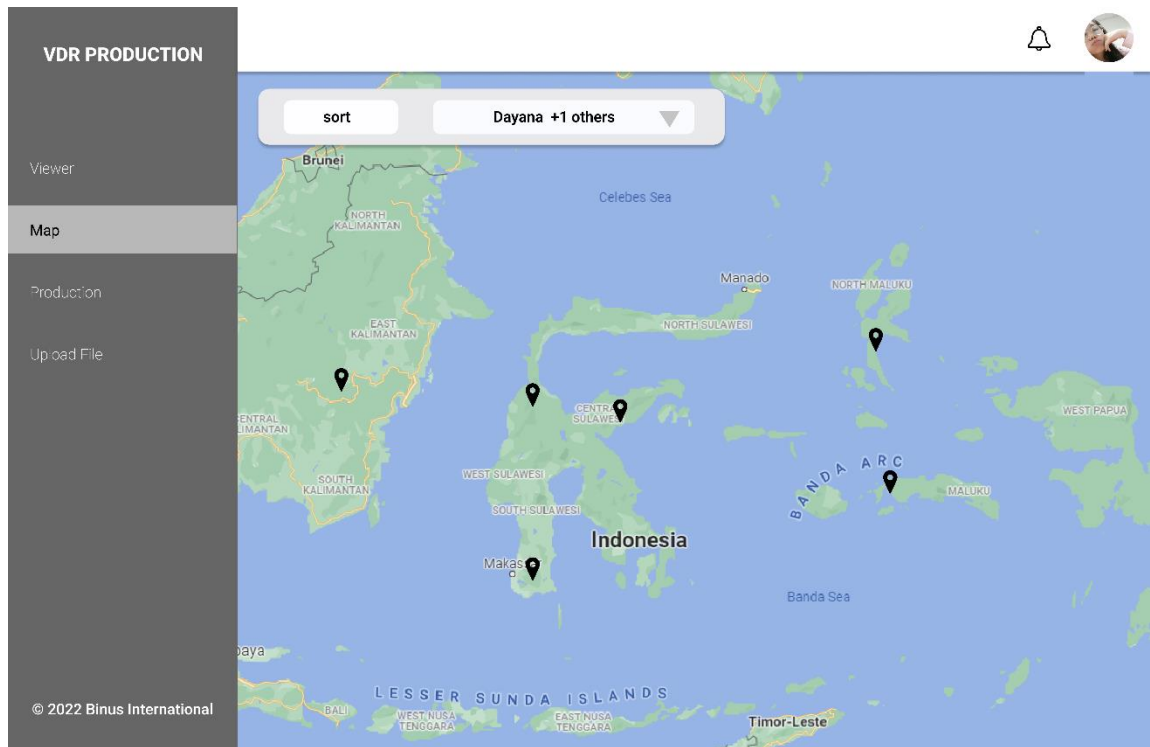


Figure 3.7 Sketch Design for Map Page

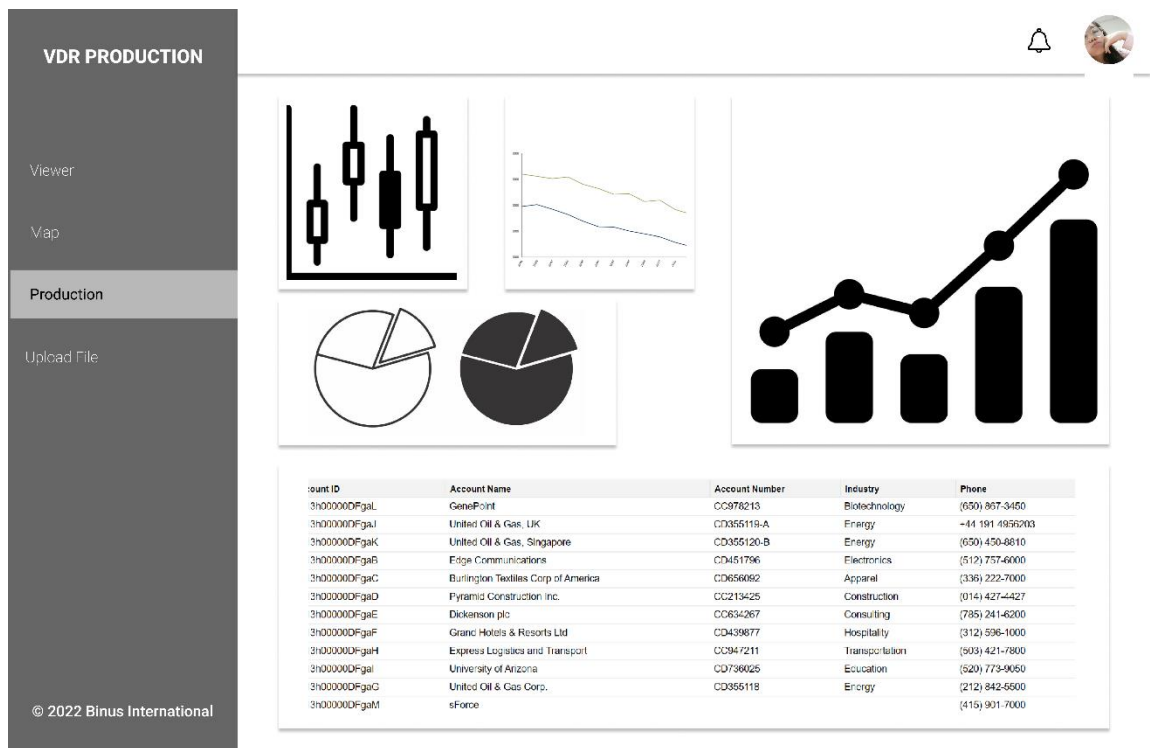


Figure 3.8 Sketch Design for Production Page

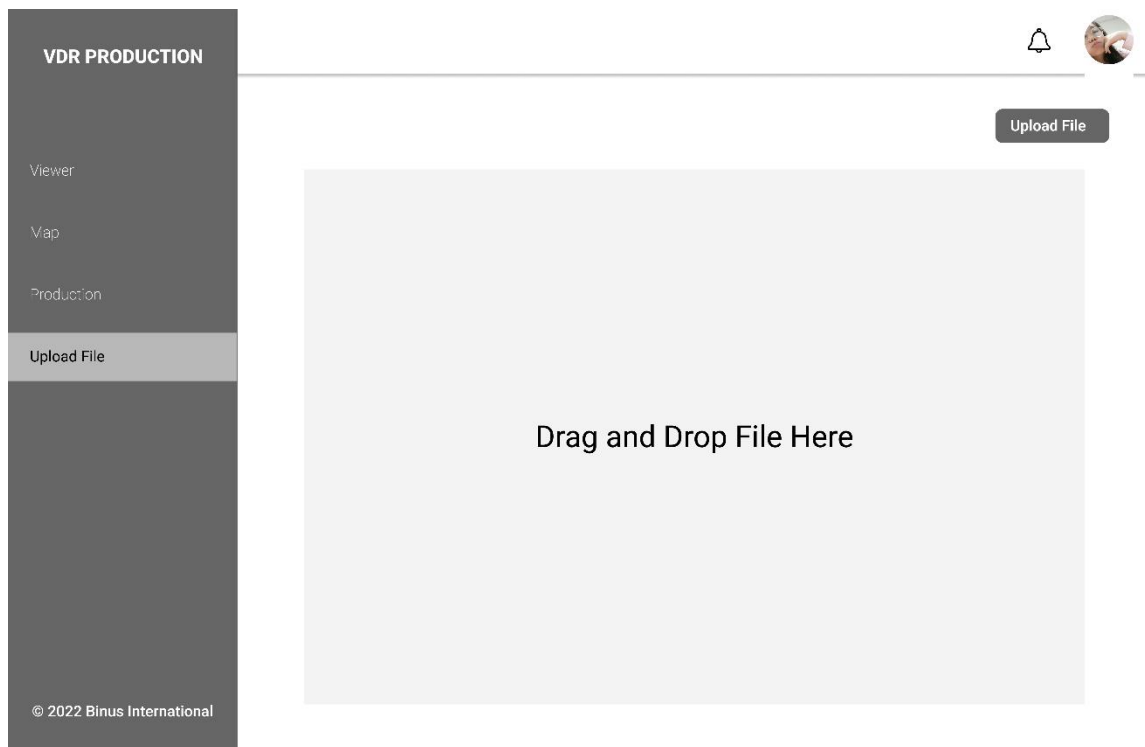


Figure 3.9 Sketch Design for Upload File Page

Additionally, the sketch UI/UX designs for the admin are presented in figure 3.10 and figure 3.11 for viewing the list of client pages as well as registering new client pages respectively.

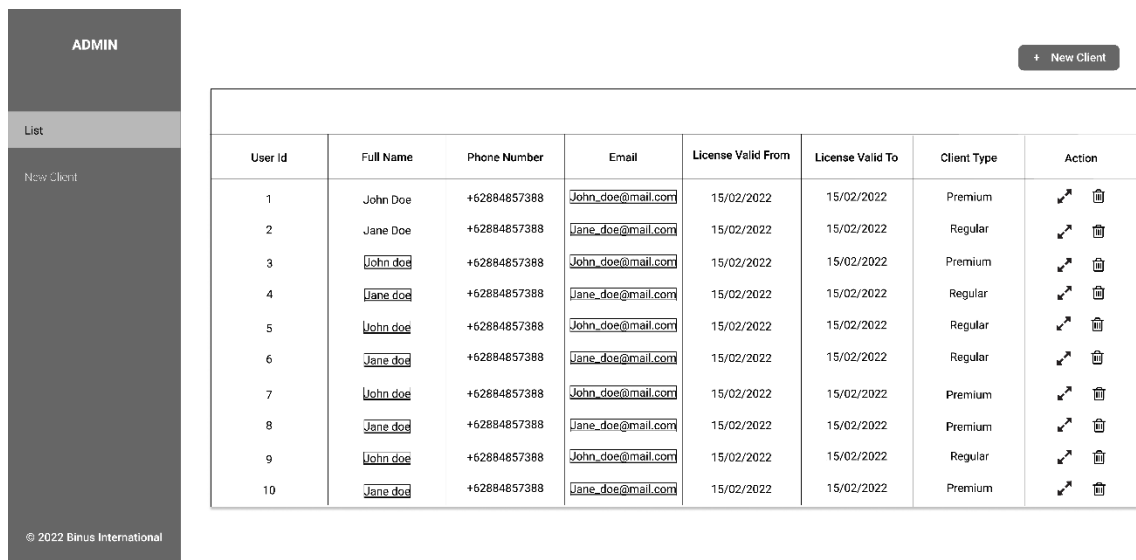
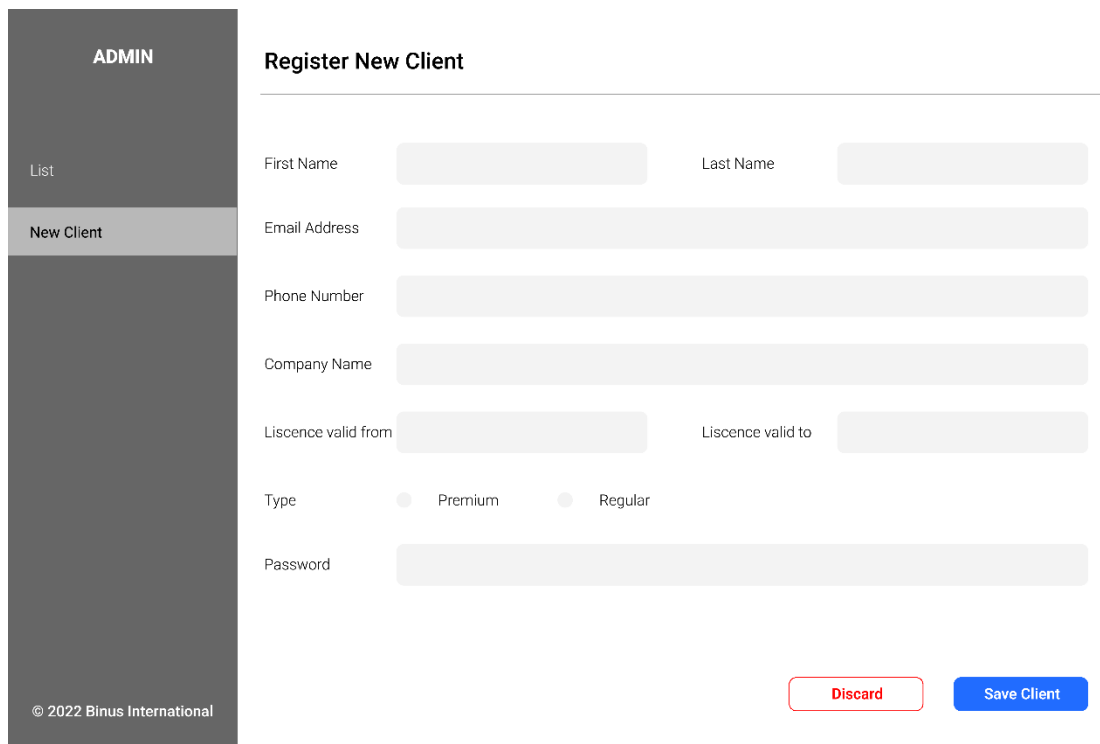


Figure 3.10 Sketch Design for List of Clients



The image shows a web interface for registering a new client. On the left is a dark grey sidebar with the word 'ADMIN' at the top. Below it are two menu items: 'List' and 'New Client', with 'New Client' highlighted in a lighter grey. At the bottom of the sidebar is the copyright notice '© 2022 Binus International'. The main content area is titled 'Register New Client' and contains a form with the following fields: 'First Name' and 'Last Name' (two separate input boxes), 'Email Address' (one wide input box), 'Phone Number' (one wide input box), 'Company Name' (one wide input box), 'Licence valid from' and 'Licence valid to' (two separate input boxes), 'Type' (with radio buttons for 'Premium' and 'Regular'), and 'Password' (one wide input box). At the bottom right of the form are two buttons: a red 'Discard' button and a blue 'Save Client' button.

Figure 3.11 Sketch Design for Registering New Client

The login page, client/user profile information, as well as the overlay in the map page, could be seen in figure A.4 as well as figure A.5.

3.4 Test Plan

The test of the web frontend that the author planned to perform is unit testing. Unit testing allows an isolation test for an individual set of codes [41]. There are several unit testing frameworks available to support a developers' unit testing, for instance, Jest and Mocha [41]. The author chooses Jest as the unit testing framework, it is part of the Javascript test framework which focuses on simplicity [41].

The author will conduct the test to evaluate the client's main functions of the website application, which are

- Login
- Adding new file(s)
- Remove and download the file(s)
- Download report
- ???

CHAPTER 4

SOLUTION DESIGN

This chapter will specifically elaborate and explain the solution on how the frontend website application would be built and connected. The final UI/UX design that the author modified from the sketch as they are developing the website, how the web application is operated through route and state, as well as the communication of the website application with the identity provider and the backend.

4.1 Final User Interface / User Experience Design

After the sketch, some modifications in the UI/UX design are made based on the sketch, the home or the landing page as well as the file management page is changed as presented in figure 4.1 and figure 4.4 respectively. The login page and user/client profile of the website application is presented in figure A.6. According to the design, the map page as shown in figure 4.2 is the page that will have the map GIS implemented, the overlay can be seen in figure A.7. Furthermore, the production page as presented in figure 4.3 is the page that will contain the prediction model implemented in the website application. The result of the prediction model is shown in figure A.8. The pages for the administrator side are shown in figure 4.5 and figure 4.6, presenting how the administrator will view the clients and register new client.

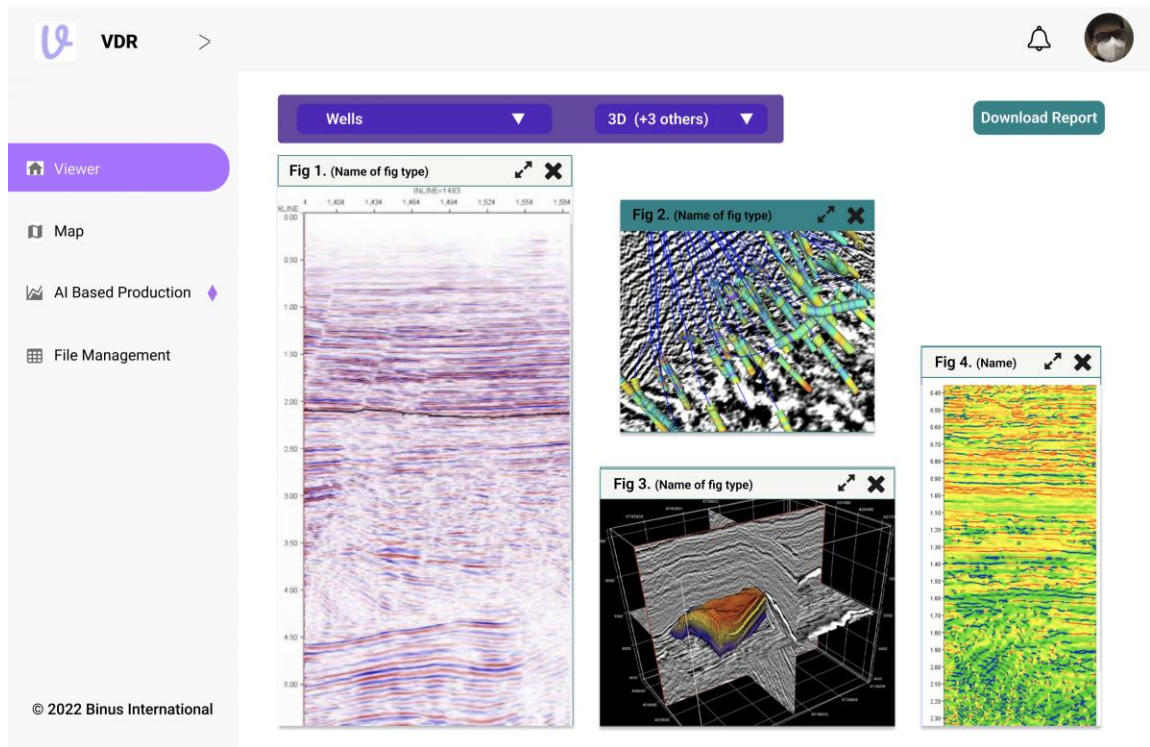


Figure 4.1 Final Design Home / Landing Page

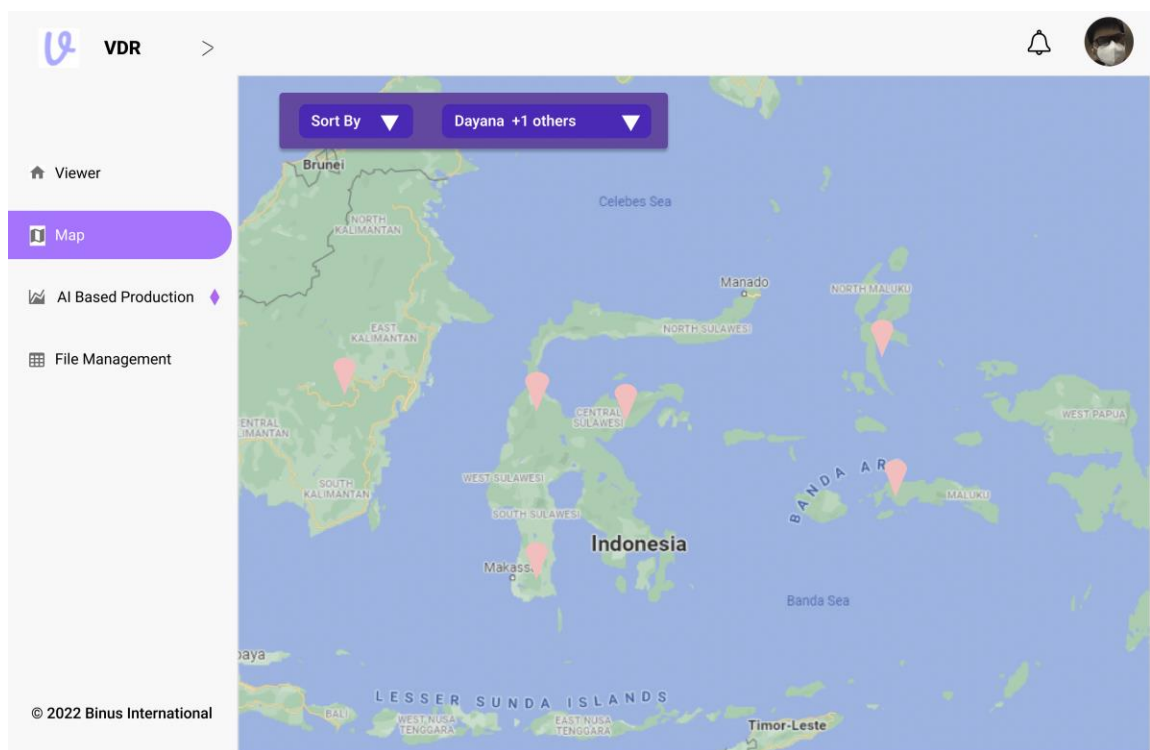


Figure 4.2 Final Design Map Page

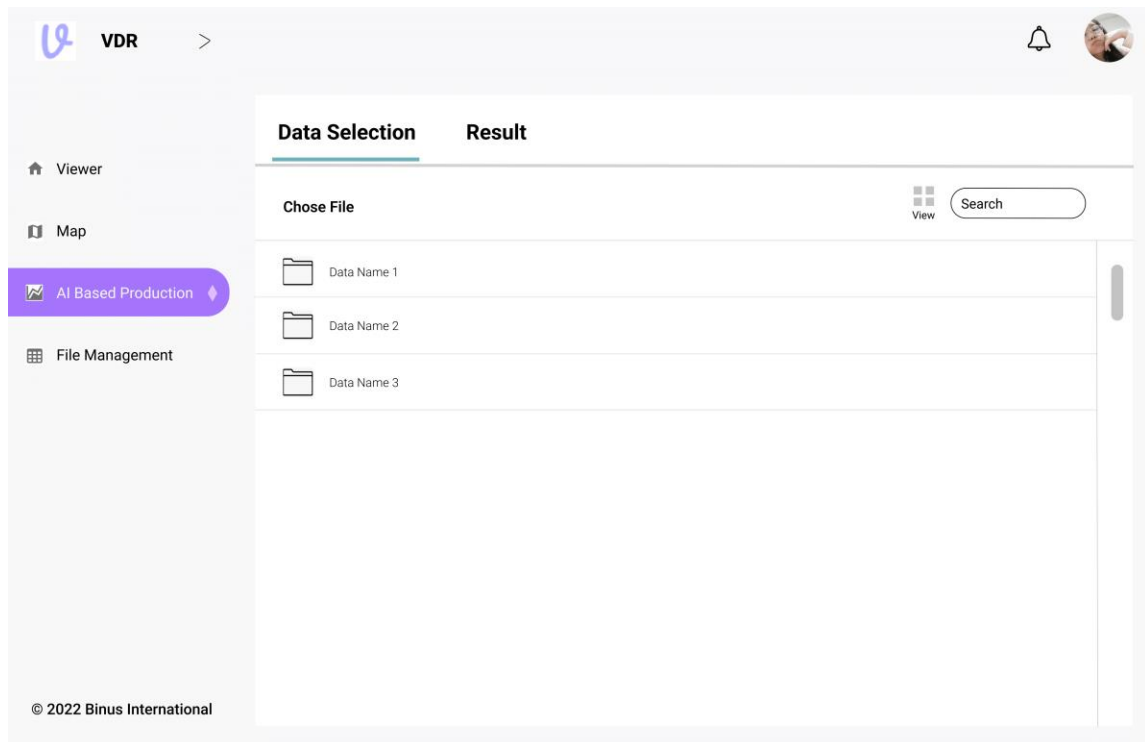


Figure 4.3 Final Design Production Page



Figure 4.4 Final Design File Management Page

ADMIN

+ New Client

List

New Client

Search

User Id	Full Name	Phone Number	Email	License Valid From	License Valid To	Client Type	Action
1	John Doe	+62884857388	John_doe@mail.com	15/02/2022	15/02/2022	Premium	
2	Jane Doe	+62884857388	Jane_doe@mail.com	15/02/2022	15/02/2022	Regular	
3	John doe	+62884857388	John_doe@mail.com	15/02/2022	15/02/2022	Premium	
4	Jane doe	+62884857388	Jane_doe@mail.com	15/02/2022	15/02/2022	Regular	
5	John doe	+62884857388	John_doe@mail.com	15/02/2022	15/02/2022	Regular	
6	Jane doe	+62884857388	Jane_doe@mail.com	15/02/2022	15/02/2022	Regular	
7	John doe	+62884857388	John_doe@mail.com	15/02/2022	15/02/2022	Premium	
8	Jane doe	+62884857388	Jane_doe@mail.com	15/02/2022	15/02/2022	Premium	
9	John doe	+62884857388	John_doe@mail.com	15/02/2022	15/02/2022	Regular	
10	Jane doe	+62884857388	Jane_doe@mail.com	15/02/2022	15/02/2022	Premium	

© 2022 Binus International

Total 50 show 10 records per page

←

1

2

3

4

5

→

Figure 4.5 Final Design List of Clients

ADMIN

Register New Client

List

New Client

First Name

Last Name

Email Address

Phone Number

Company Name

Liscence valid from

Liscence valid to

Type

☐ Premium
 ☐ Regular

Password

Discard

Save Client

© 2022 Binus International

Figure 4.6 Final Design Registering New Client

4.2 Routing

Vue route

4.3 State

vuex

4.4 Communication with Identity Provider

Protocol OpenID

4.5 Communication with Backend

use axios

CHAPTER 5

CONCLUSION

REFERENCES

- [1] M. S. Vassiliou, Historical Dictionary of the Petroleum Industry, Rowman & Littlefield, 2018.
- [2] W. Faisol, S. Indriastuti and A. Trihartono, "Indonesia and OPEC: Why does Indonesia maintain its distance?," *IOP Conference Series: Earth and Environmental Science*, vol. 485, p. 012010, 2020.
- [3] I. S. Chandranegara. and Z. A. Hoesein, "Policy concept and designs of oil and gas governance in Indonesia's oil companies," *International Journal of Energy Economics and Policy*, vol. 9(3), pp. 121-127, 2019.
- [4] "Intviewer - Fast Geoscience Visualization, Analysis & QC," INT, 02 August 2021. [Online]. Available: <https://www.int.com/products/intviewer/>.
- [5] "Lynx Information Systems," Licence Pricing - Lynx Information Systems, [Online]. Available: <http://www.lynxinfo.co.uk/download-pricing.html>.
- [6] Schlumberger, ProSource Front Office User Guide, Texas, 2013.
- [7] "ProSource front office," ProSource, [Online]. Available: <https://www.software.slb.com/products/prosource/prosource-front-office#sectionFullWidthTable>.

- [8] G. Gurung, R. Shah and D. P. Jaiswal, "Software development life cycle models-A comparative study," *International Journal of Scientific Research in Computer Science, Engineering and Information*, pp. 30-37, 2020.
- [9] N. S. Yadav, V. Goar and M. Kuri, "AGILE METHODOLOGY -A PERFECT SDLC MODEL WITH SOME IMPROVEMENTS," *Journal of Critical Reviews*, vol. 7, no. 19, pp. 2511-2514, August 2020 .
- [10] R. Sherman, "Project Management," *Business Intelligence Guidebook*, pp. 449-492, 2015.
- [11] "Manifesto for Agile Software Development," Agile Manifesto, 2001. [Online]. Available: <https://agilemanifesto.org/>.
- [12] P. Adi, "Scrum Method Implementation in a Software Development Project Management," *International Journal of Advanced Computer Science and Applications*, vol. 6, no. 9, September 2015.
- [13] S. Herbold, U. Bünting, J. Grabowski and S. Waack, "Deployable Capture/Replay Supported by Internal Messages," *Advances in Computers*, vol. 85, pp. 327-367, 2012.
- [14] F. Lonetti and E. Marchetti, "Emerging Software Testing," *Advances in Computer*, vol. 108, pp. 91-143, 2018.
- [15] C. Eaton, "Advances in Web Testing," *Advances in Computer*, vol. 75, pp. 281-306.
- [16] Z. Wei, "Research on the application of Open source software in Digital Library," *Procedia Engineering*, vol. 15, pp. 1662-1667, 2011.

- [17] Q. Jiang., J. Qin. and L. Kang, "A literature review for Open Source Software Studies," *Lecture Notes in Computer Science*, pp. 699-707, 2015.
- [18] M. Nelson, R. Sen and C. Subramaniam, "Understanding open source software: A research classification framework," *Communications of the Association for Information Systems*, vol. 17, pp. 266-287, February 2006.
- [19] E. N. Hahn, "An Overview of Open-Source Software Licenses and the Value of Open-Source Software to Public Health Initiatives," *JOHNS HOPKINS APL TECHNICAL DIGEST*, vol. 32(4), 2014.
- [20] M. Jazayeri, "Some Trends in Web Application Development," *Future of Software Engineering (FOSE '07)*, pp. 199-213, 2007.
- [21] J. Manhas, "Comparative Study of Cross Browser Compatibility as Design Issue in Various Websites," *BVICAM's International Journal of Information Technology (BIJIT)*, vol. 07, no. 1, pp. 815-820, January 2015.
- [22] L. N. Sabaren, M. N. Mascheroni, C. L. Greiner and E. Irrazábal, "A Systematic Literature Review in Cross-browser Testing," *Journal of Computer Science and Technology*, vol. 18, no. 1, 2018.
- [23] Y. Xing, J. Huang and Y. Lai, "Research and analysis of the front-end frameworks and libraries in E-Business Development," *Proceedings of the 2019 11th International Conference on Computer and Automation Engineering - ICCAE 2019*, 2019.

- [24] N. Li and B. Zhang, "The research on Single Page Application Front-end development based on Vue," *Journal of Physics: Conference Series*, vol. 1883, 2021.
- [25] N.-A. Sireteanu and D. Homocianu, "Front-end frameworks for development of spa and MPA web application," *SSRN Electronic Journal*, December 2021.
- [26] V. Hutagikar and V. Hegde, "Analysis of Front-end Frameworks for Web Applications," *International Research Journal of Engineering and Technology (IRJET)*, vol. 07, no. 04, April 2020.
- [27] "Getting started - vue.js," Vue.js, [Online]. Available: <https://012.vuejs.org/guide/>.
- [28] R. Hartson and P. S. Pyala, "Introduction," in *The UX Book*, 2012, pp. 1-46.
- [29] J. M. Carroll, "CHAPTER 1 - Introduction: Toward a Multidisciplinary Science of Human-Computer Interaction," in *HCI Models, Theories, and Frameworks*, 2003, pp. 1-9.
- [30] MacKenzie and I. Scott, "Chapter 1 - Historical Context," in *Human-computer Interaction*, 2013, pp. 1-26.
- [31] S. R. Venna, R. N. Gottumukkala and V. V. Raghavan, "Chapter 3 - Visual Analytic Decision-Making Environments for Large-Scale Time-Evolving Graphs," in *Handbook of Statistics*, vol. 35, 2016, pp. 81-115.
- [32] J. Wang, "From self-efficacy to human-computer interaction design," *Journal of Physics: Conference Series*, vol. 1168, no. 3, February 2019.

- [33] B. Shneiderman, "The Eight Golden Rules of Interface Design," University of Maryland, [Online]. Available:
<https://www.cs.umd.edu/users/ben/goldenrules.html>.
- [34] Google, "Material Design," [Online]. Available: <https://material.io/>.
- [35] "Material Components," [Online]. Available: <https://github.com/material-components/material-components>.
- [36] B. Harrison, "The Data Room," *Developments in Petroleum Science*, vol. 69, pp. 21-26, 1 October 2020.
- [37] "Secure, Remote Access to Field Datasets Enables Potential Investors to Complete Asset Evaluations," Schlumberger, [Online]. Available:
<https://www.slb.com/resource-library/case-study/dss/delfi-virtual-data-room-generic-asia-pacific-cs>.
- [38] Y. Waykar, "A Study of Importance of UML diagrams: With Special Reference to Very Large-sized Projects," in *International Conference on Reinventing Thinking beyond boundaries to Excel*, Faridabad, 2013.
- [39] H. Koc, A. M. Erdoğan, Y. Barjakly and S. Peker, "UML Diagrams in Software Engineering Research: A Systematic Literature Review," *Proceedings*, vol. 74, no. 13, March 2021.
- [40] "What is a Flowchart," Lucid, [Online]. Available:
<https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial>.
- [41] Vue.js, "Vue.js," Vue.js, [Online]. Available:
<https://v2.vuejs.org/v2/guide/testing.html>.

APPENDICES

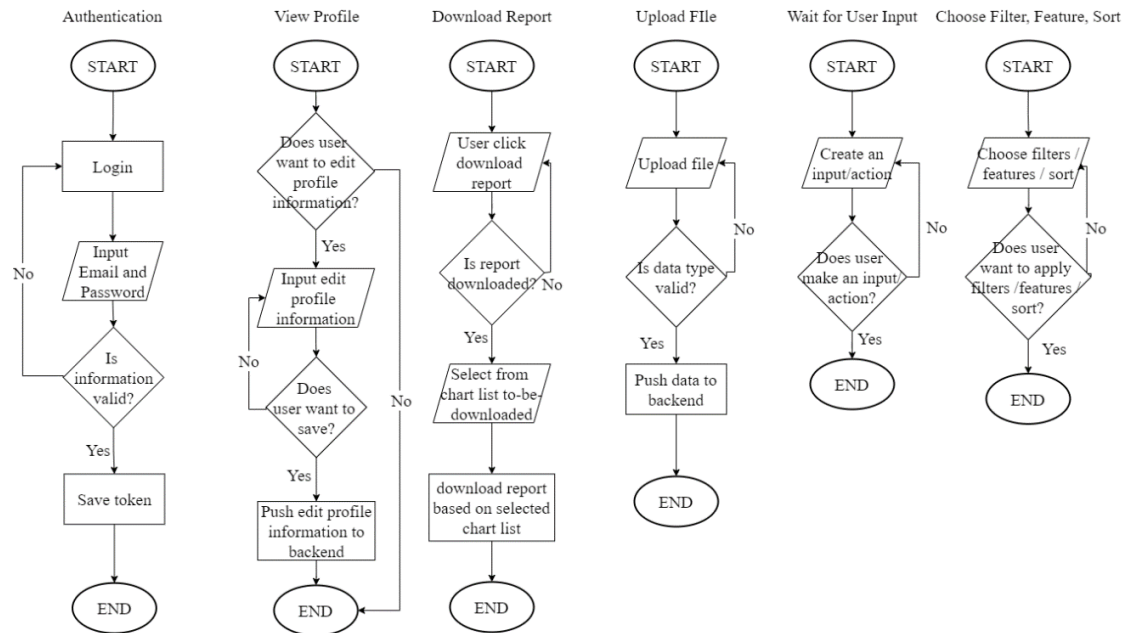


Figure A. 1 Predefined Processors part 1

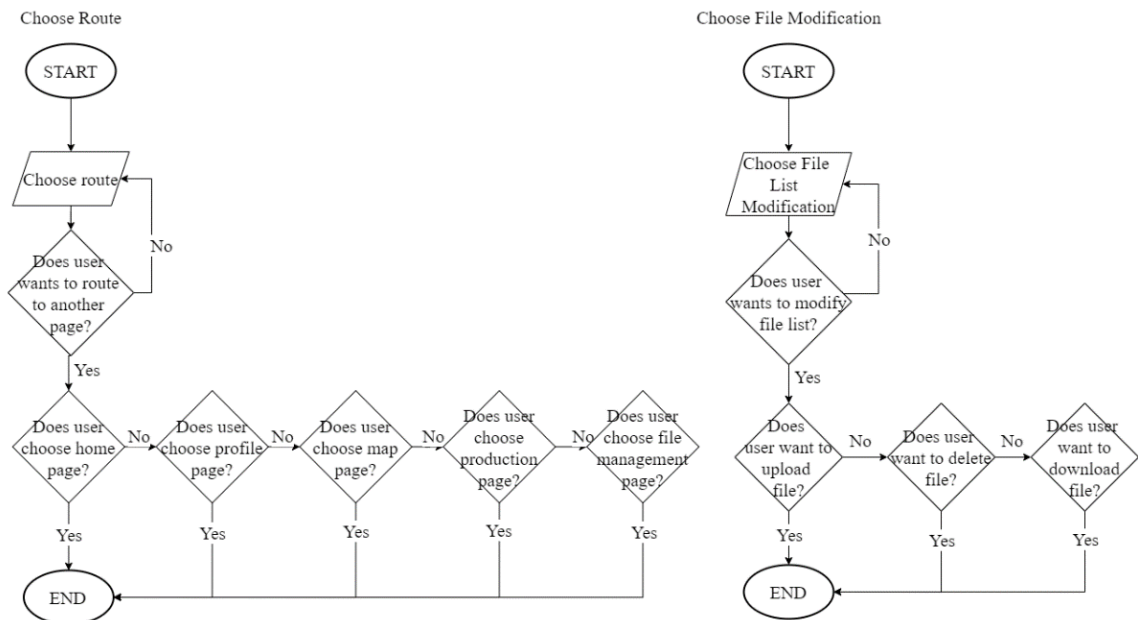


Figure A. 2 Predefined Processors part 2

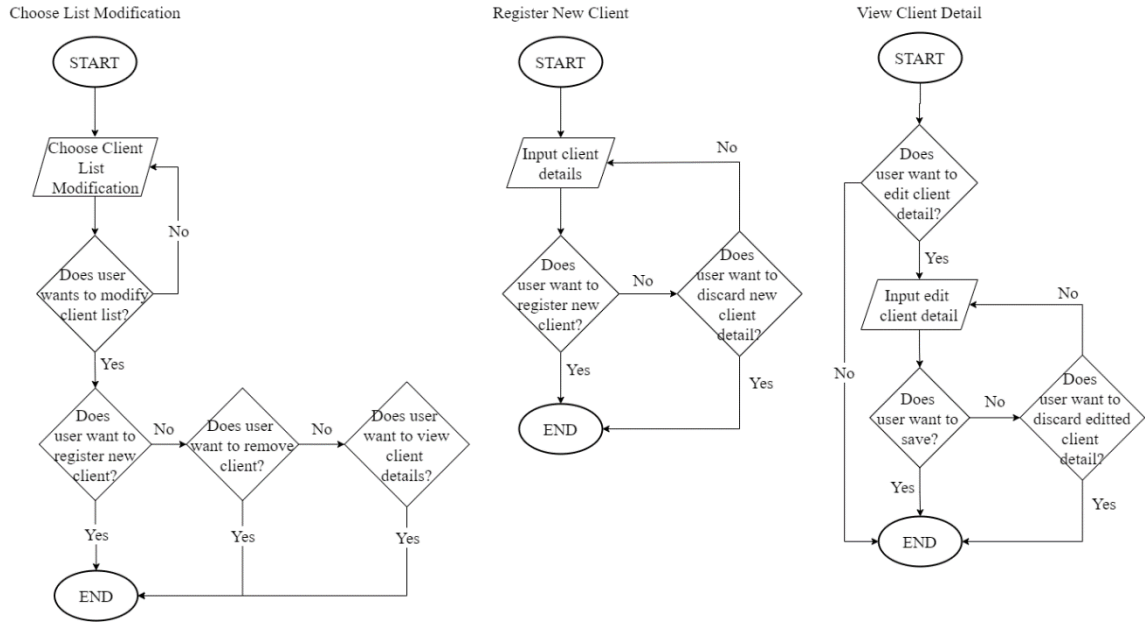


Figure A. 3 Predefined Processors part 3

LOGIN

email

password

LOGIN

VDR PRODUCTION

Home
Log
Documents
Profile

Profile [Edit Profile](#)

First Name	John	Last Name	JOE
Email Address	joe@vdr.com		
Phone Number	+1234 567 8901		
Company Name	ABC Company		
Created with team	06/10/2021	Document id	06/10/2021
IP	192.168.1.1		
Passport	123456		

© 2022 Virus International

Figure A. 4 Sketch Design (a) Login Page (b) View Client Detail/Profile

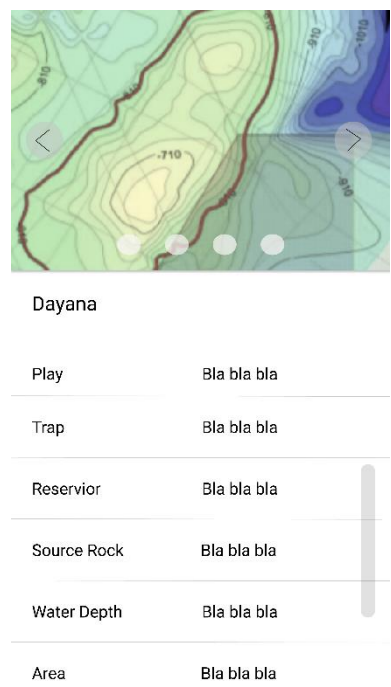


Figure A. 5 Sketch Design Map Overlay Detail

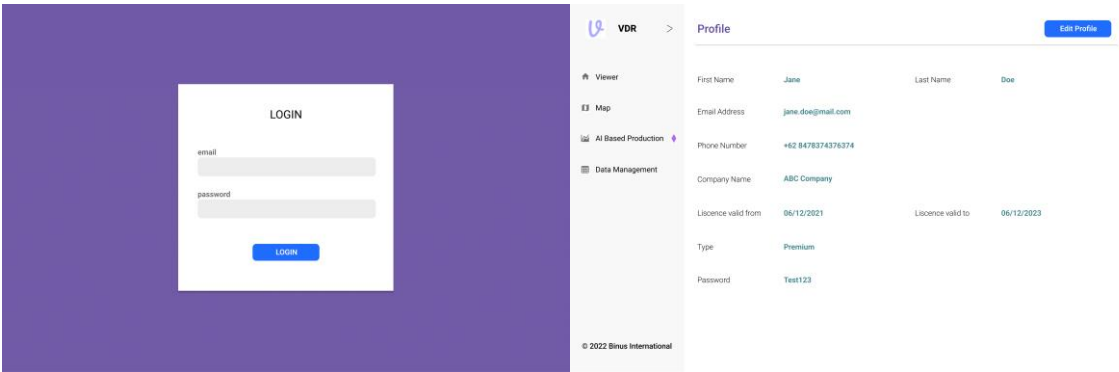


Figure A. 6 Final Design (a) Login Page (b) View Client Profile Detail

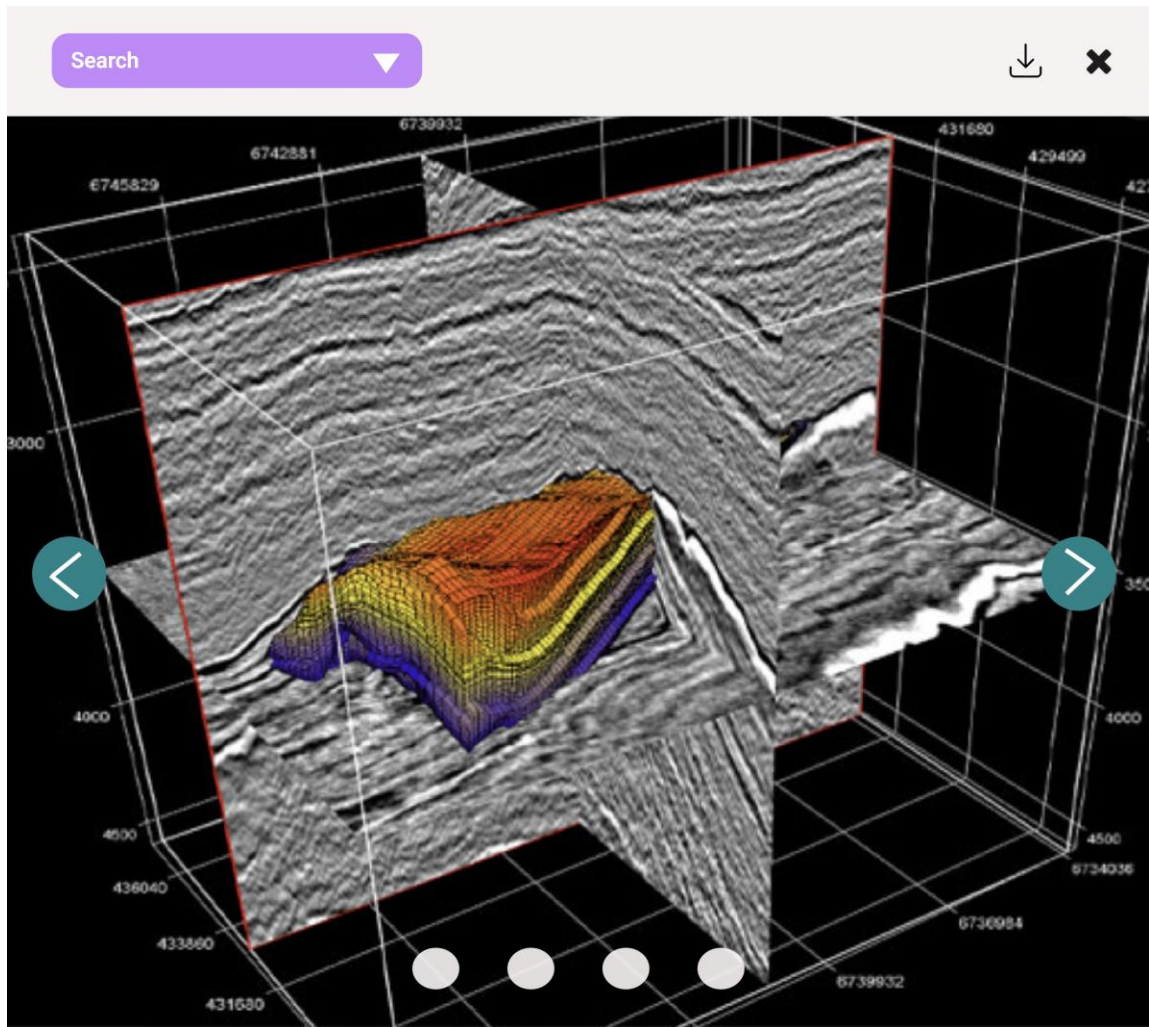


Figure A. 7 Final Design Map Overlay Detail

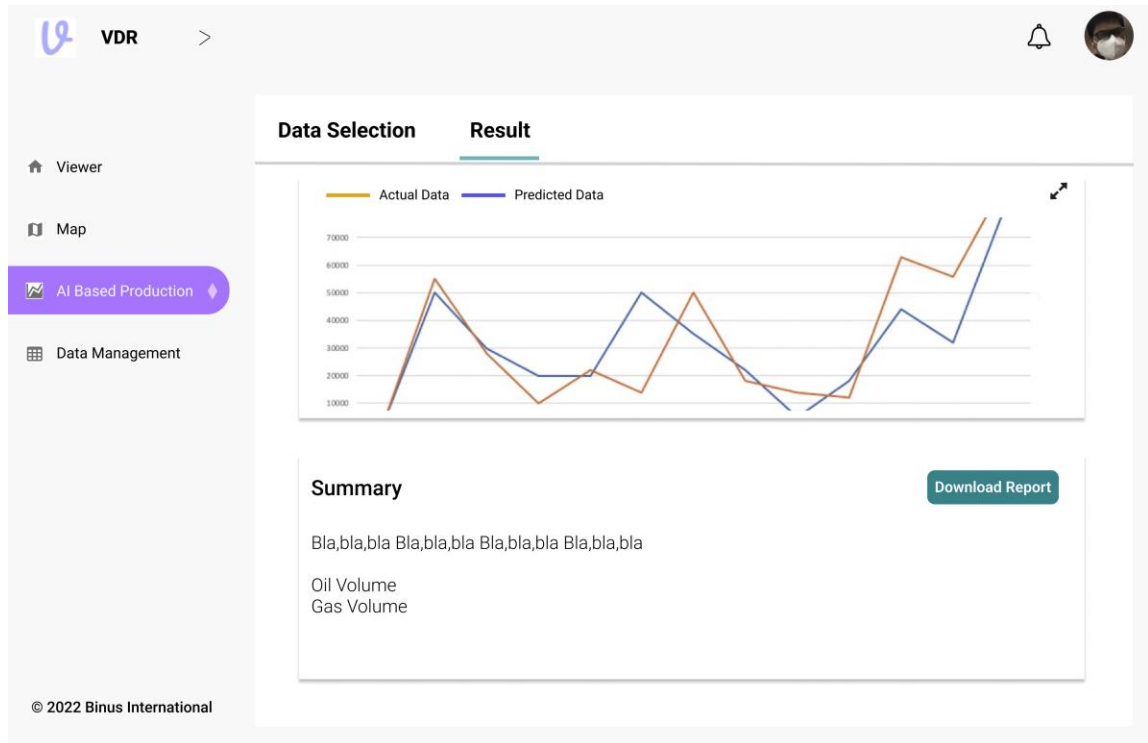


Figure A. 8 Final Design Production Page Result Page