My Dissertation Title

My Name

May 12, 2023

Contents

List of Figures

List of Tables

General Introduction

The rapid advancements in information technology have transformed the way organizations operate and manage their processes. One such domain that has witnessed significant growth is the development of web applications for various purposes. In particular, web applications have become essential tools for improving the efficiency of service delivery and reducing operational costs for many organizations, including government institutions.

Problematic and Objectives

The National Social Security Fund (CNAS : Caisse Nationale des Assurances Sociales des Travailleurs Salariés) in Algeria is one such organization that can benefit from the adoption of web-based solutions as it is responsible for providing a range of social security services to Algerian citizens, including health insurance, retirement benefits, and unemployment benefits. The organization serves a large number of people and has several applications that we will talk about in upcoming chapters.

Although the current system of managing queues at CNAS helps with the organization and the process of the work, it has proved to be inefficient and time-consuming for both the employees and the beneficiaries of social security, and it has been struggling to keep up with the increasing demand. For instance, imagine coming all the way to CNAS and having to wait for an hour just to get information about a document, knowing that it could be obtained in seconds through a web-based solution. This highlights the inefficiency of the current system, which is not only time-consuming but also inconvenient for the beneficiaries who have to take time off from work to visit CNAS. A web-based solution that streamlines the appointment management process will save time and effort for both the employees and the beneficiaries and will enhance the overall efficiency of the services provided by CNAS.

Therefore, the objective of this project is to create a web application that streamlines the appointment management process to improve the overall efficiency of the services provided by CNAS. The proposed web application has a key feature that enables users to choose the service and the task they want to do at CNAS before booking an appointment. At the beginning of the user's journey through the application, they are prompted to complete a questionnaire that helps generate a personalized Checklist of the necessary documents and steps they need to complete in order to achieve their goal. This questionnaire feature streamlines the process for the user by providing clear guidance and ensuring that no important documents or steps are missed. This feature ensures that the user is directed to the appropriate service desk for their needs, reducing the time wasted on unnecessary visits and allowing users to access all the necessary information online and plan their appointments accordingly.

Moreover, this web application will include a range of features designed to enhance the appointment management process, including the ability to track the status of appointments and documents, a reminder and notification system, customization of appointments and schedules, an authentication and security system, and multilingual support.

Dissertation Plan

This dissertation is structured into three main chapters. Chapter one provides an in-depth analysis of the state of the art in virtual counters and appointment management systems, highlighting the existing solutions, challenges, and limitations. Chapter two focuses on the conception phase, where the requirements for the virtual counter system are identified, user needs are analyzed, and the system design and architecture are described. Chapter three delves into the implementation phase, discussing the practical aspects of developing the virtual counter system, including GitHub setup and configuration, development methodologies, and integration of relevant technologies. The dissertation concludes with a comprehensive evaluation of the implemented system, future recommendations for enhancements, and the significance of the findings. Throughout the dissertation, a thorough examination of the relevant literature and appropriate methodologies is conducted to ensure the research objectives are met and to provide valuable insights into the field of virtual counters and appointment management systems.

Chapter 1

State of The Art

1.1 Virtual Counters

Virtual Counters, or Guichets Virtuels in French, are online platforms that allow users to access services remotely without having to physically visit a location. They are designed to facilitate the interaction between users and service providers in a user-friendly, efficient and secure manner. The rise of digital technology has led to the development of various types of virtual counters, each with its own features and benefits.

1.1.1 Types of Virtual Counters

There are various types of virtual counters, such as:

- **Web-based virtual counters:** These virtual counters are accessible through a web browser, and they allow users to access various online services offered by service providers.
- **Mobile-based virtual counters:** These virtual counters are accessible through mobile devices such as smartphones and tablets, and they offer users the convenience of accessing services on the go.
- **Kiosk-based virtual counters:** These virtual counters are installed in designated locations and allow users to access various services through self-service kiosks.
- **Chat-based virtual counters:** These virtual counters use instant messaging applications to facilitate communication between users and service providers, allowing users to access services through a chatbot or live chat.

1.1.2 Examples of Virtual Counters

Virtual counters have become increasingly popular in Algeria, and several organizations have adopted them to improve their services. Some examples of virtual counters in Algeria include:

- **ElHanna:** The Caisse Nationale de l'Assurance Maladie (CNAS) in Algeria has created an application called "El Hanna" that allows its members to access various services related to their health insurance coverage, such as checking their eligibility for medical procedures, viewing their medical history

- **BaridiMob:** Algérie Poste has developed a virtual counter that allows customers to access their banking services online, such as transferring funds and paying bills.
- **Sonelgaz:** Sonelgaz has developed a virtual counter that allows customers to access their energy bills and make payments online.
- **Algerie Telecom E-Paiement:** Algerie Telecom E-Paiement is a mobile application-based electronic payment service provided by Algerie Telecom. It offers customers a secure and convenient platform to make online payments, pay bills, recharge mobile credit, and make purchases using their smartphones. By embracing mobile technology, Algerie Telecom E-Paiement enhances the accessibility and convenience of electronic payments in Algeria.

Virtual counters have also been implemented in other countries, such as:

- **eVisa:** The eVisa platform allows travelers to apply for visas online, reducing the need to physically visit an embassy or consulate.
- **eCNI:** The eCNI platform in France allows citizens to apply for their national identity cards online, reducing the need to visit a physical office.

In the next section, we will explore the benefits of virtual counters and their impact on the user experience.

1.1.3 Benefits of Virtual Counters

Virtual counters offer several benefits for both users and service providers. Some of the key benefits include:

- **Convenience:** Virtual counters can be accessed from anywhere with an internet connection, making it more convenient for people to access services without having to physically go to a government office.
- **Time-saving:** Virtual counters eliminate the need for users to physically visit a service center, saving them time and effort. Users can complete their transactions from the comfort of their own homes or offices, without having to wait in long lines or take time off work.
- **Accessibility:** Virtual counters provide users with greater accessibility to services. They can access services from anywhere, at any time, as long as they have an internet connection. This is particularly beneficial for people with disabilities or those who live in remote areas and have limited access to physical service centers.
- **Efficiency:** Virtual counters streamline the service delivery process by reducing paperwork, eliminating redundancies, and increasing transparency. This allows service providers to process transactions more efficiently and with greater accuracy.
- **Cost-effective:** Virtual counters are typically more cost-effective for service providers than physical service centers. They require less physical infrastructure, fewer staff, and have lower operating costs. This can help service providers reduce costs and improve their bottom line.

1.1.4 Challenges and Limitations

Despite the benefits of virtual counters, there are also some challenges and limitations to consider. These include:

- **Access and Connectivity**

  One of the biggest challenges of virtual counters is ensuring that they are accessible to everyone, regardless of their location or technical ability. This requires reliable internet connectivity, as well as user-friendly interfaces and support for multiple languages.

- **Security and Privacy**

  Virtual counters also raise concerns about security and privacy. Users may be hesitant to share sensitive personal information online, and there is always the risk of data breaches or cyber attacks.

- **Digital Divide**

  Another limitation of virtual counters is the digital divide, which refers to the gap between those who have access to digital technologies and those who do not. This can be a particular challenge in developing countries or among low-income populations.

- **Technical Issues**

  Finally, virtual counters may also face technical issues such as server downtime, software bugs, or compatibility problems with different devices and platforms. These can all affect the user experience and the efficiency of the service.

Despite these challenges, virtual counters have the potential to revolutionize the way we access public services and interact with government agencies. By addressing these limitations, we can ensure that virtual counters are accessible, secure, and efficient for everyone.

1.2 Introduction to CNAS Organization

1.2.1 Definition of CNAS organization

The CNAS (Caisse Nationale des Assurances Sociales) is a public institution with specific management under Article 49 of Law No. 88-01 of January 12, 1988. It has legal personality and financial autonomy and is considered a merchant in its relations with third parties. The CNAS is responsible for managing social insurance benefits (illness, maternity, disability, and death), as well as occupational accidents and diseases (AO/D), and family allowances on behalf of the state. It also manages the collection, control, and litigation of contributions for financing benefits, as well as the management of the litigation related to the collection of subscriptions for financing rendered.

The CNAS assigns a national registration number to insured persons and employers and contributes to promoting the policy of prevention of AO/D and managing the AO/D prevention fund. It also manages benefits for beneficiaries of bilateral social security agreements, carries out medical control of beneficiaries, and undertakes actions to provide workers and their dependents with collective benefits in the form of health and social achievements. The CNAS also manages the aid and relief fund and concludes agreements with healthcare providers while ensuring the information of beneficiaries and employers.

The CNAS provides benefits to salaried workers, apprentices, job seekers, students, trainees in vocational training, disabled persons, veterans, social security beneficiaries (pensioners and annuitants), and beneficiaries of the lump sum solidarity allowance (sick, elderly and inactive persons). Dependents, including the spouse, minor children, unmarried inactive daughters, and dependent ascendants, are also eligible for benefits.

The CNAS covers healthcare and medication costs at 80%, and in some cases 100% (particularly for chronic diseases). Compensation for sick leave is 50% of the salary for the first 15 days and is increased to 100% of the salary beyond the 16th day, with a maximum duration of three years. Maternity benefits are fully covered, and working women are entitled to a 98-day maternity leave. The minimum amount of invalidity pensions is equal to 75% of the guaranteed minimum wage. In the event of the insured person's death, a death benefit is paid to his or her dependents. Occupational risks are covered 100% for healthcare and sick leave, and annuities are paid in the event of bodily harm or death resulting from occupational accidents or diseases. [1]

1.2.2 Organization of CNAS

CNAS is managed by a Board of Directors and is under the supervision of the Minister of Labor, Employment and Social Security. Its headquarters is located in Algiers (BEN AKNOUN), and it has national jurisdiction with both central and local services.[2]

To fulfill its missions, CNAS has:

Structure	**Number**
General Directorate	1
Provincial agencies(including 2 in Algiers)	49
Payment structures	826
Payment centers	356
Payment branches	401
Local correspondences	69
Specialized clinics (pediatric heart surgery, orthopedics and rehabilitation, ENT, dental)	4
Regionval centers for medical imaging	4
Diagnostic and treatment centers	35
Pharmaceutical offices	55
Nurseries and kindergartens	30
Printing house in Constantine	1
Family social center in Ben Aknoun	1

Table 1.1: CNAS Structures

Source: CNAS website (`http://www.cnas.dz/`)

[1] CNAS. (n.d.). Presentation of CNAS. Retrieved from `https://www.cnas.dz/`.
[2] CNAS. (n.d.). Presentation of CNAS. Retrieved from `https://www.cnas.dz/`.

1.2.3 CNAS Organigram

the CNAS organigram is made up of various departments, subdivisions, and services that work together to manage CNAS operations and deliver services to its beneficiaries.

Director: This is the topmost position in the CNAS hierarchy and is responsible for overseeing all CNAS operations.

Division of Benefits: This department is responsible for managing CNAS' various benefit programs, including health, maternity, and disability benefits.

Division of Administration and General Resources: This department is responsible for managing CNAS' administrative operations, such as human resources, procurement, and general resource management.

Data Processing Center: This department is responsible for managing CNAS' information technology systems and infrastructure.

Division of Recovery and Finance: This department is responsible for managing CNAS' financial operations, including revenue collection and disbursement.

Medical Control Division: This department is responsible for monitoring and controlling the quality of medical services provided by CNAS.

Contracting Service: This department is responsible for managing CNAS' contracts with healthcare providers.

Personnel Division: This department is responsible for managing CNAS' human resources operations, including recruitment, training, and personnel records management.

Statistics, Archives and Documentation Service: This department is responsible for managing CNAS' data and document management systems.

Recovery Division: This department is responsible for collecting outstanding debts owed to CNAS.

Medical Control Service: This department is responsible for conducting medical audits and reviewing medical claims.

Pharmacy Service: This department is responsible for managing CNAS' pharmacy operations, including the provision of pharmaceutical services to CNAS beneficiaries.

Conventions Service: This department is responsible for managing CNAS' relationships with healthcare providers, including contract negotiations and payment processing.

General Resources Division: This department is responsible for managing CNAS' facilities, equipment, and other general resources.

Internal Control Unit: This department is responsible for ensuring compliance with CNAS policies and procedures.

Prevention Service: This department is responsible for promoting public health and disease prevention.

Contentious Service: This department is responsible for managing CNAS' legal affairs, including dispute resolution and litigation.

Payment Structures: This department is responsible for managing CNAS' payment processing systems.

Realization Service: This department is responsible for managing CNAS' development and construction projects.

C.I.W.Q: This is a service that is responsible for managing CNAS quality control.

CHIFA Service: This department is responsible for managing CNAS' maternal and child health services.

Affiliation and Transfer Service: This department is responsible for managing CNAS' beneficiary registration and transfer operations.

CLRQP: This department is responsible for managing CNAS' social and family benefits.

Accounting Service: This department is responsible for managing CNAS' accounting operations.

Finance Service: This department is responsible for managing CNAS' finance operations.

Security Service: This department is responsible for managing CNAS' security operations, including physical security and cybersecurity.

Employer Control Service: This department is responsible for monitoring employers' compliance with CNAS regulations.

High-Risk Service: This department is responsible for managing CNAS' high-risk cases.

Legal Affairs Service: This department is responsible for providing legal advice and support to CNAS.

Here is the CNAS organigram :

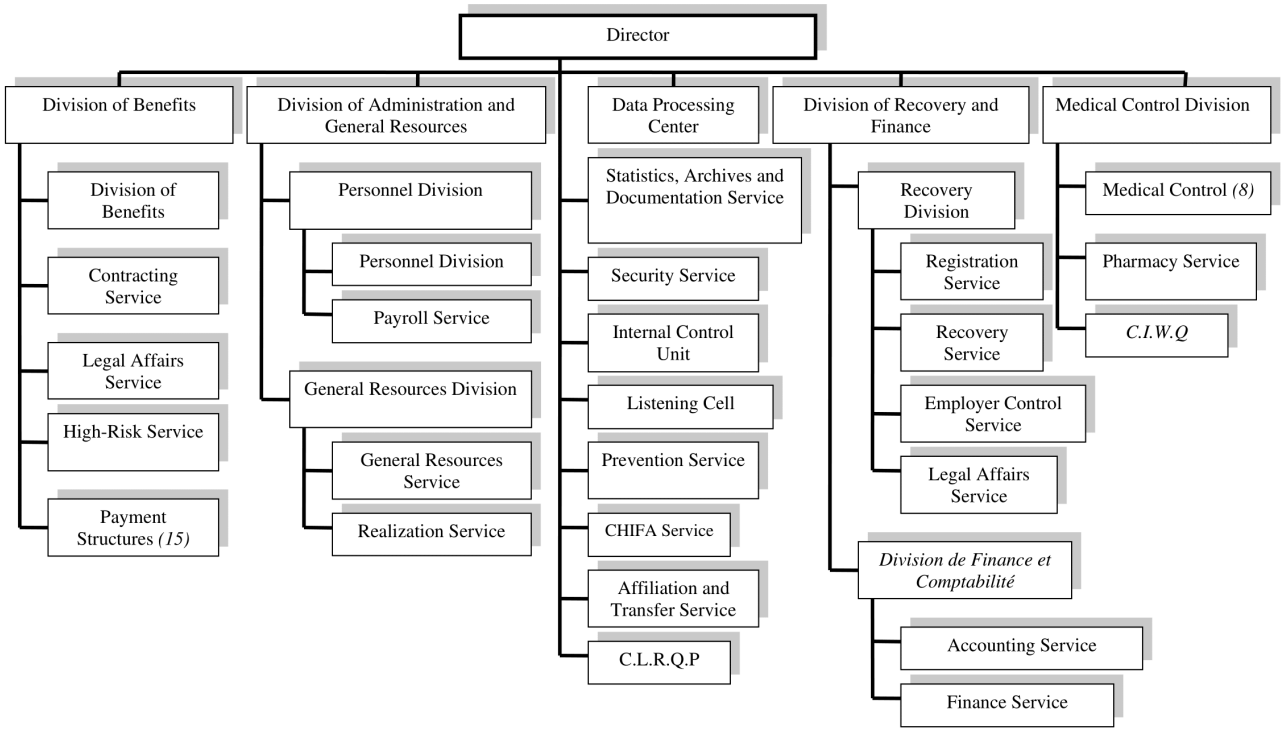

Figure 1.1: Organigram of CNAS

1.2.4 CNAS Services

CNAS provides a range of services related to social security and healthcare to the Algerian population. These services include:

- **Healthcare services:** CNAS operates its own specialized clinics and medical facilities, including four specialized clinics for cardiac surgery, orthopedics and rehabilitation, otorhinolaryngology, and dental care. It also runs 35 diagnostic and treatment centers, 55 pharmacies, and four regional medical imaging centers.
- **Social security services:** CNAS provides social security services to its members and their families, including health insurance, maternity leave benefits, disability benefits, and retirement pensions. It also offers services related to workplace safety and injury compensation.
- **Family services:** CNAS operates 30 nurseries and childcare centers to support working parents.
- **Payment services:** CNAS manages a network of payment centers and local correspondents to ensure the timely payment of social security benefits to its members.

1.2.5 Importance of CNAS services for the Algerian society

These services are essential for the Algerian society, as they help provide access to healthcare and social security benefits to millions of people. CNAS's role in ensuring workplace safety and providing compensation for work-related injuries is also crucial in protecting the rights and wellbeing of workers across Algeria.

1.3 Literature Review

Virtual counters and appointment management systems have emerged as essential tools in various domains, revolutionizing the way organizations handle customer interactions and streamline service processes. In this literature review, we explore the significance of virtual counters, and we discuss the importance of adopting web-based solutions for appointment management, highlighting the benefits they offer to CNAS and similar organizations.

Virtual counters provide an innovative approach to managing customer queues and optimizing service delivery. They enable users to access services remotely, eliminating the need for physical presence and reducing waiting times. CNAS's El Hanna application serves as a prime example of a virtual counter, enabling users to perform various tasks online, such as checking eligibility, submitting documents, and scheduling appointments. The application has streamlined operations, enhancing user experience, and improving overall efficiency.

Virtual counters offer several advantages over traditional counter-based systems. Firstly, they provide convenience and accessibility by allowing users to access services anytime, anywhere, without the constraints of physical presence. This improves customer satisfaction and reduces the burden on physical infrastructure. Secondly, virtual counters enhance operational efficiency by automating processes, reducing paperwork, and enabling efficient resource allocation. Thirdly, they provide accurate data collection and analysis, facilitating informed decision-making and resource planning. Overall, virtual counters improve service quality, increase efficiency, and enhance user experience.

Appointment management systems complement virtual counters by facilitating the scheduling and organization of appointments. These systems enable users to book appointments online, choose suitable time slots, and receive automated reminders. By implementing an appointment management system, CNAS can streamline the appointment booking process, reducing waiting times, minimizing no-shows, and optimizing resource allocation. This not only improves operational efficiency but also enhances the overall patient experience.

Considering the vast scale of CNAS's operations and the increasing demand for its services, leveraging web-based solutions, including virtual counters and appointment management systems, becomes imperative. These technologies offer CNAS the opportunity to enhance service accessibility, optimize resource utilization, and improve the overall efficiency of its operations. By implementing these solutions, CNAS can reduce administrative burden, improve customer satisfaction, and ensure efficient service delivery to its beneficiaries.

1.3.1 Conclusion

Virtual counters and appointment management systems have revolutionized service delivery in various sectors, offering convenience, efficiency, and improved user experience. CNAS's El Hanna application serves as a successful example of a virtual counter, demonstrating the benefits of such systems. By adopting web-based solutions and integrating appointment management systems, CNAS can further enhance its service delivery, optimize resource utilization, and provide a seamless experience to its beneficiaries. It is evident that these technologies have become necessary tools for CNAS and similar organizations to meet the growing demands of service users in today's digital era.

1.4 Requirements analysis

The requirements analysis phase identified several key features that the virtual counter for CNAS must provide. First, the system should provide a questionnaire for users to fill out, generating a checklist of necessary documents that must be obtained before the appointment. Second, it should allow users to book appointments online and provide them with a ticket number to avoid the need to wait in long queues. Third, it should allow CNAS staff to manage and monitor the appointments, including rescheduling or cancelling them if necessary. Fourth, the system should allow users to view their appointment history and provide feedback on their experience with the virtual counter. Finally, the system should ensure the security and privacy of all user data.

These requirements will be used as a basis for the design, the conception and the implementation of the virtual counter system.

1.5 Conclusion

In this chapter, we have explored the current state of the art related to virtual counters, the organization of CNAS, and the literature review of virtual counters in various contexts. The use of virtual counters has been found to offer several benefits, including improved efficiency, reduced wait times, and increased convenience for users. CNAS, as an Algerian social security institution, provides a range of essential services to the Algerian society, including healthcare, childcare, and employment-related services.

Additionally, we have discussed the organization of CNAS and its numerous structures, such as its 49 Agences de wilaya and 826 structures de paiement.

Finally, we highlighted the importance of CNAS services for the Algerian society, emphasizing the need for modernization and innovation to ensure that these services continue to meet the evolving needs of its users.

Overall, this chapter provides a comprehensive understanding of the state of the art related to virtual counters and CNAS organization which serves as a foundation for the subsequent chapters in this dissertation.

Chapter 2

Conception

2.1 Purpose of the chapter

The purpose of this chapter is to present the conception of a virtual counter system for the Algerian National Social Security Fund (CNAS). This chapter will provide a detailed explanation of the system design and architecture, database design, as well as the different diagrams and models used during the conception phase. The virtual counter system aims to improve the current management system used by CNAS by providing users with a more efficient and user-friendly way to gather necessary information and book appointments.

2.2 Overview of the topics covered

This chapter focuses on the conception of the virtual counter system for CNAS. It includes the analysis and design of the system, from the identification of user requirements to the development of the system architecture and database design. The chapter also includes the presentation of the different diagrams that were created, such as the use case diagram, class diagram, sequence diagram, and flowchart. The aim of this chapter is to provide a comprehensive understanding of the virtual counter system, its components, and its functionalities.

2.3 System design and architecture

The system design and architecture of a virtual counter is a crucial aspect in developing a successful web application. It involves designing the components of the system and specifying how they interact with each other to achieve the desired functionality. In the case of a virtual counter for CNAS, the system design and architecture must take into account the different types of users, such as clients and agents, and the various tasks they need to perform. It must also ensure that the application is secure and reliable, with measures in place to protect user data and prevent unauthorized access. The system design and architecture will involve selecting suitable technologies and frameworks, such as Laravel and VueJs, and designing a database schema to store and retrieve data efficiently. Overall, a well-designed system architecture will contribute to the effectiveness and efficiency of the virtual counter and improve the user experience for both clients and agents.

2.3.1 Description of the overall system architecture

The overall system architecture of the virtual counter for CNAS is designed to be a web-based application with a client-server architecture. The client-side will be a user-friendly interface, developed using Vue.js framework, that allows users to interact with the system and perform different tasks, such as filling in a questionnaire that will generate a checklist of required documents, booking appointments, and checking their status. On the other hand, the server-side of the application will handle all the processing and data storage. It will be developed using the Laravel framework, which is a powerful and reliable PHP web application framework that enables rapid application development with a robust and scalable codebase. The application will also use a MySQL database to store all the necessary data, such as user information, appointment schedules, and queue status. The overall system architecture is designed to be modular and scalable, allowing for easy maintenance and future updates.

2.4 Diagrams illustrating the different components of the system

Diagrams can help to provide a visual representation of the different components and processes involved in the virtual counter system, making it easier to understand and communicate to stakeholders.

The use of UML (Unified Modeling Language) which is a standered Language for visualizing and creating views to illustrate the different parts of a system , presenting us with a various types of diagrams that facilitates the conception phase for the virtual counter and makes it more comprehensive .

2.4.1 Use case diagram

Use case diagram is one of the most used static diagrams in UML , it consist on explaining the different actions preformed by the user and helps understanding the main functions that can be preformed by the system.

When the user is interacting with the system, the virtual counter enables him to consult the various services provided by CNAS without the need to log in.

Additionally, the user can also complete a variety of tasks, such as selecting a service and completing a questionnaire related to that service. The system will then generate a checklist of the documents he will need to submit. The user can stop at printing that checklist or he can move on to booking an appointment which will require him to be authenticated. When an appointment is booked, an appointment ticket, that contains the previous checklist along with some appointment details such as the date and time, the counter number and the name of the employee responsible for treating your concerns, will be available to print.

In the second hand of the virtual counter, both the employee and the supervisor have their own interactions with the system; however, in both their cases, they both need to be logged in order to access the various functionalities of the system. In addition to managing their work flow, both can manage the appointments by treating, rescheduling or canceling them if necessary.

Here is the diagram:

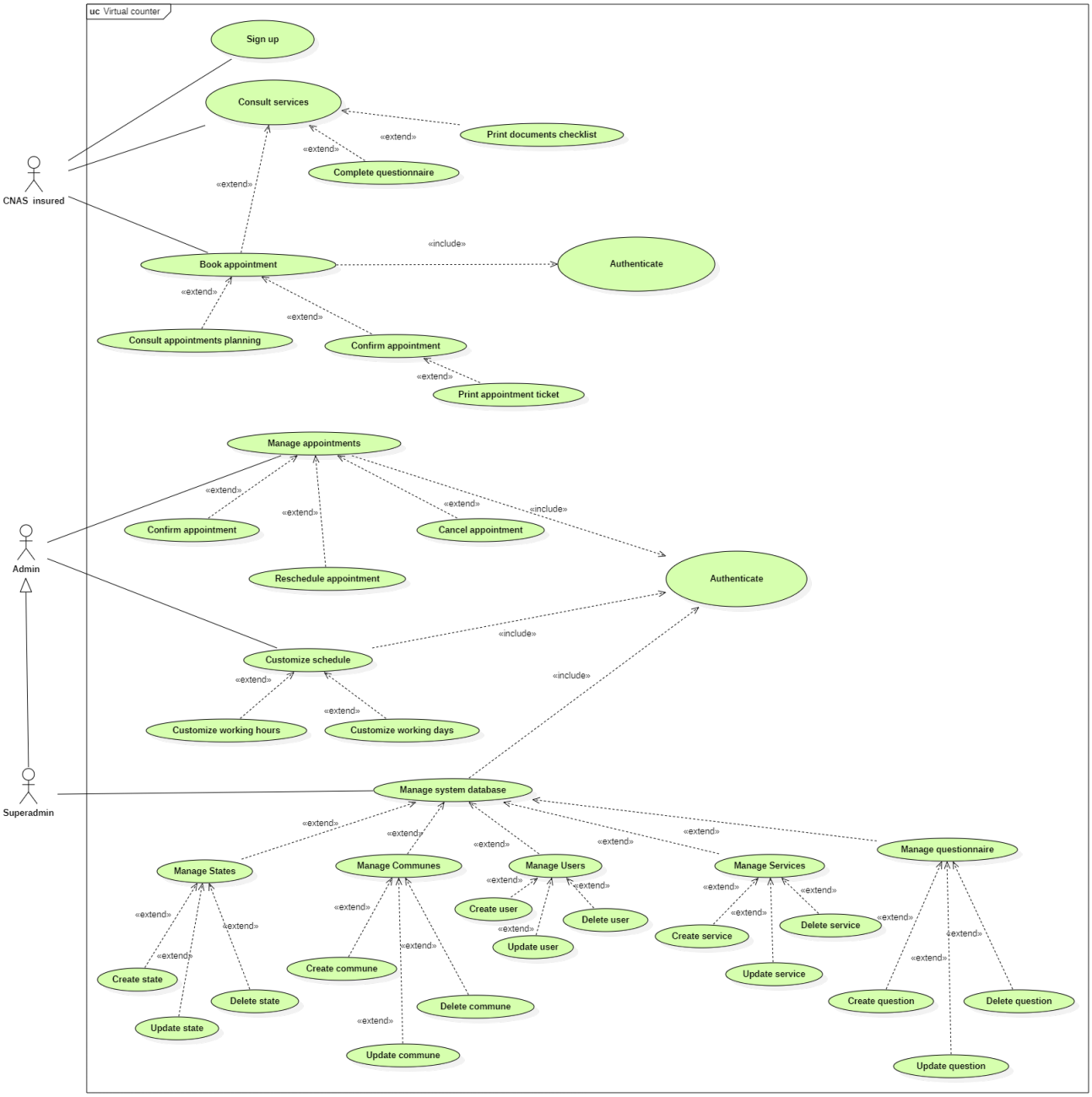

Figure 2.1: Use case diagram

2.4.2 Class diagram

A class diagram is a type of UML diagram that represents the structure of a system by showing the classes, interfaces, and their relationships. It is an important tool for software engineers to design and communicate the architecture of a system.

In this section, we present the class diagram of the CNAS virtual counter system. The diagram illustrates the key components of the system and their relationships, including the classes for managing users, services, appointments, and other relevant data. This diagram provides a visual representation of the system's architecture, which will help in understanding how the virtual counter works and how it can be further developed and maintained.

Here's the class diagram:

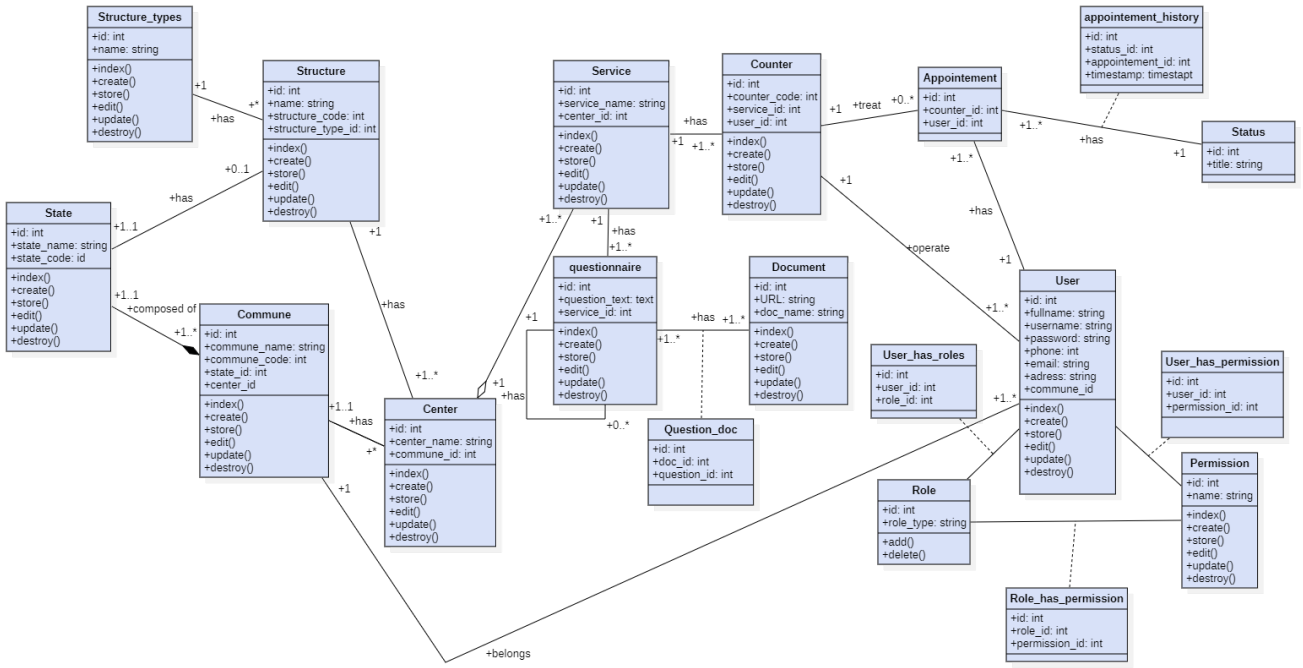

Figure 2.2: Class diagram

2.4.3 Sequence diagram

Sequence diagram in one of the well known dynamic diagrams that allow the overall understanding for the hidden functionalities , and streamlines the developemnt phase .

Here are the different diagrams related to every user in the application as well as the registration sytem .

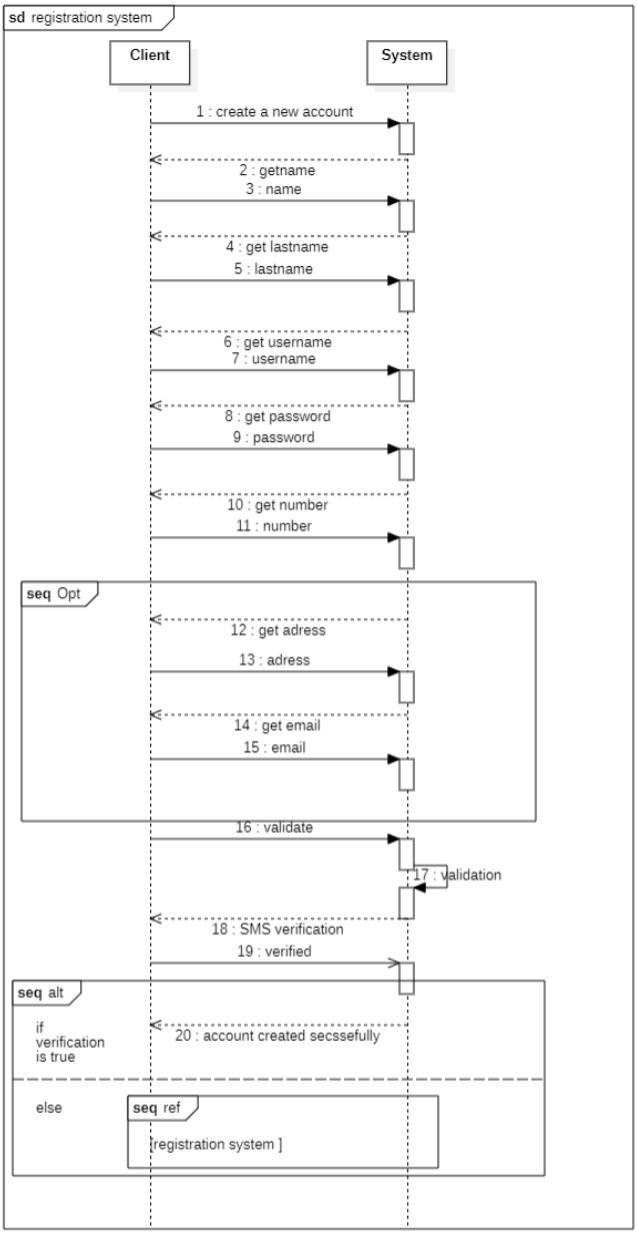

Figure 2.3: Sequence diagram for Registration

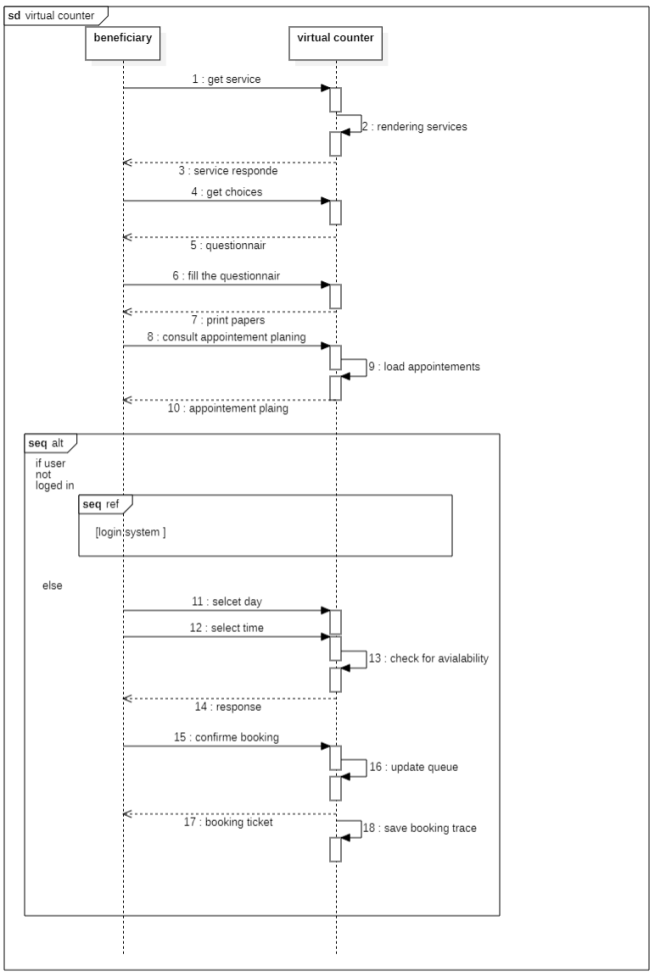

Figure 2.4: Sequence diagram for client interaction

2.4.4 Discussion of the design decisions made

Overall, the decision discussions we had during the process of elaborating the UML diagrams helped us to refine the system's design and functionality and to ensure that it met the requirements and expectations of both the users and the organization. The final UML diagrams illustrate the system's architecture and behavior in a clear and concise manner, and provide a solid foundation for building a robust and efficient virtual counter system for CNAS.

We came to a final conception of the virtual counter on the basis of the notion of eliminating the waiting time in such an effective way and to ensure an efficient system, which can provide by far a great user experience and a sturdy system.

2.5 Database design

At this juncture of conception, our primary focus was on creating a reliable database because a well-designed database is crucial for the efficient functioning of any application, including the virtual counter. We also made sure that accessing the data would be a secure process while also allowing stakeholders to track and store their data effectively.

To dive in more, the virtual counter's database include .... table, we arrived at this effective design after thorough analysis and testing, its optimized for performance, scalability, and ease of maintenance, and it is capable of handling large volumes of data with minimal overhead.

We achieved this by carefully analyzing the data requirements of the CNAS virtual counter, identifying potential bottlenecks and inefficiencies and refining the design when it needed updates until we arrived at the final design that satisfies the performance goals.

2.5.1 Overview of the database schema

//

2.5.2 Explanation of the different tables and their relationships

//

2.5.3 Discussion of the design decisions made

//

Chapter 3

Implementation

3.1 Chapter Overview

In this chapter, we will delve into the comprehensive implementation details of CNAS's virtual counter system. We will explore the utilization of various cutting-edge technologies that have played a pivotal role in the development process. Firstly, we will highlight the significance of Version Control Tool GitHub, which facilitated seamless collaboration and ensured efficient code management throughout the project. Additionally, we will examine the utilization of the widely acclaimed PHP framework Laravel, known for its robustness and flexibility, which provided a solid foundation for building the web application. Furthermore, we will explore the integration of VueJs, a powerful JavaScript framework, that enabled us to develop an interactive and user-friendly interface. Together, these technologies synergistically contributed to the creation of a highly efficient and functional virtual counter system for CNAS.

3.2 Introduction to Git

Software development involves managing a large number of files and assets that undergo frequent changes. As developers, we require a tool that facilitates the administration of these files and ensures consistent updates. This is where Git proves invaluable, providing us with the ability to handle such tasks with ease and flexibility. At its core, Git is a powerful tool that enables multiple individuals to collaborate on the same project while effectively tracking all changes made to the code and files over time.

3.2.1 GitHub implementation

In order to facilitate the development phase and ensure efficient version control for our virtual counter project, we have implemented GitHub. GitHub is a web-based platform that serves as a central repository for Git-based version control systems. It provides a range of tools and features that enable collaborative development, code management, and tracking of changes.

With GitHub, we have a centralized location where we can store and manage our project's codebase. It allows us to create and manage repositories, branches, and commits, making it easy to track changes and work on different features or bug fixes simultaneously. GitHub's version control capabilities ensure that we have a complete history of all modifications, allowing us to roll back changes if needed and maintain code integrity.

GitHub also offers collaborative features that enhance team collaboration and communication. We can create issues and assign them to team members, facilitating task management and bug tracking. Additionally, GitHub provides a platform for code review, allowing team members to review and provide feedback on each other's code, ensuring code quality and consistency.

By utilizing GitHub, we benefit from a robust and scalable infrastructure for our project's version control needs. It streamlines our development process, enables efficient collaboration, and ensures the traceability and integrity of our codebase.

3.2.2 Advantages of Github in Development

In this section, we will explore the advantages of utilizing GitHub in our development workflow. GitHub, as a powerful version control system and collaboration platform, offers a range of benefits that enhance the efficiency and effectiveness of our project.

- **Version Control:** GitHub allows for efficient and effective version control, enabling easy tracking of changes, branching, and merging of code. This ensures that the project's codebase is well-managed and allows for easy collaboration among team members.
- **Collaboration and Teamwork:** GitHub provides a platform for seamless collaboration and teamwork. It allows multiple developers to work on the same project simultaneously, facilitating efficient communication, code sharing, and coordination of tasks. Features like pull requests and code reviews enhance collaboration and ensure high code quality.
- **Code Integrity and History:** GitHub maintains a complete history of all code changes, making it easy to track modifications, roll back to previous versions if necessary, and maintain code integrity. This helps in identifying and resolving issues, ensuring a stable and reliable codebase.
- **Project Management:** GitHub offers project management features such as issue tracking, task assignment, and milestone tracking. These tools streamline project management, enhance organization, and ensure that tasks are tracked and completed in a timely manner.
- **Community and Open Source Collaboration:** GitHub has a large community of developers and provides a platform for open-source collaboration. It enables easy sharing of code, contribution to open-source projects, and learning from others in the community.

3.2.3 GitHub Setup and Configuration

In order to effectively utilize the features of GitHub for version control and collaboration, it is necessary to set up a GitHub account and configure Git on your local machine. This section provides step-by-step instructions on how to set up and configure GitHub, enabling you to seamlessly manage and contribute to your project repositories. Follow the steps below to get started:

1. **Create a GitHub Account**: Begin by creating a GitHub account. Visit the GitHub website (`https://github.com`) and sign up for a new account. Provide the required information, such as your username, email address, and a secure password. Once registered, verify your email address to activate your GitHub account.

2. **Install Git**: Proceed to install Git on your local machine if you haven't done so already. Git provides the necessary command-line tools to interact with GitHub repositories. Download the Git installer from the official website (`https://git-scm.com/downloads`) and follow the installation instructions for your operating system.

3. **Configure Git**: After installing Git, configure your Git identity by setting your username and email address. Open the command-line interface (e.g., Terminal, Git Bash) and execute the following commands:

```
$ git config --global user.name "Your Name"
$ git config --global user.email "your-email@example.com"
```

   These settings will be associated with your Git commits and will be visible in the commit history.

4. **Generate SSH Key**: For secure interaction with GitHub repositories, it is recommended to generate an SSH key pair. Generate a new SSH key by executing the following command:

```
$ ssh-keygen -t rsa -b 4096 -C "your-email@example.com"
```

   Follow the prompts to specify the location for storing the key pair and provide a passphrase (optional but recommended). Once generated, add the SSH public key to your GitHub account by navigating to "Settings" -¿ "SSH and GPG keys" and adding the public key.

5. **Configure Remote Repository**: If you are collaborating on an existing GitHub repository, clone the repository to your local machine using the following command:

```
$ git clone git@github.com:username/repository.git
```

   Replace `username` with your GitHub username and `repository` with the name of the repository. This command creates a local copy of the repository on your machine.

By following these steps, you will have successfully set up and configured GitHub for your project, empowering you to effectively manage version control and collaborate with others in your development process.

3.2.4 Github commands

GitHub provides a powerful set of commands that enable efficient collaboration and version control in software development projects. These commands allow developers to clone repositories, create and manage branches, commit changes, push and pull code, merge branches, and initiate pull requests. Understanding these essential commands is crucial for effective GitHub usage and seamless teamwork. In this section, we will explore the key GitHub commands along with their descriptions and usage examples.

- **Clone:** The `git clone` command allows you to create a local copy of a remote repository on your computer. For example: `git clone <repository URL>`.
- **Branch:** The `git branch` command is used to create, list, or delete branches in your repository. For example: `git branch <branch name>`.
- **Commit:** The `git commit` command is used to save changes made to your local repository. For example: `git commit -m "Commit message"`.
- **Push:** The `git push` command is used to upload local repository commits to a remote repository. For example: `git push origin <branch name>`.
- **Pull:** The `git pull` command is used to update your local repository with the latest changes from the remote repository. For example: `git pull origin <branch name>`.
- **Merge:** The `git merge` command is used to combine changes from different branches into the current branch. For example: `git merge <branch name>`.
- **Pull Request:** The `git pull request` command is used to propose changes from a branch to be merged into another branch. For example: `git pull request`.

SCREENSHOTS : GIT BASH, VS CODE TERMINAL

3.2.5 Collaboration and Teamwork

GitHub provides powerful features that enable seamless collaboration and effective teamwork on software development projects. This section explores the various collaborative capabilities offered by GitHub, allowing multiple developers to work together efficiently and coordinate their efforts. From managing branches and pull requests to resolving conflicts and conducting code reviews, GitHub facilitates a collaborative environment that fosters teamwork and enhances productivity. This section demonstrates how to leverage these collaborative features to streamline the development process and maximize the effectiveness of your team.

SCREENSHOTS : ISSUES, Pull requests and code reviews, project boards, Commenting..

3.2.6 Conclusion

In this section, we explored the implementation of GitHub as a powerful collaboration and version control tool for our project. We discussed the setup and configuration process, essential commands for managing repositories, and the benefits of using GitHub for collaboration and teamwork. By leveraging GitHub's features such as branch management, pull requests, issue tracking, and project boards, we have enhanced our team's productivity and streamlined our development process. The use of GitHub has enabled us to effectively collaborate, track changes, and ensure the integrity of our codebase. With its robust features and user-friendly interface, GitHub has become an indispensable tool for our project's success.

3.3 Introduction to Laravel framework

Laravel is one of the most well known web frameworks that is used widely among developers, it is an open-source PHP based framework that uses MVC (Modal-View-Controller) Architecture and offers various tool and features that allows developers to build high-quality applications with such an efficiency and quickness.

Benefits of using Laravel for web development:

- **Expressive syntax:** Laravel offers an expressive and readable syntax that simplifies the process of writing code. It provides a wide range of functions and shortcuts that allow developers to accomplish complex tasks with minimal effort.
- **MVC architecture:** Laravel follows the Model-View-Controller (MVC) architectural pattern, which promotes separation of concerns and enhances code organization. This architectural approach enables developers to create modular and maintainable applications.
- **Powerful ORM:** Laravel's Eloquent ORM (Object-Relational Mapping) simplifies database operations by providing an intuitive and fluent interface to interact with databases. It allows developers to work with database records as objects, making database management and querying a breeze.
- **Robust routing system:** Laravel's routing system allows developers to define clean and flexible routes for their web applications. It supports various HTTP methods, route parameters, and route grouping, making it easy to handle complex URL structures.
- **Blade templating engine:** Laravel's Blade templating engine offers a concise and powerful way to create dynamic views. It provides features like template inheritance, control structures, and reusable components, enabling developers to build modular and reusable UI components.
- **Authentication and authorization:** Laravel simplifies user authentication and authorization processes with built-in functionalities. It provides secure user registration, login, and password reset mechanisms, as well as fine-grained access control using gates and policies.
- **Rich ecosystem and community support:** Laravel has a vibrant and active community of developers who contribute to its growth. The framework benefits from a vast ecosystem of packages and libraries that extend its capabilities, allowing developers to leverage existing solutions and accelerate development.
- **Testing and debugging tools:** Laravel provides robust testing and debugging tools that help developers ensure the quality and reliability of their applications. It supports unit testing, feature testing, and includes convenient debugging tools for efficient troubleshooting.

Overall, Laravel is an excellent choice for building web applications of any size and complexity, the choice of implementing this particular framework has been proven to be a wise decision, and that's due to its powerful set of tools an features that enabled us to create a robust and scalable web application that meets the needs of CNAS and its users.

3.4 Laravel implementation

3.4.1 Installation and setup

In this section we will discuss the installation guide for laravel and its different components. In order to install and setup laravel correctly and without any issues, there is some requirements needs to be fulfilled in case of not using the homestead virtual machine.

These requirements are

- PHP version 7.2.5 or greater.
- BCMath PHP Extension
- Ctype PHP Extension
- Fileinfo PHP extension
- JSON PHP Extension
- Mbstring PHP Extension
- OpenSSL PHP Extension
- PDO PHP Extension
- Tokenizer PHP Extension
- XML PHP Extension

If the previous requirements are validated, we then move on to the installation guide for laravel, to do so it is highly recommended to follow the steps listed.

1. **Install Laravel and its dependencies:**

   Ensure that you have PHP installed on your system. Laravel requires PHP 7.4 or higher.

   Install Composer, a dependency manager for PHP, if you haven't already. Composer is used to install Laravel and manage its dependencies. Open a terminal or command prompt and run the following command to install Laravel globally on your system:

   ```
   composer global require laravel/installer
   ```

2. **Configure the development environment:**

   Laravel requires a web server and a database to run. You can use popular web servers like Apache or Nginx, along with databases like MySQL or SQLite.

   Ensure that your web server and database server are properly installed and configured. If needed, consult their respective documentation for installation and setup instructions.

3. **Initialize a new Laravel project:**

   Once Laravel is installed and your development environment is set up, you can create a new Laravel project.

   Open a terminal or command prompt and navigate to the directory where you want to create your project. Run the following command to create a new Laravel project:

   ```
   laravel new your-project-name
   ```

   Replace "your-project-name" with the desired name for your project. This command will create a new directory with the specified project name and install the necessary files and dependencies.

4. **Test your installation:**

   Change into the project directory:

   ```
   cd your-project-name
   ```

   Start the local development server by running the following command:

   ```
   php artisan serve
   ```

   By default, the development server will start on (`http://localhost:8000`)

   Open your web browser and visit that URL. If you see the Laravel welcome page, it means your installation was successful.

5. **PHP configuration:**

   Open the PHP configuration file (php.ini) on your system. The location of this file may vary depending on your operating system and PHP installation.

   Ensure that the following PHP extensions are enabled by uncommenting their respective lines (remove the semicolon ";" at the beginning of the line if present):

   ```
   extension=fileinfo
   extension=openssl
   extension=pdo_mysql
   ```

6. **Creating the ".env " file :**

   Laravel comes with a '.env.example' file by default.

   - Make a copy of this file and rename it to '.env' by running the following command:

     ```
     cp .env.example .env
     ```

   - Setting up the environment variables:

     Open the '.env' file in a text editor.

     Update the variables according to your development environment. For example, you might need to set the database credentials:

     ```
     DB_CONNECTION=mysql
     DB_HOST=127.0.0.1
     DB_PORT=3306
     DB_DATABASE=your_database_name
     DB_USERNAME=your_database_username
     DB_PASSWORD=your_database_password
     ```

     You can also configure other variables like the application URL, mail settings, caching drivers, and more. Refer to the comments in the '.env.example' file or Laravel's documentation for more information on available options.

7. **Generating the application key:**

   Laravel requires an application key for secure encryption and other purposes.

   Run the following command to generate the key:

   ```
   php artisan key:generate
   ```

8. **Protecting sensitive information:**

   Ensure that the .env file is not publicly accessible. It should be kept outside of your version control system or any public directories.

   If you deploy your application to a production server, you may need to set the environment variables directly on the server or through the server's configuration management tools.

3.4.2 Laravel directory structure

In a Laravel project, the directory structure is designed to provide a clear organization for your application's files. Understanding the key directories and files will help you navigate and manage your Laravel project effectively. Here's an explanation of the purpose of each directory and some important files:

- **app:** Contains the core application code, including controllers, models, and other PHP classes specific to your application's domain logic.
- **bootstrap:** Contains the files responsible for bootstrapping the Laravel framework and initializing the application environment.

- **config:** Contains configuration files for various aspects of your application, such as database connections, caching, mail settings, and more.
- **database:** Contains database-related files, including migrations for managing database schema changes, seeders for populating the database with sample data, and factories for generating test data.
- **public:** The web server's document root should be set to this directory. It contains the entry point for your application (index.php) and serves as the public-facing directory for static assets, such as CSS, JavaScript, and image files.
- **resources:** Contains views, language files, and frontend assets used by your application. views: Contains the Blade templates that define the UI of your application. lang: Contains language files for localization and internationalization. assets: Contains frontend assets, such as CSS, JavaScript, and images, that will be compiled and optimized by Laravel Mix.
- **routes:** Contains route definition files that specify how incoming requests should be handled by your application. web.php: Defines routes for web-based endpoints. api.php: Defines routes for API endpoints. You can create additional route files for organizing routes based on specific functionalities or modules.
- **storage:** Contains files generated by your application, such as logs, cached views, and uploaded files. app: Contains files generated by your application, such as cached config files, logs, and other temporary files. framework: Contains framework-generated files, including cached views, sessions, and routes. logs: Contains log files generated by your application.
- **tests:** Contains test files and directories for automated testing of your application. Feature: Contains feature tests, which test the application's behavior from the user's perspective. Unit: Contains unit tests, which test individual components of your application in isolation.
- **.env:** The environment file that holds environment-specific configuration values for your application. Contains settings such as database connections, mail configurations, and environment variables. It's important to keep this file secure and not expose any sensitive information.
- **composer.json and composer.lock:** These files manage the project's dependencies using Composer, a PHP dependency manager. composer.json lists the project's dependencies and defines autoloading rules. composer.lock locks the versions of the dependencies to ensure consistent installations.
- **artisan:** The command-line interface (CLI) tool for executing various commands within your Laravel application. Allows you to run tasks such as running migrations, generating code, and running tests.

Understanding the purpose of each directory and file in a Laravel project will help you navigate and locate the appropriate locations for adding or modifying code, configurations, and assets. It's important to maintain the integrity of the directory structure while organizing your code and assets within the appropriate directories.

3.4.3 Routing

Routing is an essential aspect of web development, and Laravel provides a powerful and flexible routing system. Here's an explanation of routing in Laravel, including how to define routes, work with route parameters, and utilize route grouping and naming.

Routing in Laravel refers to the process of mapping incoming HTTP requests to specific actions or handlers within your application. It determines how different URLs are handled and defines the endpoints through which users can access various functionalities of your application.

In Laravel, routes are typically defined in the 'routes' directory, specifically the 'web.php' and 'api.php' files.

1. **Basic route definition:**

   A basic route is defined using the 'Route' facade's methods, such as 'get', 'post', 'put', 'patch', and 'delete'.

   Here's an example of a basic route definition:

   ```
   Route::get('/home', function () {
       return 'Welcome to the home page!';
       });
   ```

   This route responds to the 'GET' request to the '/home' URL and returns the specified message.

2. **Route parameters:** You can define routes with parameters that are passed as segments in the URL.

   Here's an example of a route with a parameter:

   ```
   Route::get('/users/{id}', function ($id) {
       return 'User ID: ' . $id;
       });
   ```

   This route matches URLs like '/users/1', '/users/2', etc., and the parameter 'id' is passed to the route closure as an argument.

3. **Route grouping and naming:**

   Laravel allows you to group related routes and assign names to them for easy referencing and organization.

   - **Route grouping:**
     Route grouping allows you to apply common attributes or middleware to a group of routes.

     Here's an example of route grouping with a shared middleware:

```
Route::middleware('auth')->group(function () {
    Route::get('/dashboard', function () {
        return 'Welcome to the dashboard!';
        });
        Route::get('/profile', function () {
            return 'Welcome to your profile!';
            });
            });
```

In this example, the routes '/dashboard' and '/profile' are grouped together and share the 'auth' middleware, which ensures that only authenticated users can access them.

- **Route naming:**
  Assigning names to routes helps in referencing them within your application, such as generating URLs or redirecting to specific routes.

  Here's an example of naming routes:

```
Route::get('/posts', function () {
    return 'List of posts';
    })->name('posts.index');

    Route::get('/posts/{id}', function ($id) {
        return 'Post ID: ' . $id;
        })->name('posts.show');
```

  In this example, the routes '/posts' and '/posts/id' are named as 'posts.index' and 'posts.show', respectively. These names can be used later to generate URLs or redirect to these routes.

  By understanding and utilizing routing in Laravel, you can define the endpoints for your application, handle various HTTP methods, work with dynamic route parameters, group related routes, and assign names for easy referencing. Laravel's routing system provides the flexibility and convenience required to build robust and maintainable web applications.

3.4.4 Controllers

Controllers play a crucial role in Laravel applications as they handle the logic and actions associated with different routes. Here's an explanation of creating and using controllers in Laravel, defining controller methods and actions, and understanding the separation of concerns between routes and controllers.

The creation of an controller is done by the following artisan command:

```
php artisan make:controller
```

For instance, to create the user controller we used the following command:

```
php artisan make:controller UserController
```

This command will generate a new ‘UserController‘ class in the ‘app/Http/Controllers‘ directory.

3.4.5 Views and Blade templates

//

3.4.6 Models and Eloquent ORM

//

3.4.7 Database Migrations and Seeders

//

3.4.8 Form Handling and Validation

//

3.4.9 Authentication and Authorization

//

3.4.10 Middlewares

//

3.4.11 Error Handling and Logging

//

3.4.12 Testing in Laravel

//

3.4.13 Deployment and Production Considerations

//

3.4.14 Best Practices and Tips

//

3.4.15 Conclusion

//

3.5 VueJs implementation

//

3.5.1 Overview of the VueJs framework

//

3.5.2 Explanation of the different components of the system implemented using VueJs

//

3.5.3 Code snippets and screenshots to illustrate the implementation details

//

3.5.4 Discussion of the challenges faced and how they were overcome

//

3.6 Integration of Laravel and VueJs

//

3.6.1 Explanation of how Laravel and VueJs were integrated to create the final system

//

3.6.2 Code snippets and screenshots to illustrate the integration details

//

3.6.3 Discussion of the challenges faced and how they were overcome

//

3.7 Conclusion

//

3.7.1 Summary of the key points covered

//

3.7.2 Reflection on the overall implementation process

//

3.7.3 Discussion of future work and potential improvements

//