Statement and Confirmation of Own Work

A signed copy of this form must be submitted with every assignment.
If the statement is missing your work may not be marked.

Student Declaration

I confirm the following details:

Candidate Name:	SHIHAB SAJID MIRZA
Candidate ID Number:	P00190603
Qualification:	NCC L5DC
Unit:	COMPUTING PROJECT (PROJECT REPORT)
Centre:	ZCAS UNIVERSITY

I have read and understood both NCC Education's *Academic Misconduct Policy* and the *Referencing and Bibliographies* document. To the best of my knowledge my work has been accurately referenced and all sources cited correctly.

I confirm that this is my own work and that I have not colluded or plagiarised any part of it.

Candidate Signature:	
Date:	20/04/2023

TABLE OF CONTENTS:

ACKNOWLEGEMENTS:

First and foremost, I show my heartfelt gratitude towards my parents who sponsored me for this journey and provided unwavering support throughout it. Additionally, I thank VTE for providing me with the opportunity to produce a computing project/solution for them. I also thank VTE staff for helping me with any queries I had regarding their business processes. Finally, I extend my appreciation towards all the supervisors, lecturers and colleagues that helped me during the development phase of the system.

ABSTRACT

The scope and nature of this project is such that it involved the analysis, design, implementation and lastly, testing of the system developed for VTE (Visionary Technologies Enterprise). the analysis step consisted of interviewing or just generally conversing with the VTE staff. The design step was short and usually involved coming up with Backend architecture using class diagrams etc., frontend architecture and design using Figma and the necessary Database ERDs. The implementation or coding of the system was done using the MERN stack (Mongo dB, Express.js, React.js (With typescript) and Node.js). However, MySQL was also used as the primary data store. The system was developed in a responsive manner to allow mobile users to also use it effectively.
The testing strategy used was black box testing to find anything that is not working or having bugs and apply an immediate fix to it.
The system was given the nickname/name ***"Tasked.it"***.
The website is an admin dashboard style website, so it is only to be used by VTE staff.

A summary of the outcomes:

Easier freelancer management: The admins can now easily manage the freelancers by assigning projects to them, creating new freelancers and editing existing freelancers.

Better communication: Freelancers and admins can easily exchange messages through the system. Without the need of third-party apps.

Easier project management: Projects can easily be added, removed and broken down into smaller tasks which can then be assigned to the freelancers with the appropriate skill roles.

Client satisfaction: Clients more satisfied with well managed projects that are of high quality and are produced within ample time.

Chapter 1 - INTRODUCTION

1.1 System developed.

The system that was developed was a multifunctional admin dashboard which was later nicknamed or named “Tasked.it”. The system has a login page for freelancers that are registered with VTE and a login page for Admins which will be the VTE staff.
The admins of the system are able to add or create new freelancers or even create new admins using an admin key. During the creation of freelancers, they are assigned with roles or skills that they are qualified with e.g., front end developer, database administrator, UI designer etc.
The admins can also create, edit and remove projects and freelancer roles from the system. During the creation of a project, the project is broken down into what’s known as “tasks” and these “tasks” are then assigned to freelancers and given a due date to finish.
The admins can also view (on the home page of admin panel) high level analytics and statistics such as: the number of freelancers in the system, number of tasks finished. Freelancer to task ratio, admin and freelancer engagement etc.
Other functions are also available such as exchanging messages between freelancers and admins, editing profile, changing password, Freelancers being able to see the projects they are working on; they can mark a task as complete if its complete etc.

1.2 Justification of methods used.

The Methodologies and frameworks used during the lifecycle of the system was mainly the agile methodology and the object-oriented methodology. The Agile approach is a technique for overseeing a project by dividing it into numerous stages. It necessitates continuous communication with stakeholders and constant enhancement at each step. When the project commences, teams undergo a cycle of preparation, implementation, and assessment. (*asana.com, October 15th 2022).*
The object-oriented methodology objects prioritize analysis, design, and implementation. During the analysis stage of this approach, UML diagrams are crafted to create a visualization of the software, which facilitates the comprehension of the system by those who are not technically inclined *(ogcio.gov.hk, date unknown).* At the analysis stage of VTE, the object-oriented methodology was used to provide a lucid and non-technical visual representation of the system to stakeholders, to help them fully understand its intended purpose. For the remainder of the project, the Agile methodology was utilized to fragment the work into iterations with set timeframes for completion, and evaluated based on the objectives of the project.

1.3 Solution that emerged.

The final solution that had emerged was a web-based admin dashboard style website that allowed VTE registered freelancers to easily see the tasks they are assigned to for the projects they are working on and admins to manage freelancers and projects. The website enhances the work flow for VTE causing a boosted level of confidence in their projects that they make for their clients, hence increasing revenue. The system allows admins to create new projects with tasks mentioning the due dates and assign them to the appropriately skilled freelancers to complete them. The tasks are given a price allocation for freelancers to see how much they will get after they complete the task. This will cause more freelancers to come and register with VTE hence causing a net increase in the number of projects VTE can take on at once.
The Home page of the admin panel shows various statistics. Such as the number of projects completed. The type of projects completed, the number of freelancers in the system, number of tasks finished. Freelancer to task ratio, admin and freelancer engagement etc. VTE can benefit from quick and precise decision-making by senior management, which is facilitated by the information provided by these statistics. As a result, the company can reduce mistakes and financial losses, leading to overall growth of the business.

1.4 Aims.

- Design and develop a flexible and multipurpose web-based application
- Allow seamless management of projects within the company
- Enable faster work flows and no delays by freelancers using deadline functionality
- Enhance project management within the company and strengthen the link between freelancers, company and clients
- Digitize the workflow of projects within the company.

1.5 Objectives.

- Conducting an analysis of business processes.
- Identifying the system requirements.
- Roughly planning the system's appearance and functionality.
- Creating a user interface design.
- Implementing the user interface using React.js and bootstrap for styling assistance.
- Designing the database using ERDs and other tools.
- Building and implementing the database in MySQL.
- Developing the backend architecture on paper.
- Implementing the backend using Node.js and connecting it to the database.
- Consuming backend APIs with the frontend.
- Testing and debugging all system layers (frontend, backend, database).
- Deploying all system layers (frontend, backend, database).
- Preparing a project report after achieving the desired system functionality.

1.6 Overview Of remaining chapters.

Chapter 1 introduced the project.
Chapter 2 covers analysis, including requirements, UML diagrams, and system architecture.
Chapter 3 outlines the structural and behavioral models.
Chapter 4 covers the programming aspect of the project i.e., language choice, data migration, and training.
Chapter 5 discusses project, risk, configuration management, and testing.
Chapter 6 concludes the report with results, problems, and potential improvements.
Chapter 7 reflects on what was learnt, project approach, and future improvements.

1.7 Conclusion.

This chapter of the report introduces the multifunctional admin dashboard called "Tasked.it," developed for VTE. The chapter describes the features of the system, including login pages for both freelancers and admins, creation of new freelancers and admins, project and freelancer role management, and high-level statistics analysis. The chapter justifies the use of Agile and object-oriented methodologies in the development process. The solution that emerged is a web-based admin dashboard that enhances workflow for VTE and allows admins to manage projects and freelancers effectively. The chapter also outlines the aims and objectives of the project and provides an overview of the remaining chapters.

Chapter 2 - ANALYSIS

2.1 Introduction.

In the following analysis phase, the focus is on gathering user requirements and presenting them in a clearer and organized way. This section or chapter will discuss the techniques used to obtain requirements and categorize them into functional and non-functional requirements. The chapter will also depict a use case diagram that illustrates the collected requirements. Furthermore, it will include a system architecture diagram that will be presented through an initial class diagram and a system architecture model.

2.2 Requirements Gathering.

The process of requirements gathering plays a critical role in defining the system's purpose, functionality, and implementation approach, and to accomplish this, there are several techniques that can be used, which are outlined right below.

2.2.1 Interviews.

Interviews are a classic approach to gathering requirements, involves conversing with stakeholders in a face-to-face or virtual setting. The key objective of this method is to extract the expectations, desires, and apprehensions of stakeholders about the system in question. These conversations can be conducted in various styles: structured, semi-structured, or unstructured, and can yield both functional and non-functional requirements. *(Tutorialspoint.com, 17th January 2020)*

For these reasons, I chose to interview the both the CEO and many other staff Working in VTE. The information that was gathered included Sales records overview, Business processes of VTE, how VTE stores data, how VTE communicated with freelancers and the future plans of VTE as a business.

2.2.2 Questionnaire's.

Questionnaires, a commonly employed technique for requirements gathering, is an approach that involves sending out a set of pre-defined questions to stakeholders via postal or digital means to gather their perspectives on the system being developed. This approach is well-received because it's straightforward to administer and can be used to reach a large number of stakeholders quickly. The questions can either be closed or open ended, and are intended to collect both functional and non-functional requirements. *(www.pmmajik.com. No date)*

To prove this method effective, questionnaires were emailed to freelancers that were registered with VTE. The questions attempted to extract various types of information such as freelancer satisfaction with the system, how long they have been working, the pay etc.

2.2.3 Prototyping.

Prototyping, a modern and widely used approach to requirements gathering, involves collecting preliminary requirements to construct an initial version of the solution, also known as a prototype. This prototype is presented to the client for feedback, additional requirements, and further modifications, which are integrated into the application, and the process is repeated. This cycle continues until the final product meets the critical needs of the business or the agreed-upon number of cycles is reached. *(Tutorialspoint.com, 17th January 2020).*
This technique was used in a way that every other day I used to present how far I have gone with the system to VTE and they would tell me what to change, what to remove and what to add.

2.3 Requirements Documentation.

In this section, a short overview of the gathered requirements will be presented, outlining both the functional and non-functional requirements, as well as their priority ranking based on the MOSCOW prioritization technique.

2.3.1 Functional Requirements.

Functional requirements are the specific product characteristics or functions that developers need to include into the system to facilitate user tasks. These requirements represent the specific actions that the system must be able to perform, without these, the system would be considered incomplete. The functional requirements for this system are mentioned below:

Requirement ID	Name	Description	MOSCOW Ranking
1	Login functionality	Login pages for admins and freelancers and authentication mechanism to differentiate privilege levels.	MUST HAVE
2	View list of freelancers	Admins should be able to view the list of freelancers registered in the system	SHOULD HAVE
3	View list of admins	Admins should be able to view the list of other admins registered in the system	SHOULD HAVE
4	Edit and remove freelancers	Admins should have a feature to edit and remove the list of freelancers	SHOULD HAVE
5	Edit and remove Admins	Admins can’t edit and remove admins from the list of admins	WON’T HAVE
6	Message freelancer	Admins and freelancers can exchange messages	COULD HAVE
7	Message admin	Admins and other admins can exchange messages	COULD HAVE
8	List Clients	Admins should be able to see list of clients	SHOULD HAVE

9	Add new clients and freelancers	Admins should be able to add new clients, and register new freelancers	MUST HAVE
10	Edit and Remove clients	Admins should have a feature to edit and remove the list of clients	SHOULD HAVE
11	Create new projects	Admins should be able to create new projects	MUST HAVE
12	Edit and Remove projects	Admins should be able to edit and remove the list of clients	SHOULD HAVE
13	View Tasks	Admins should be able to view all tasks	SHOULD HAVE
14	Edit and remove tasks	Admins should have a feature to edit and remove the list of tasks	SHOULD HAVE
15	Create tasks	Admins should be able to create and assign tasks	MUST HAVE
16	List projects	Admins should be able to view all projects	SHOULD HAVE
17	Show analytics and statistics	A high-level statistics page should be there showing data using graphs charts etc.	COULD HAVE
18	Editing profile for both admins and freelancers	Admins and freelancers can edit their profiles	COULD HAVE
19	Change password for both admins and freelancers	Admins and freelancers can change passwords	COULD HAVE
20	Mark as complete	Freelancers can mark task as complete if completed	SHOULD HAVE

TABLE 1 – FUNCTIONAL REQUIREMENTS

2.3.2 Non-Functional Requirements.

When it comes to Non-Functional requirements, we mostly refer to the quality aspect or goodness of a given system. These requirements describe how the system should look and feel, how it should behave and perform, and how it should handle different situations and scenarios. The non-functional requirements for this system are given below:

Requirement ID	Name	Description	MOSCOW Ranking
1	Usability	The final system should be easy to use for users and easy to navigate around for new users.	SHOULD HAVE
2	Compatibility	The Final system should be capable of running on different browsers.	MUST HAVE
3	Security	The final system should be able to store sensitive user data securely behind tight authentication mechanisms and security defenses such as firewalls anti-virus and hashing of passwords.	MUST HAVE
4	Availability	The final system should be available to its end users at all times of the day unless there is maintenance or downtime	SHOULD HAVE
5	Performance	The final system should run smoothly on low end devices and the loading times should be as fast as possible	COULD HAVE

TABLE 2 – NON-FUNCTIONAL REQUIREMENTS

2.4 Use Cases.

Use cases are functional requirements of a system and can be depicted in a use case diagram that shows how the user interacts with the system. This diagram includes actors (users of the system), use cases, and the relationship between actors and use cases. Actors and use cases represent the users and functional requirements of the system, respectively.

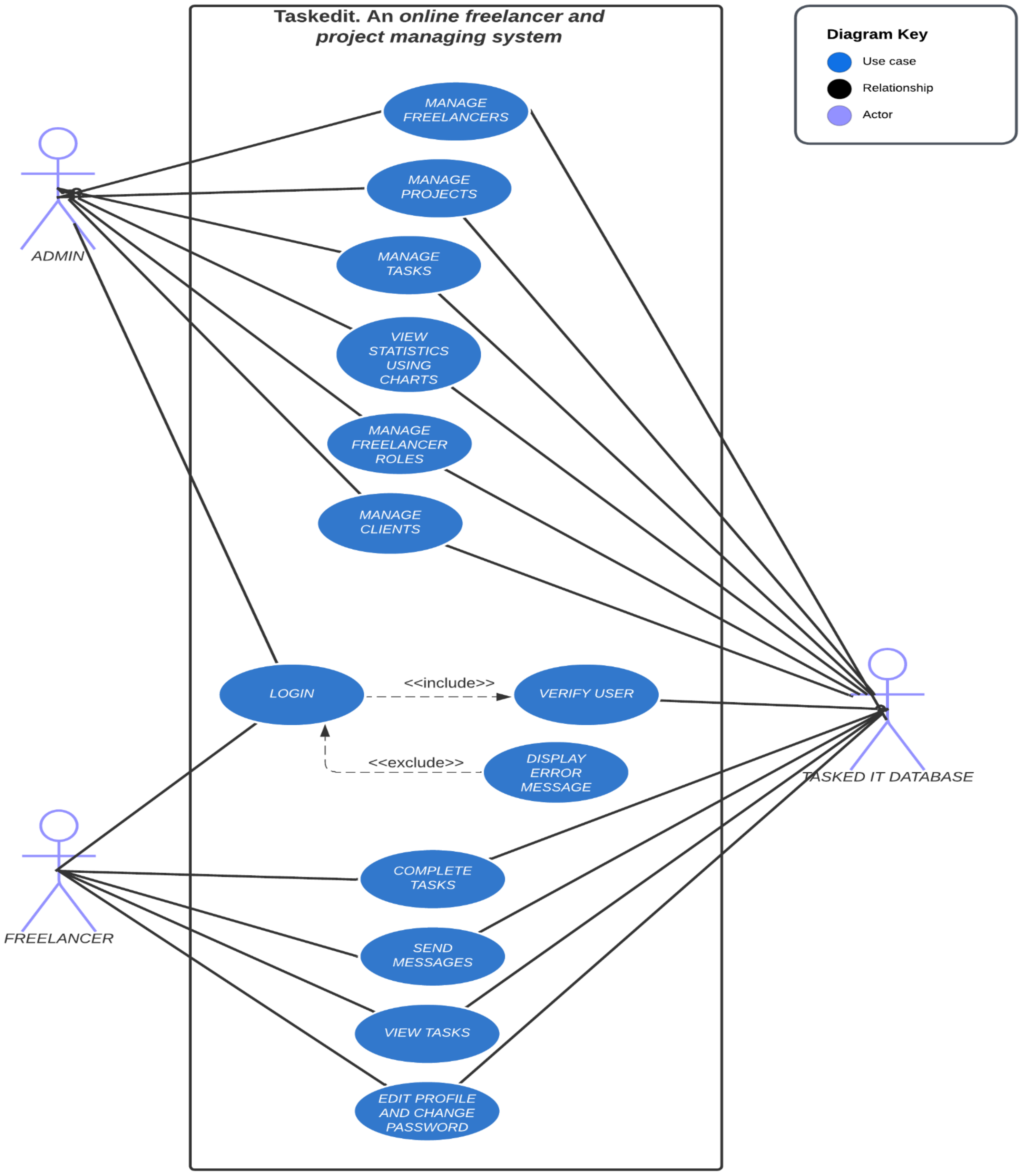


Figure 1 – use case diagram

2.5 System Architecture.

A System architecture diagram is a diagram that outlines the components of a system and how they are connected to each other, or it shows the interactions that happen between them and the relationships between them. *(Wikipedia.com, July 2012).*

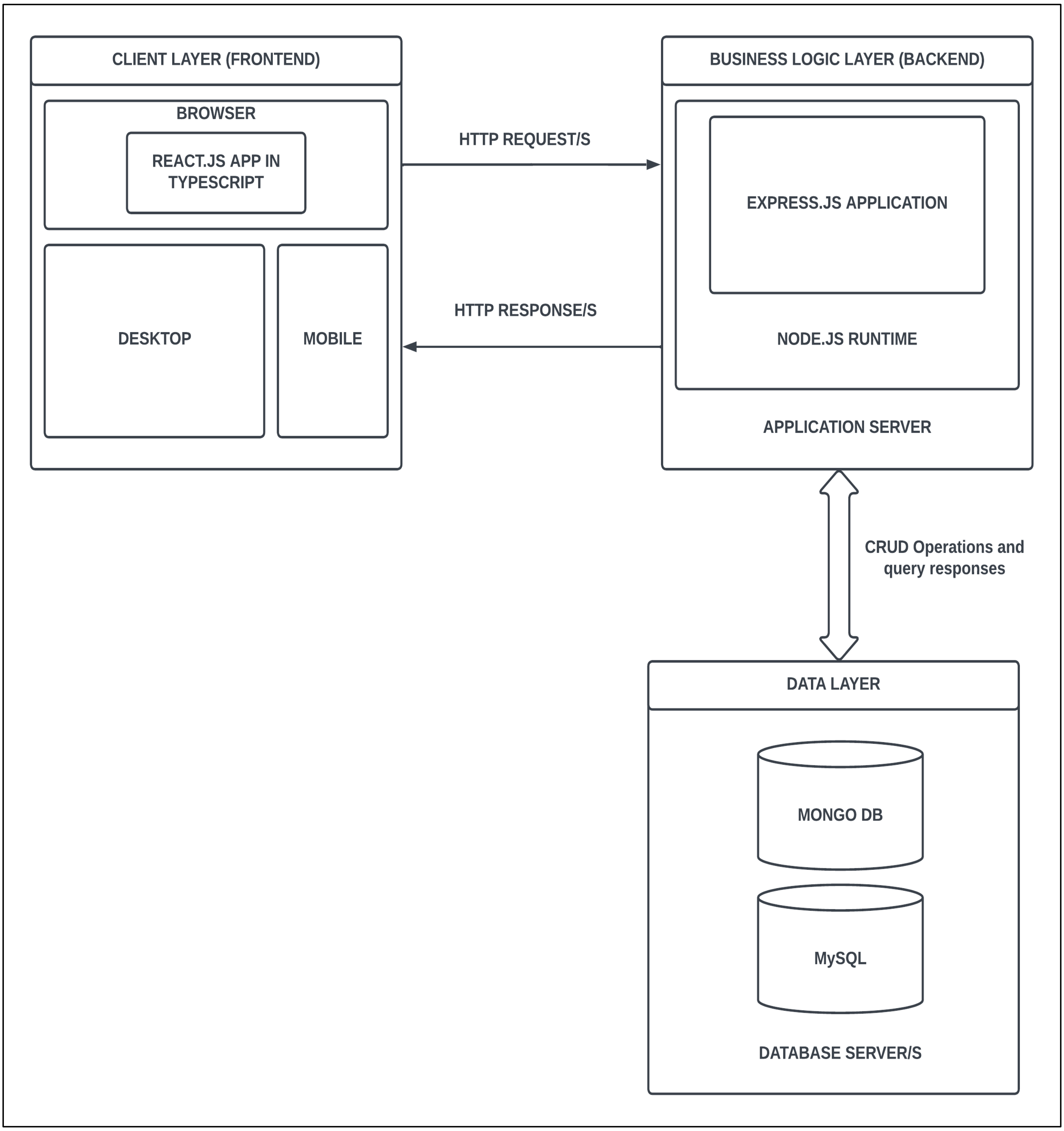


Figure 2- system architecture.

2.6 Initial Class Diagram.

The following is just an initial class diagram that presents the classes without any attributes or methods. For further information regarding class diagrams, please refer to chapter 3, section 3.2.1.

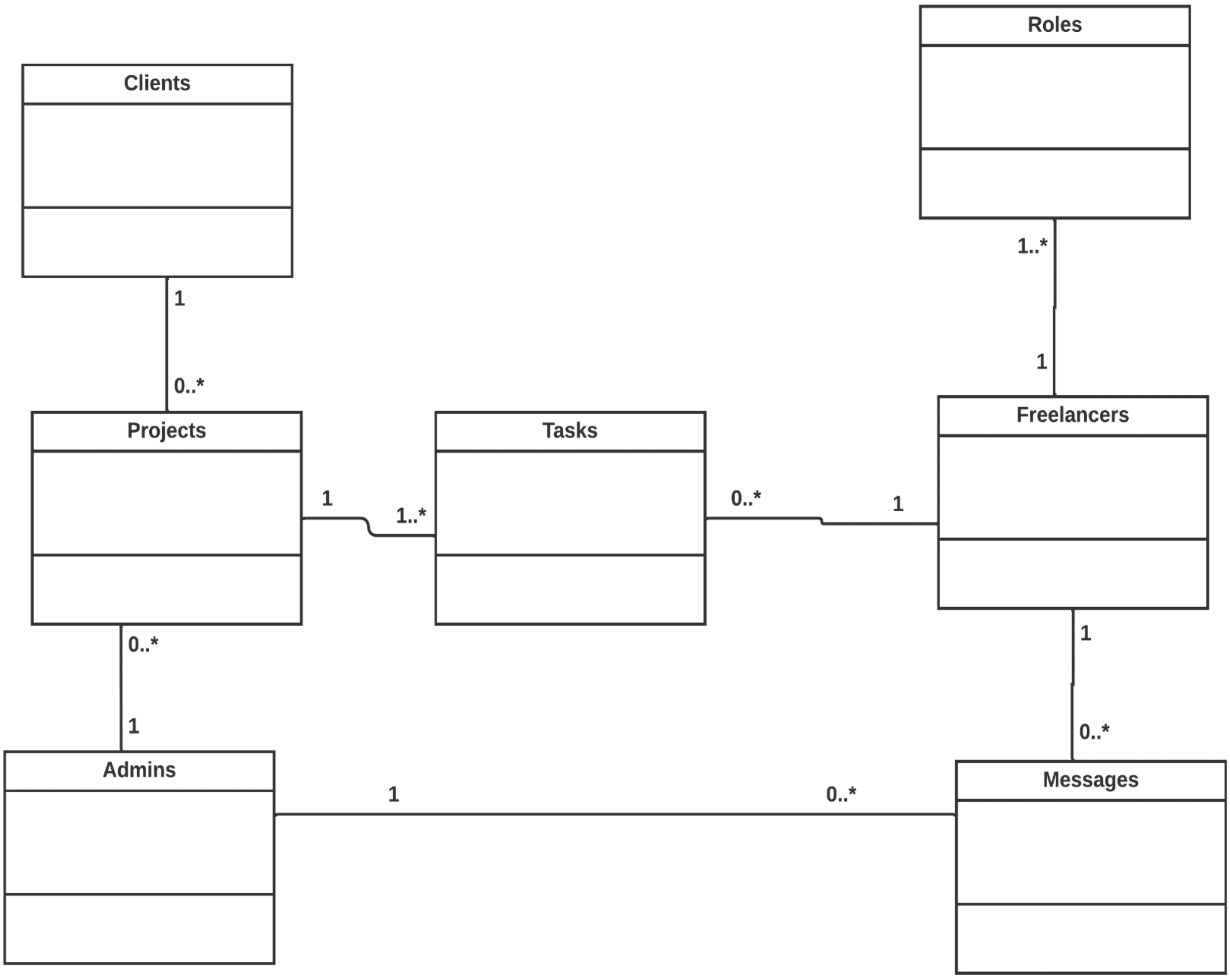


Figure 3 – initial class diagram

2.7 Conclusion.

This chapter has managed to discuss the various Requirements gathering methods that were used to collect the overall functional requirements and non-functional requirements for the System to be developed for VTE. It has then laid out these FRs and NFRs in the two tables respectively and ranked them based on the MOSCOW prioritization technique. Furthermore, the chapter managed to illustrate the use cases of the system using a use case diagram, the system architecture using a system architecture diagram and finally it showed an initial class diagram. The chapter after this one will discuss the design of the system based on the behavioral and structural models.

Chapter 3 – DESIGN

3.1 Introduction.

The design phase involves coming up with various was to implement the functional requirements and non-functional requirements that were previously identified during the analysis phase. In order to do this, we have to different types of modelling methods. During the following phase, we can use the structural and behavioral models. A detailed class diagram which will now be a continuation of the unfinished class diagram presented in chapter 2.6 will be part of the structural model. Additionally, a sequence diagram will be illustrated which will be part of the behavioral model. This sequence diagram will be split up into two, (one for Admins and one for Freelancers).

3.2 Structural Model.

The structural model is composed of the various objects within the final system and the fixed connections that exist among them. These objects can be grouped together into kind of small packages or subsystems. To define the structural model, diagrams known as class diagrams are used. *(ibm.com, 2023-01-16).* The detailed class diagram showcased represents the structural model, which illustrates the composition of the developed system for VTE. The system is also known as Tasked.it.

3.2.1 Class Diagrams.

Class diagrams are utilized to depict, document, and describe the system which will be created before it is built. Class diagrams also aid in the creation of code for the system when using object-oriented programming. Class diagrams show the attributes and methods that describe each system element as illustrated below:

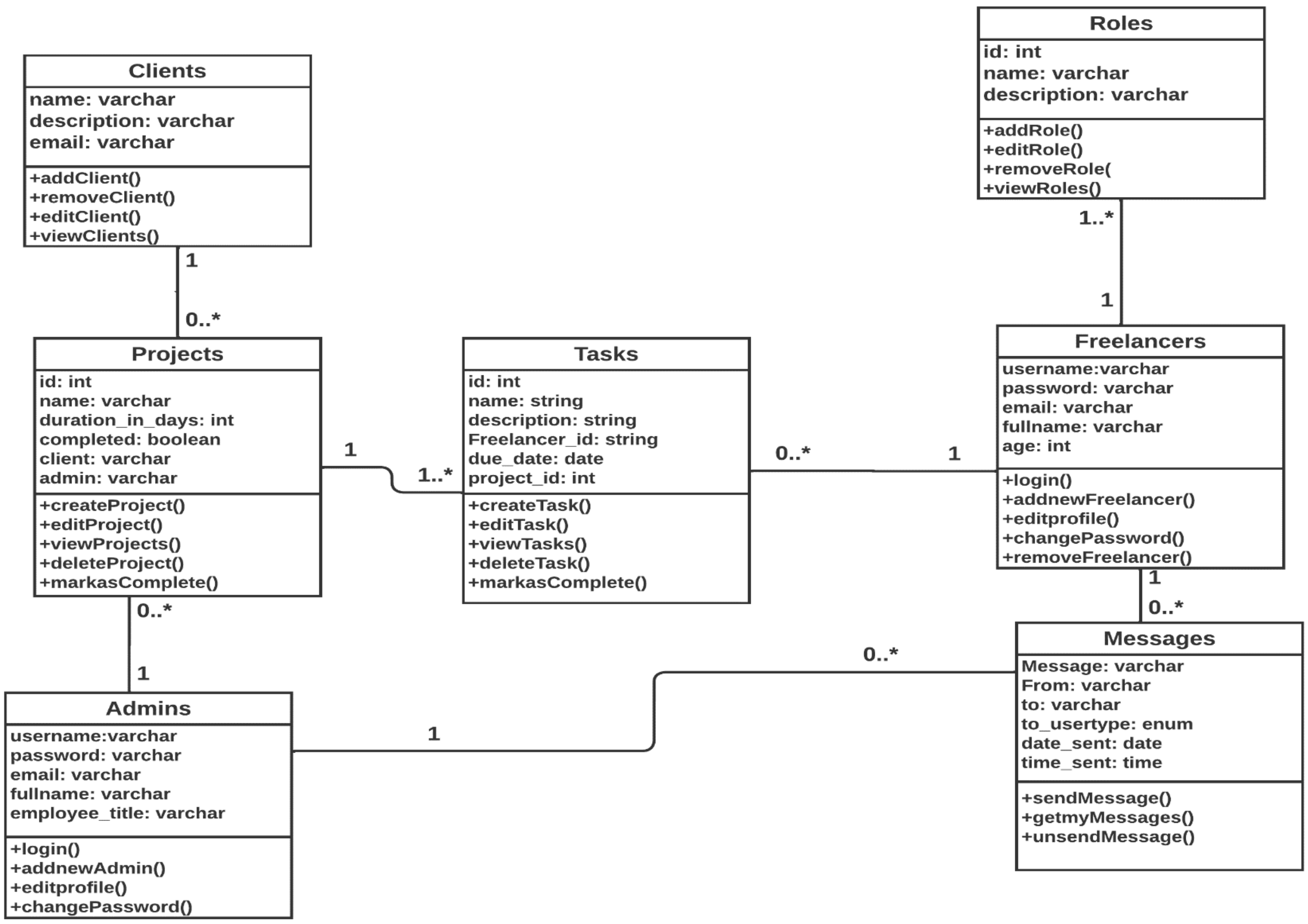


Figure 4 – class diagram final

3.3 Behavioral Model.

The behavior of the system when it is used by outside users is represented by behavioral models. It depicts how a system will act under specific circumstances. *(geeksforgeeks.com, 17th June 2020).*
The sequence diagram is the behavioral model I've chosen. Sequence diagrams are used to display the sequential interactions among the system's objects. The sequence diagrams also show the user's responses that the system provides based on various circumstances.
Sequence diagrams show the system's users, subsystems, and their interactions.

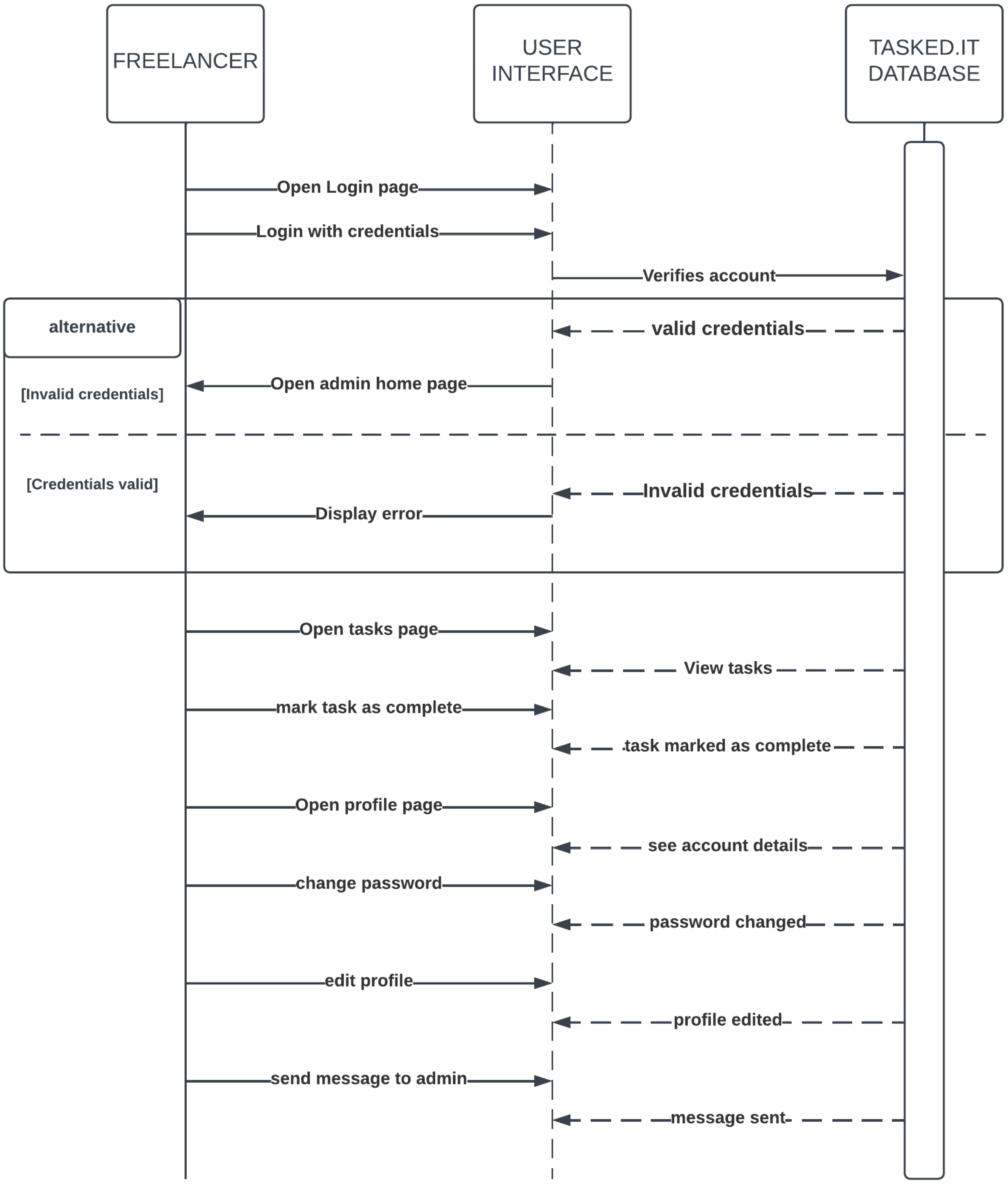


Figure 5 – sequence diagram for freelancers

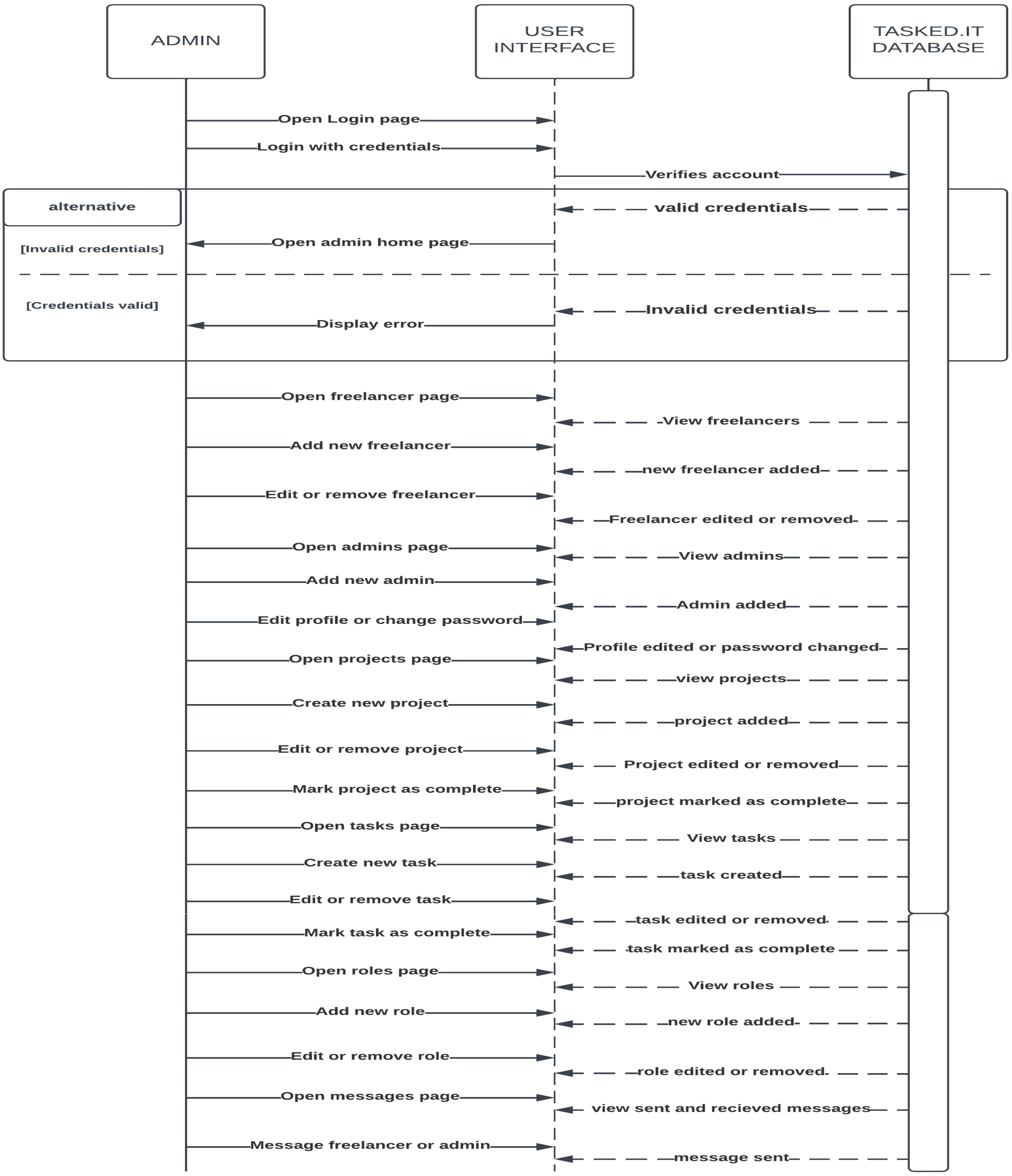


Figure 6 – sequence diagram for admins

3.4 Conclusion.

To conclude, this chapter focused on the designing aspect of developing software. It did this by illustrating the system through structural models and behavioral models which in turn used a class diagram and sequence diagrams (one for freelancers in the system and one for admins of the system). The next chapter will now start discussing on the implementation process for developing the system.

Chapter 4 – IMPLEMENTATION.

4.1 Introduction.

This stage is when we now put all the planning we have done so far into action and come up with a working system the way it was planned to work and desired. All and any research and analysis done during previous stages so far is now going to be transformed into a fully working system.
(opentextbc.ca, no date)
The following chapter is going to cover any programming languages or frameworks used for the development of the system, the system cutover and data migration aspects which were done from the development to the implementation architecture and it will also cover the training program conducted for the system.

4.2 programming language choices.

A programming language is utilized by coders to produce software, scripts, or other instructions for computers to follow. It is a language that computers can understand. *(computerhope.com, 03/03/2021).* For this project, the only Languages used that count as languages are **Typescript** for the frontend, **JavaScript** for the backend (Running on **Node.js**) and **SQL** for the primary database and **Mongo dB** for secondary database. The use of **CSS** was very little to none and **bootstrap** was rather used a lot. For the front-end, a JavaScript library known as **React.js** was used. This was used to build the front-end UI via **Typescript**. As for the backend, a framework known as **Express.js** was used to build the http server. These languages in combination with the frameworks and libraries mentioned were used to create the Tasked.it dashboard website.

4.2.1 Typescript (with React.js) for frontend (styled using bootstrap).

Typescript is an open-source and free programming language built on top of JavaScript and is a superset of JavaScript. JavaScript does not have a static type system, Typescript fixes this by introducing variable types to JavaScript. *(www.typescriptlang.org, no date).* I used Typescript because it is mainly used to avoid production errors and bugs causes by the weakly typed language JavaScript.
React.js on the other hand, is a JavaScript library for building user interfaces. React.js allows us to build websites quickly and more efficiently compared to using vanilla (plain) JavaScript. It does this by making use of reusable components which be thought of as Lego blocks. *(blog.hubspot.com, June 27 2022).*
I used react.js because it was to reuse the components that I made compared to plain JavaScript.
The boilerplate or skeleton of the website was come up with using react.js but instead of using JavaScript, we used typescript for type safety. Bootstrap is a framework for CSS that exposes CSS selector classes which can be used on html elements to style the html elements. It contains design templates for forms, navigation, typography and many other interface components.

4.2.2 JavaScript (with Express.js) for backend running on Node.js.

For many years, JavaScript has been mainly used as a client-side language or "a browser language" to manipulate the HTML found on web pages. However, nowadays it is also used as a server-side language that runs on a powerful JavaScript runtime known as Node.js. It is very possible to create powerful http servers using the standard JavaScript http library. However, there are many other http frameworks. The one I went with is known as Express.js. It allowed me to create an http server seamlessly and allowed me map it to different routes using its router functionality known as express router. The reason I used JavaScript for the backend or server side is because it is one language for the front end and one language for the backend and it is easy to debug, code and find any errors, because we are dealing with only one language.

4.2.3 SQL for primary database and Mongo dB for secondary database.

SQL or (structured query language) is a database language used to manipulate, store and retrieve data and create the data structures (tables) that hold the data. It is especially useful in manipulating structured or relational databases. The reason I chose SQL for the primary database is because it allows me to maintain relational integrity among relational data, in addition to that, I had related data entities in my class diagram.
Mongo dB is a NoSQL database. This means unlike SQL; it has no relational database structure that you have to follow. With mongo dB data is structured how you want it and any restrictions that you want to place on data can be placed in the code layer. I chose mongo dB as the secondary database in order to store the refresh-Tokens for the authenticated users on the dashboard. Additionally, I also stored the messages exchanged between admins and freelancers.

4.2.4 The Tools Used

The tools used for developing the entire system included Visual studio code (for coding), chrome web browser (For designing and debugging UI), (Heidi SQL) for designing and managing database, git and GitHub for version control.
Visual studio code is a lightweight text editor that helps in coding complex applications. It is lightweight and runs smoothly on lower end devices and provides stunning visuals and pretty-code-formatters to format the code you write in a neat and organized way.
I chose chrome web browser, because it has good developer tools to allow debugging and checking for responsiveness for multiple devices.
Heidi SQL is a database administration tool for MySQL, MariaDB etc. The UI is user friendly and it is also free to use.

4.3 System Cutover.

A system cutover is a process that involves going from the old way of doing thing (old system) to the new way of doing thing (new system). this new system in our case is the Tasked.it dashboard website created. There are many ways that this can be done. Some of them are mentioned below:

Immediate Cutover:
This method is exactly what It sounds like. We immediately put a stop to the old system and dismantle it and put into operation the new system simultaneously.
There is no transition period between the two actions. Instead, the organization may select a date and time to perform the termination of the old system and the activation of the new one.

Phased Cutover:
In this method, we perform the cutover in phases and we start to adopt the new system bit by bit. The risk is lower here as it is easy to simply find the problem in one particular phase.

Parallel Cutover:
The parallel cutover method will allow the old and new system to work and run alongside each other or in parallel. These will run for some time until the users decide that the new system operates successfully and then the old one will be terminated.

Pilot cutover:
This cutover method involves bringing the new system in only one part or component of the organization or company as a test. If the system runs successfully without any challenges, we then use one of the other cutover methods i.e., immediate, phased or parallel cutover to introduce the new system.

The cutover method most suitable for VTE would be Parallel cutover. They can still continue their old way of operating using paper-based records or excel records while also using the new system so that the new system starts to be understood and adopted by more employees faster. This is low risk as well because if any problem arises in the new system, they can immediately put a stop to it and wait for the troubleshooting to be finished, and the data in the old system will still remain unaffected.

4.4 Data Migration.

The process of data migration involves migrating or moving data from one place to another, one storage media or location to another, one format to another or one application to another. *(techtarget.com, 2022).* Below are some of the methods of data migration.

Database migration:
The act of migrating information from one or multiple original databases to one or multiple intended databases is known as database migration. *(cloud.google.com, no date)*

Cloud migration:
Cloud migration involves transferring a business's digital data, services, databases, IT infrastructures, and programs either partly or completely to the cloud. Additionally, moving from one cloud to another is also classified under cloud migration. *(accenture.com, no date)*

Storage migration:
Storage Migration is a process that allows the transfer of data that is stored in an old storage system to new volumes on a recently installed storage system.

Application migration:
Application migration involves relocating applications from one computational setting to another. This could involve moving applications from one data center to a different one, like from a public cloud to a private cloud, or from an organization's internal server to a cloud service provider*. (vmware.com, 2022).*

Migration method used:
The migration method used for the purpose of this project was database migration. Although VTE had no previous relational or even digital database, VTE did have paper-based records and excel spreadsheet records for freelancers and the staff that were working in VTE, additionally they also had records about the roles for freelancers that they offer, previous projects that they have completed and some clients that they worked with. All that had to be done was to manually add these records to the new systems database.

4.5 Training

Giving Instructions of the website's purpose and how it is to be used is known as training. In this instance, it requires instructing the website users, both freelancers and administrators, about its intended use. The way I managed to do this is by giving in person tutorials on how to use the system. This was done separately for admins and freelancers for security and privacy reasons. I also distributed video tutorials and user manuals to the end users in order to clarify any areas of difficulty in using the system.

4.6 Conclusion

In conclusion, what was discusses in this chapter provided a solid overview of the implementation step of the software development process. It covered many key areas such as the choice of programming languages, the tools used to develop the system, system cutover methods and which one was chosen, data migration methods and the method that was chosen for our case, and finally training users about the system. The upcoming will cover other project matters such as risk management, configuration management, testing and project management.

Chapter 5 – OTHER PROJECT ISSUES.

5.1 Introduction.

The following chapter will provide an in-depth discussion of some other project matter through the Agile DSDM methodology, risk and configuration management, and testing. It will particularly focus on black-box testing through unit and integration testing.

5.2 Project Management.

Project management involves the integration of specific skill sets, tools, knowledge, and methods to produce a desirable outcome. In this case, the Agile methodology was implemented. Agile methodology is a process of managing a project by dividing it into different parts.
Clear communication with all involved parties and continuous improvement at every stage is required. After starting the project, teams go through a cycle of preparation, execution, and reviewing.

Pre-Project
This part was done in the project proposal. This is the point before starting the project. At this point the problems were identified that the company is facing, an idea of a solution was come up and the mention of the aims and objectives etc. was done.

Feasibility
This phase is when we determine whether the project that is proposed appears to be feasible or doable from a technical perspective and whether or not it is cost effective from a business perspective. In our case, the project was considered feasible and doable, the project was about creating an admin dashboard style website for managing projects. The users to be focuses were VTE staff (admins) and VTE registered freelancers. This project was considered realistic and achievable from a technical and cost point of view.

Foundation
In this phase, the project objectives were aligned with the company's requirements and a set of achievable objectives were determined within the boundary of the project. The requirements were between functional and non-functional requirements and sorted out in order of priority with the MOSCOW technique. Furthermore, a Gantt chart showing the estimated time of the project was come up with and included in the proposal of the project.

Development
This phase is when things started getting into action and the development of the proposed system began iteratively. The time allocations given in the Gantt chart were followed and the components of the system were slowly starting to come together in incremental phases. Thorough testing and reviewing were done along the way.

Deployment
After achieving the given functional requirements and non-functional requirements of the system, the system was reviewed, tested if it meets the aims stated at the beginning of the project. If so, the system was ready for use and now could be cutover with the old system.

5.3 Risk management.

The process of risk management involves find out threats or risks to an organization, analyzing them and looking for ways or solutions on how to control or manage then once they arise (action plan). (techtarget.com, January, 2023). Risks are elements or factors that have a bad impact on the project. The risks have not yet been faced or experienced, but they do come apparent, they can be dealt with by checking the risk management document.

There are three approaches to risk management: **1. Avoidance, 2. Acceptance** and **3. mitigation**.

This projects risk management is given below:

Risk	**Impact**	**Likelihood**	**Action plan**
Losing all the project code files. This can be due to one of several reasons.	High	Medium	Look for any possible backups or maybe on GitHub if you have pushed and backed up there. If no backups are found, restart the project immediately without losing any more time.
Security issues found after project is deployed	High	Medium	Release a new version of the project and force users to update.
Spending too much time on documentation rather than actually building the system	Medium	Low	Allocate time properly and follow the Gannt chart to manage your time properly.
The PC used to build the project is not working but we have the project files backed up on external storage	Low	Low	Use someone else’s PC or borrow one from someone and continue working with the backed-up files in external storage.
The client is demanding new features even after all the requirements have been met.	Medium	Medium	Either choose to implement the new features or simply deny and show the documentation as the proof of agreement done prior and that they cannot break the agreement.
The project seems too overwhelming and developer starting to lose motivation or having imposter syndrome	High	Medium	Take a short break. Look for guidance from lecturers or online. Break up the tasks into micro tasks and start to do them one by one.

Table 3 – risk management table.

5.4 Configuration management

Software configuration management is an engineering process meant to trace and check alterations to configuration data of a software system. It is commonly used in the area of software development alongside version control and CI/CD infrastructure. *(atlassian.com, no date).*
What this simply means is that we track the changes we make to the overall system so that if the system fails or doesn't operate as desired after pushing a new change or feature, we can simply rollback to the previous change or commit.
For this reason, I used Git and GitHub as version control to track the changes I make to the system, the evidence for this can be found in the appendix.

5.5 Testing

Software testing involves analyzing the results from the system and comparing them to the outcomes that the developers intended when building the system. This helps to guarantee that the system is performing properly and that the desired outcomes of the project are achieved. Testing was done at several steps in the development process to determine if each component of the system is functioning as planned and to detect any defects in a particular area. *(ibm.com, no date).*

Usually the core techniques used are:

Whitebox testing.
White box testing is a type of software testing that gives the individual performing the test to have total understanding of the program being tested, as well as access to its source code and design documents. This allows white box testing to pinpoint issues that both gray and black box testing methods would not detect. However, for the sake of this project this was outside the scope of the project. The only type of testing to perform would be black box testing.

Blackbox testing.
This type of testing does not require the tester to known anything at all about the internal workings of the system. All that has to be done is to come up with a few inputs to test with and the expected outputs. If after testing the outputs matches the expected output, then the test passed otherwise failed. Now this type of testing is what we will use to test the system developed.
This type of testing can be separated into Integration testing and unit testing:

Unit testing.
Unit testing is when we test the smallest parts or components of the system rather than the whole system at large.

Integration testing.
This is quite the opposite of unit testing and is when we test if all or most of the components as a whole work together as expected or there are defects.

Both of these types of testing (integration testing and unit testing) done for the Tasked.it system can be found in the appendix.

5.6 Conclusion

To conclude, this chapter covered the project management aspect of this project, risk management and also illustrated a risk management table for a few risks that are likely and what to do if we face them. Lastly, it covered configuration management and the testing techniques used for testing the system.

Chapter 6 – CONCLUSION.

6.1 Introduction

This part of the project aims to assess the efficiency and success of it. We will discuss how well the project met its goals, how any issues that arose were resolved, how similarly various tasks should be performed and conducted in the future, and what features will be implemented or improvements will be made in the near future.

6.2 Achievements of aims and objectives

Thankfully, all if not most of the aims that were planned for this project have been met. The aim was to create a multipurpose flexible admin dashboard style web application that allowed VTE to manage projects and freelancers easily which would boost the number of projects to be taken by VTE in a certain period of time ultimately boosting the revenue that VTE produces.
All of the objectives have also been met as well except that the responsiveness of the website i.e., how it looks on mobile devices just needs to be a bit more polished.

6.3 Problems encountered and how they were overcome

The problems encountered were during different stages of the project.

Starting with the analysis phase, gathering the functional requirements was found a bit difficult as employees were not always giving the right answers or they were too busy to respond. This was overcome by asking the CEO to convince the employees to spare some time every other day for answering the questions, being interviewed or just reviewing the system so far.

Moving on the design phase, I found it challenging to come up with the behavioral model overall, this was overcome by doing more research and looking for tutorials on how to create one. Furthermore, there were two user types in my system i.e., Admin and freelancer. I did not know how to illustrate that on the model. This was overcome by simply creating two models (one for freelancer and one for admin).

Lastly during the implementation phase there were several problems encountered including the following:

Firstly, the any type of user admin or freelancer was unable to login even when their credentials were correct. The problem was occurring because the backend was comparing the password in plain text with the hashed password in the database. This was resolved simply by hashing the password in plain text before comparing it to the one in the database.
Secondly, the database was throwing errors when a user's account was deleted. The problem here was occurring because the foreign key field in the other tables for username was not existing after being deleted. This was resolved by simply adding an ON DELETE SET NULL constraint to foreign keys.

And finally, the refresh-token mechanism works in a way that a request is sent to the backend to ask for a new access-token every x number of minutes. In my case I did not know how to run a function every x number of minutes in react.js. This was resolved by using the use-Effect hook to run a set-Timeout function every 600000 milliseconds or 10 minutes to send a request to the /refresh endpoint to request for a new access-token.

6.4 Actions that would be performed differently

If there Is something that I would perform differently, it would definitely be to use lesser technologies or programming languages and frameworks when building a project unless its really necessary to do so or unless I really know how to use them properly or I have researched extensively on them. This is because by using more technologies, languages and frameworks the project just becomes unnecessarily complicated. One more thing I would do differently is to use different requirement gathering methods.

6.5 Future development

Some new features or improvements that can be added to the system would include verifying the users email address using an OTP (One time password) when registering and logging in for better security.
Secondly, integrating a payment system to allow admins to make payments from the directly from the dashboard to freelancer's accounts.
Finally, also allowing clients to log in to check the progress that has been made on their project so far.

Chapter 7 – REFLECTIVE PRACTICES.

The reflective practices will be done in accordance with the Rolfe, G. Freshwater, D. and Jasper, M. (2001) model.

7.1 Description.

The following project included the development of an admin dashboard style website for Visionary Technologies Enterprise (VTE). The dashboard was named Tasked.it. This project was created in order to allow VTE to improve their current way of working with digital projects and freelancers that they hire. The tactics used to develop the system included researching on similar systems, analyzing the business processes of VTE and eventually gathering the necessary requirements, designing how the system will look and operate, and finally implementing the system. The project also involved documenting all of the steps of the development process.

7.2 Analysis.

Throughout the project, all efforts made were aimed towards the original aims and objectives. I was already familiar with the technologies used in the project, however there were some that I learnt along the way by researching and watching videos. Though I felt pressure due to the amount of work necessary to complete it, I should have done more research on the necessary technologies before starting. This would have saved me time as there was research that had to be done during the project instead of just building the system. Some 'could have' features were left out and would be great if they were added. The design and responsiveness of the system could be improved on. Lastly, The Security of the system was although strong overall, but was lacking in some areas and could be improved on.

7.3 Action Plan.

My action plan would be to not just familiarize myself with the technologies that I excel in, but to master them top to bottom. Another thing is that I will only use technologies that I fully know and am comfortable working with to produce better results in the final working system. I will allocate time properly next time to not get overwhelmed by the amount of work and I will not hold back some really good features that could have been added to the overall system. I would improve on the design and responsiveness of the system. Finally, I will not put a blind eye on the security of the system and thoroughly penetration test everything.

8. References

What is agile methodology? (October 15th, 2022) | asana Available on: https://asana.com/resources/agile-methodology Accessed on: 18/04/2023

Object oriented methodology (no date) | ogcio.gov.hk Available on: https://www.ogcio.gov.hk/en/our_work/infrastructure/methodology/system_development/past_documents/oom/#:~:text=Object%20Oriented%20Methodology%20(OOM)%20is,re%2Duse%20of%20software%20components. Accessed on: 18/04/2023

Requirements gathering methods (17th January 2020) | Tutorialspoint.com Available on: https://www.tutorialspoint.com/business_analysis/business_analysis_requirement_gathering_techniques.htm Accessed on: 18/04/2023

Project Requirement Gathering: Surveys and Questionnaires (no date) | pmmajik.com Available on: https://www.pmmajik.com/project-requirement-gathering-surveys-and-questionnaires/ Accessed on: 19/04/2023

Systems architecture (July, 2012) | Wikipedia available on: https://en.wikipedia.org/wiki/Systems_architecture#:~:text=A%20system%20architecture%20is%20the,and%20behaviors%20of%20the%20system. Accessed on: 20/04/2023

Structural model (august 2022) | IBM available on: https://www.ibm.com/docs/en/elms/esdr/8.4.0?topic=rhapsody-structural-model Accessed on: 20/04/2023

Behavioral model (June 2020) | geeksforgeeks.com available on: https://www.geeksforgeeks.org/short-note-on-behavioral-model/ Accessed on: 21/04/2023

What is a programming language (no date) | computer hope available on: https://www.computerhope.com/jargon/p/programming-language.htm Accessed on: 23/04/2023

What is data migration (2022) | tech target available on: https://www.techtarget.com/searchstorage/definition/data-migration Accessed on: 23/04/2023

What is risk management and why is it important (no date) tech target available on: https://www.techtarget.com/searchsecurity/definition/What-is-risk-management-and-why-is-it-important Accessed on: 25/04/2023

Software testing (2021) | IBM available on: https://www.ibm.com/cloud/devops/software-testing Accessed on: 26/04/2023

Configuration management (no date) | Atlassian available on: https://www.atlassian.com/microservices/microservices-architecture/configuration-management Accessed on: 27/04/2023

9. Appendix

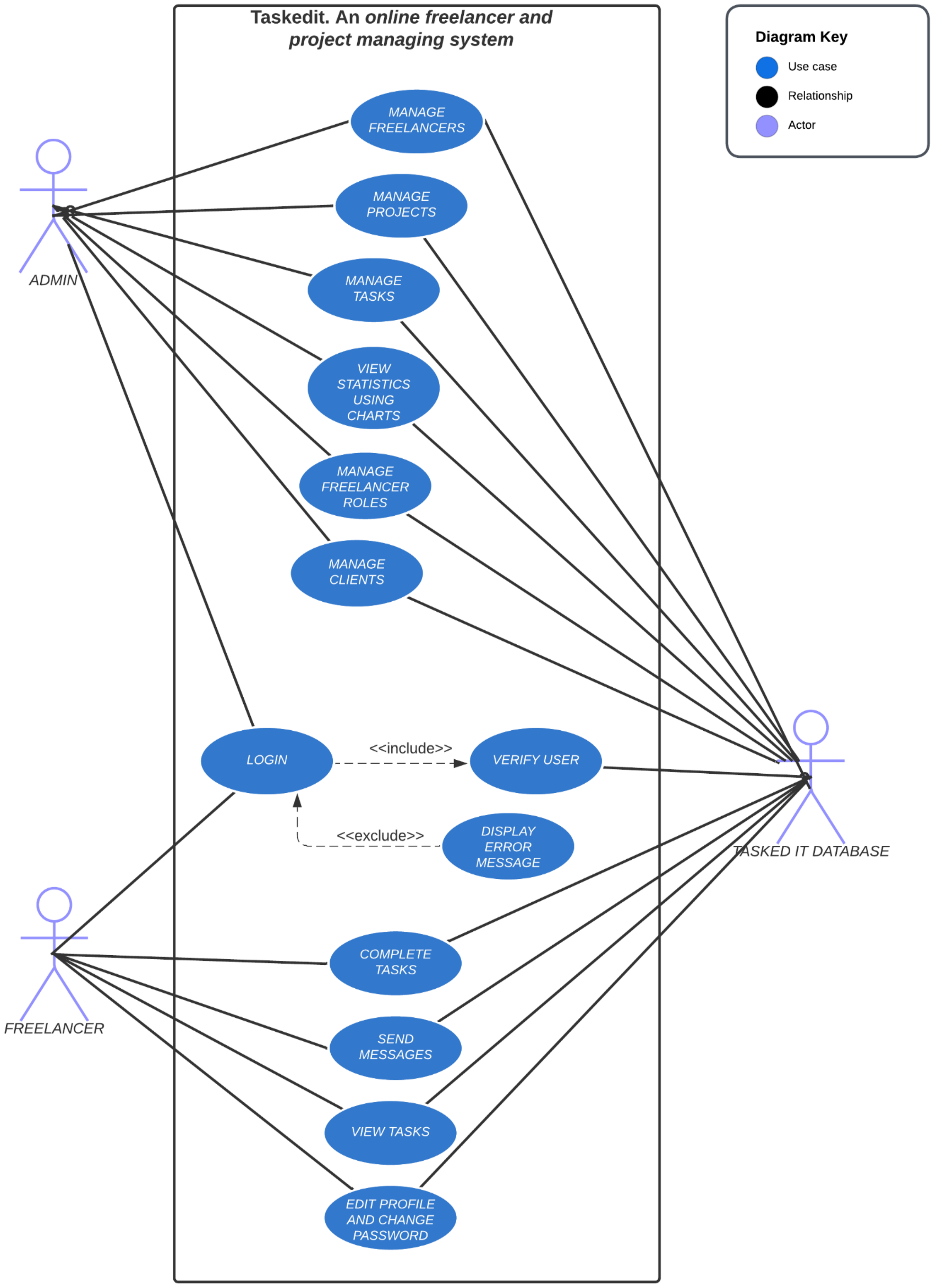
9.1 Requirements catalogue

Requirement ID	Name	Description	Ranking	Type
1	Login functionality	Login pages for admins and freelancers and authentication mechanism to differentiate privilege levels.	MUST HAVE	Functional
2	View list of freelancers	Admins should be able to view the list of freelancers registered in the system	SHOULD HAVE	Functional
3	View list of admins	Admins should be able to view the list of other admins registered in the system	SHOULD HAVE	Functional
4	Edit and remove freelancers	Admins should have a feature to edit and remove the list of freelancers	SHOULD HAVE	Functional
5	Edit and remove Admins	Admins can't edit and remove admins from the list of admins	WON'T HAVE	Functional
6	Message freelancer	Admins and freelancers can exchange messages	COULD HAVE	Functional
7	Message admin	Admins and other admins can exchange messages	COULD HAVE	Functional
8	List Clients	Admins should be able to see list of clients	SHOULD HAVE	Functional
9	Add new clients and freelancers	Admins should be able to add new clients, and register new freelancers	MUST HAVE	Functional

10	Edit and Remove clients	Admins should have a feature to edit and remove the list of clients	SHOULD HAVE	Functional
11	Create new projects	Admins should be able to create new projects	MUST HAVE	Functional
12	Edit and Remove projects	Admins should be able to edit and remove the list of clients	SHOULD HAVE	Functional
13	View Tasks	Admins should be able to view all tasks	SHOULD HAVE	Functional
14	Edit and remove tasks	Admins should have a feature to edit and remove the list of tasks	SHOULD HAVE	Functional
15	Create tasks	Admins should be able to create and assign tasks	MUST HAVE	Functional
16	List projects	Admins should be able to view all projects	SHOULD HAVE	Functional
17	Show analytics and statistics	A high-level statistics page should be there showing data using graphs charts etc.	COULD HAVE	Functional
18	Editing profile for both admins and freelancers	Admins and freelancers can edit their profiles	COULD HAVE	Functional
19	Change password for both admins and freelancers	Admins and freelancers can change passwords	COULD HAVE	Functional
20	Mark as complete	Freelancers can mark task as complete if completed	SHOULD HAVE	Functional

TABLE 4 – REQUIREMENTS CATALOGUE

9.2 Use case descriptions using diagram.



Freelancer and admin

Login – freelancers and admins should be able to log in using their credentials.

Edit profile – freelancers should be able to edit their account details.

Change password – they should also be able to change their password.

Freelancer -

Complete tasks – freelancers should be able to mark tasks as complete.

Send messages – freelancers should be able to exchange messages with admins

View tasks – freelancers should be able to view the tasks they are assigned to do.

Admin –

Manage freelancers – admins should be able to add edit and remove freelancers.

Manage clients – admins should be able to add edit and remove clients.

Manage projects – admins should be able to add edit and remove projects.

Manage tasks – admins should be able to add edit and remove tasks.

Manage freelancers’ roles – admins should be able to add edit and remove freelancer roles.

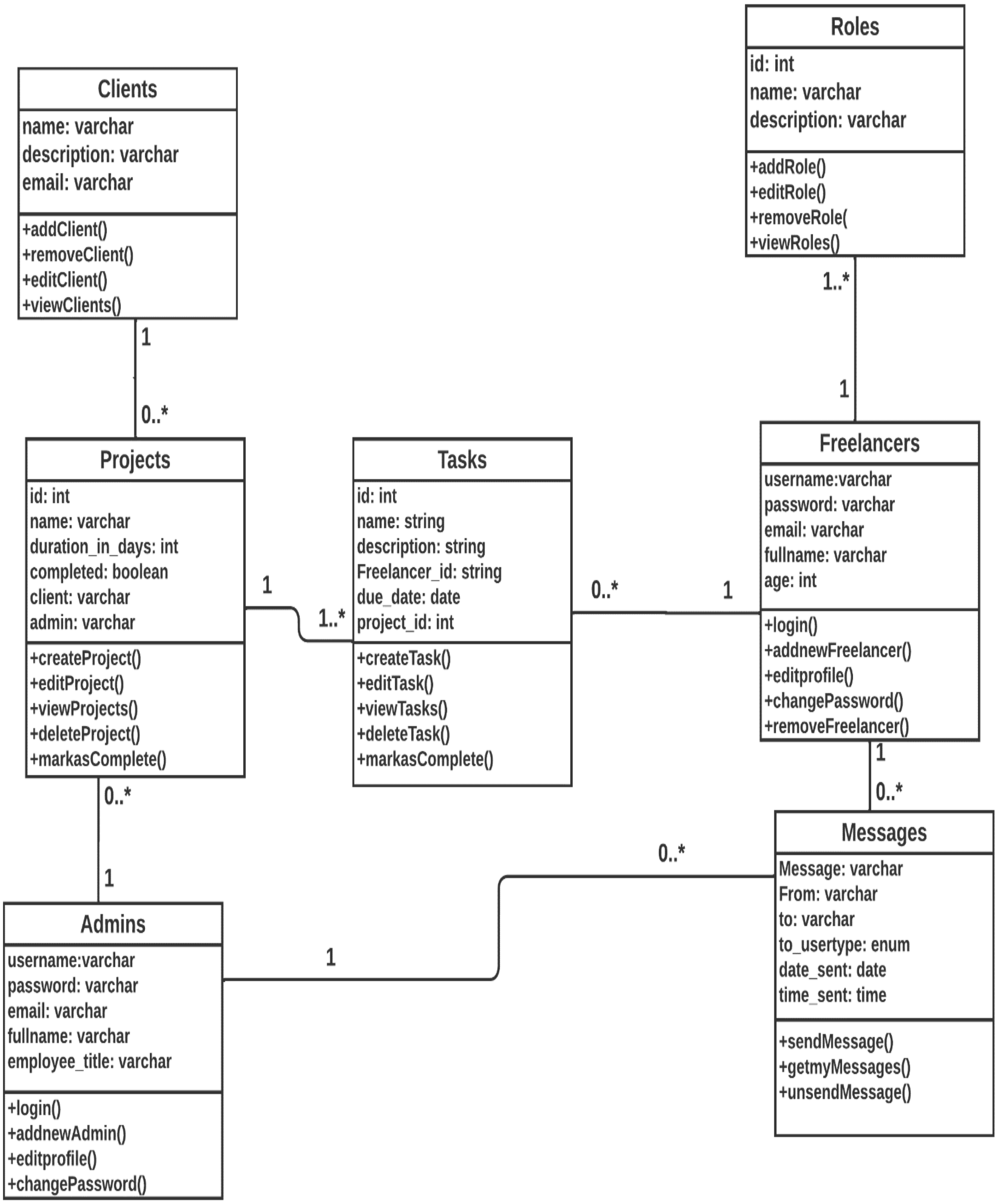
View statistics – should be able to view statistics on home page using charts etc.

Tasked.it database-

Verify credentials – check if the credentials provided to log in exists in the database and authenticate user accordingly

Display error message – throw an error if the credentials are invalid.

9.3 Detailed class definitions.



Admins class

This class stores the admins details. It has methods or functions to log in add a new admin, edit profile and change password.

Freelancers class

This class stores the freelancers' details and has methods or functions to log in, add new freelancer, edit profile, change password and remove freelancer.

Messages class

This class stores messages exchanged between admins and freelancers and has methods or functions that allow to send a message, get messages and unsend a message.

Tasks class

This class stores the details about tasks and has methods that allow it to create, edit, view, delete and mark a task as complete.

Projects class

This class stores the details about projects and has methods that allow it to create, edit, view, delete and mark a project as complete.

Roles class

This class stores the details about freelancer roles and has methods that allow it to create, edit, view and delete roles.

Clients class

This class stores the details about clients and has methods that allow it to create, edit, view and delete clients.

9.4 Testing scripts

Starting with unit tests:

Unit test ID	Test class	Test function	Description	Expected result	Actual result
1	Admin class	Login	Login with credentials Username: slide Password: slide123	Credentials are correct and redirected to home page	Credentials are correct and redirected to home page
2	Admin class	Add new admin	Create a new admin with following Credentials Username: admin1 Password: admin1 Employee_title: sales rep Email: admin1@gmail.com Fullname: shihab mirza	Admin added to database	Admin added to database
3	Admin class	Edit profile	Edit profile and change full name and email for current account	Profile editing successful	Profile editing successful
4	Admin class	Change password	Change password to a new and unique one	Password change successful	Password change successful
5	Freelancers class	Login	Login with credentials Username: slide_freelancer Password: slide_freelancer123	Credentials are correct and redirected to home page	Credentials are correct and redirected to home page
6	Freelancers class	Add new freelancer	Create a new freelancer with following Credentials Username: slide_freelancer Password: slide_freelancer123 Employee_title: sales rep Email: admin1@gmail.com Fullname: shihab mirza	freelancer added to database	freelancer added to database
7	Freelancers class	Edit profile	Edit profile and change full name and email for current account	Profile editing successful	Profile editing successful
8	Freelancers class	Change password	Change password to a new and unique one	Password change successful	Password change successful
9	Freelancers class	Remove freelancer	Remove freelancer with username slide_freelancer	Freelancer removed	Freelancer removed
10	Messages class	Send message	Send message to slide_freelancer	Message sent	Message sent

11	Messages class	Unsend message	unSend message to slide_freelancer	Message unsent	Message unsent
12	Messages class	View my messages	View all messages I have	Messages viewed	Messages viewed
14	Projects class	Create project	Project created with some dummy details	Creation should work	Creation worked
15	Projects class	Edit project	Project edited with some dummy details	Editing should work	editing worked
16	Projects class	Remove project	A random project is deleted	Project should be deleted	Project is deleted
17	Projects class	Mark as complete	Mark a random project as complete from the list	Project should be marked as complete	Project is marked as complete
18	tasks class	Create task	task created with some dummy details	Creation should work	Creation worked
19	tasks class	Edit task	task edited with some dummy details	Editing should work	editing worked
20	tasks class	Remove task	A random task is deleted	task should be deleted	task is deleted
21	tasks class	Mark as complete	Mark a random task as complete from the list	task should be marked as complete	task is marked as complete
22	Clients class	Create client	client created with some dummy details	Creation should work	Creation worked
23	Clients class	Edit client	client edited with some dummy details	Editing should work	editing worked
24	Clients class	Remove client	A random client is deleted	client should be deleted	client is deleted
25	Roles class	Create role	role created with some dummy details	Creation should work	Creation worked
26	Roles class	Edit role	role edited with some dummy details	Editing should work	editing worked
27	Roles class	Remove role	A random role is deleted	role should be deleted	role is deleted

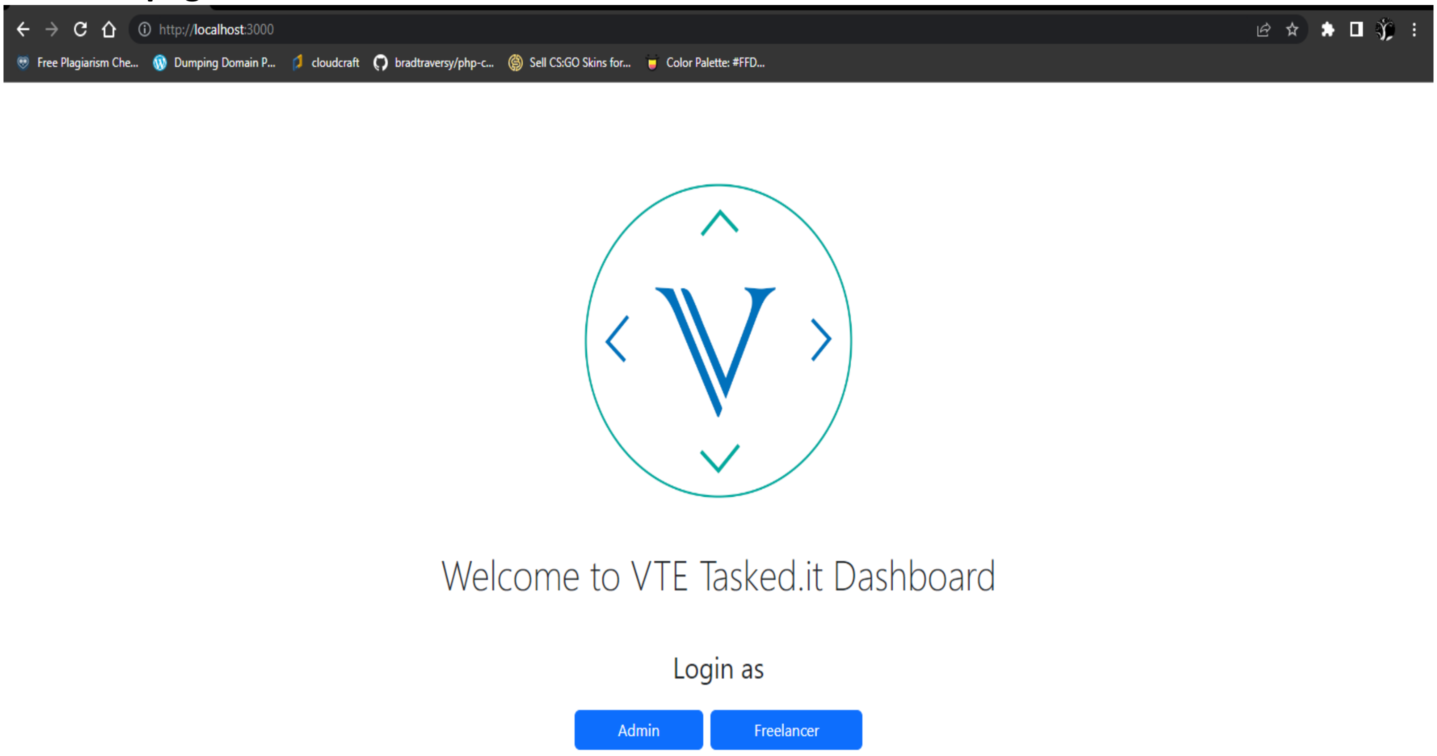
Integration tests:

This is when we test the basic functionality for all the classes individually. And check if all the methods in the class work together properly.

Test ID	Test class	Expected result	Actual result
1	Admin class	All functions should work and handle error properly	All functions work and handle errors properly.
2	Freelancer class	All functions should work and handle error properly	All functions work and handle errors properly
3	Messages class	All functions should work and handle error properly	All functions work and handle errors properly
4	Roles class	All functions should work and handle error properly	All functions work and handle errors properly
5	Projects class	All functions should work and handle error properly	All functions work and handle errors properly except that the marking as complete takes time to reflect some times.
6	Tasks class	All functions should work and handle error properly	All functions work and handle errors properly except that the marking as complete takes time to reflect some times.

9.5 Evidence of Testing using screenshots.

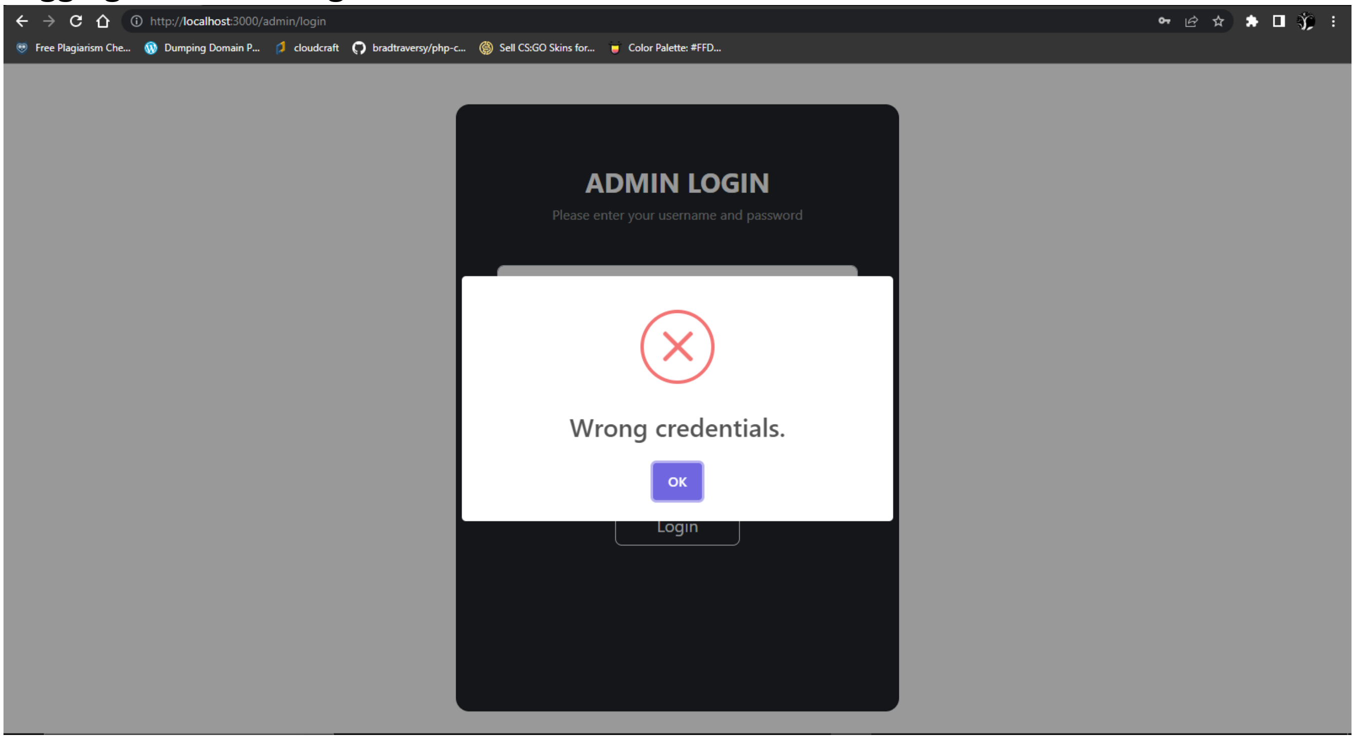
VTE root page.



Welcome to VTE Tasked.it Dashboard

Login as

Logging in with wrong credentials:

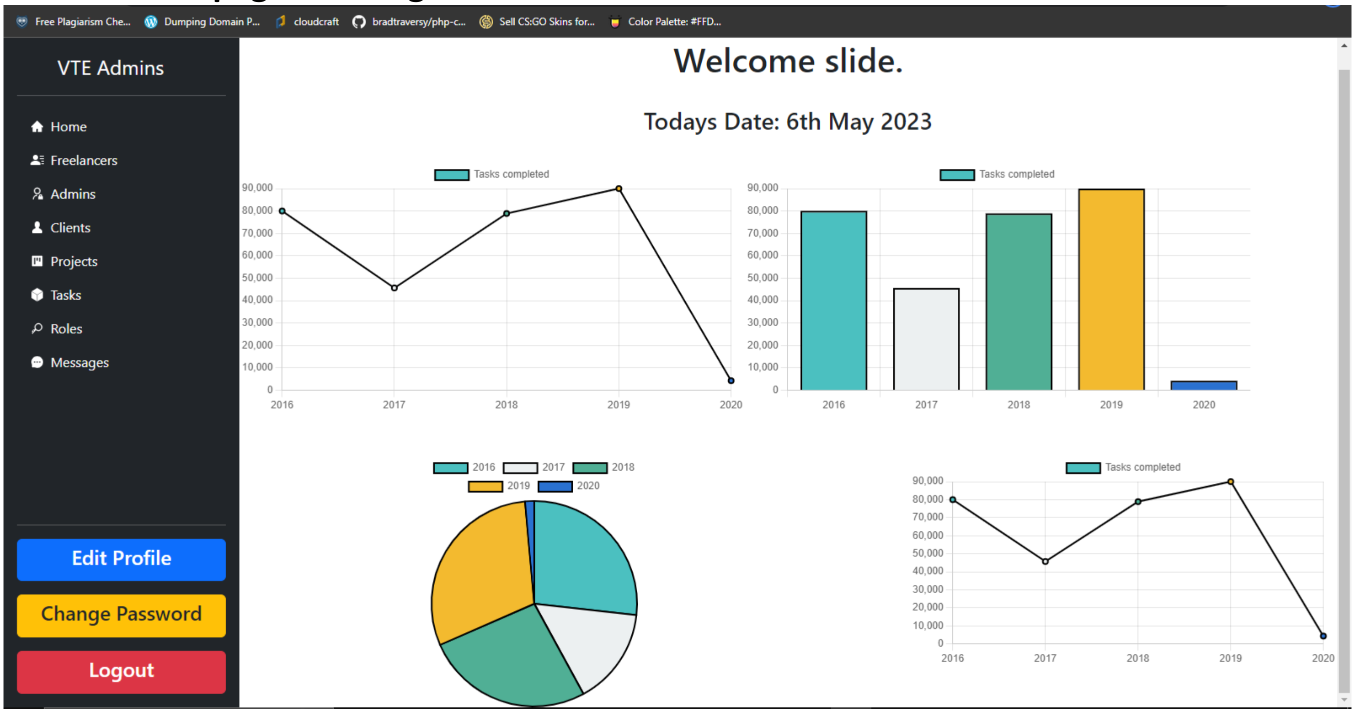


ADMIN LOGIN

Please enter your username and password

Wrong credentials.

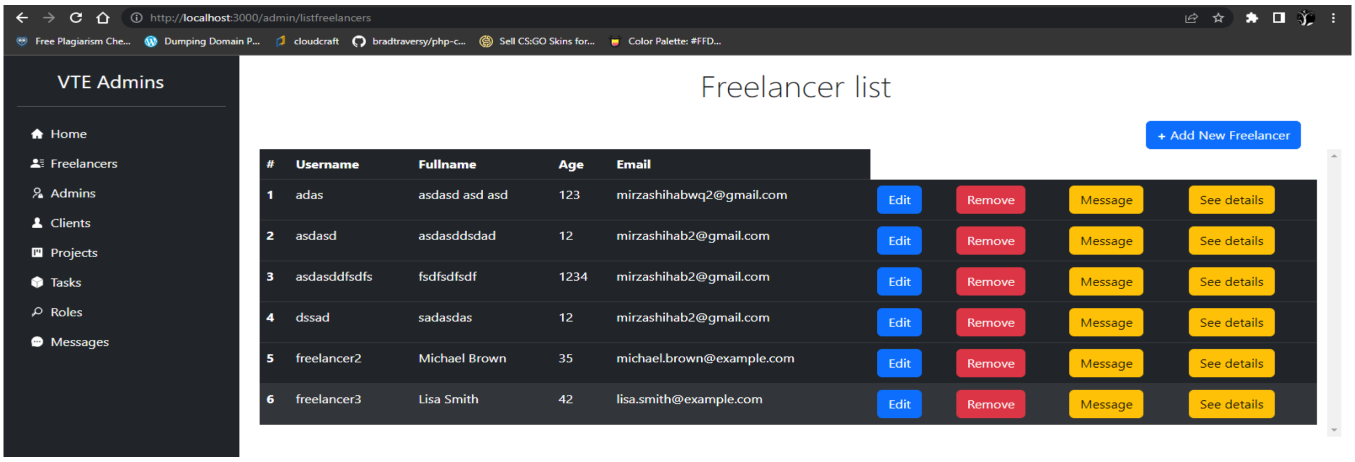
Admin home page showing statistics



Welcome slide.

Todays Date: 6th May 2023

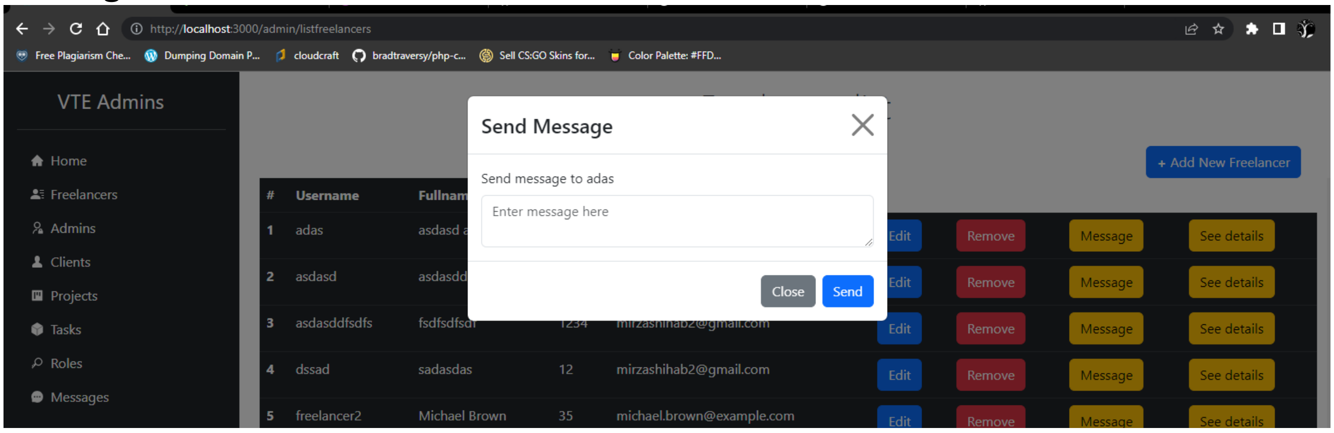
List of freelancers:



Freelancer list

#	Username	Fullname	Age	Email				
1	adas	asdasd asd asd	123	mirzashihabwq2@gmail.com	Edit	Remove	Message	See details
2	asdasd	asdasddsdad	12	mirzashihab2@gmail.com	Edit	Remove	Message	See details
3	asdasddfsdfs	fsdfsdfsdf	1234	mirzashihab2@gmail.com	Edit	Remove	Message	See details
4	dssad	sadasdas	12	mirzashihab2@gmail.com	Edit	Remove	Message	See details
5	freelancer2	Michael Brown	35	michael.brown@example.com	Edit	Remove	Message	See details
6	freelancer3	Lisa Smith	42	lisa.smith@example.com	Edit	Remove	Message	See details

Message freelancer

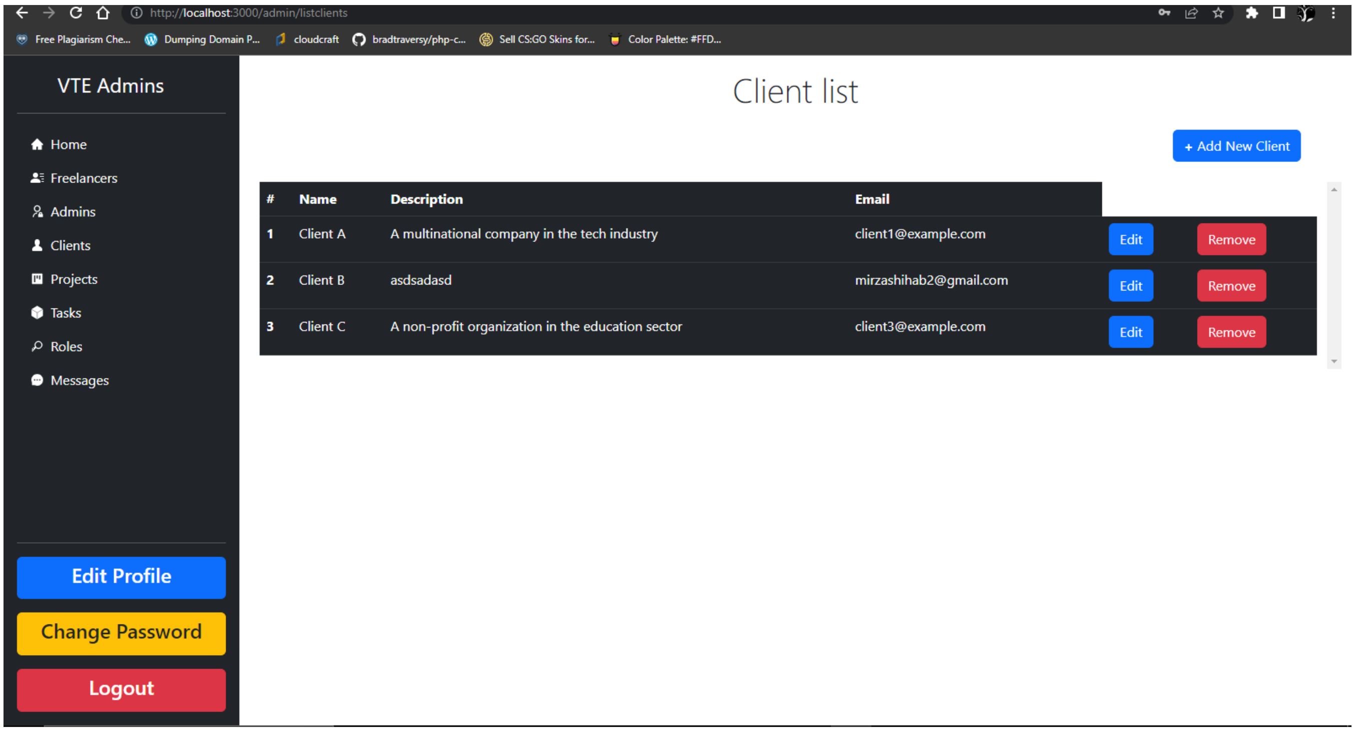


Send Message

Send message to adas

Enter message here

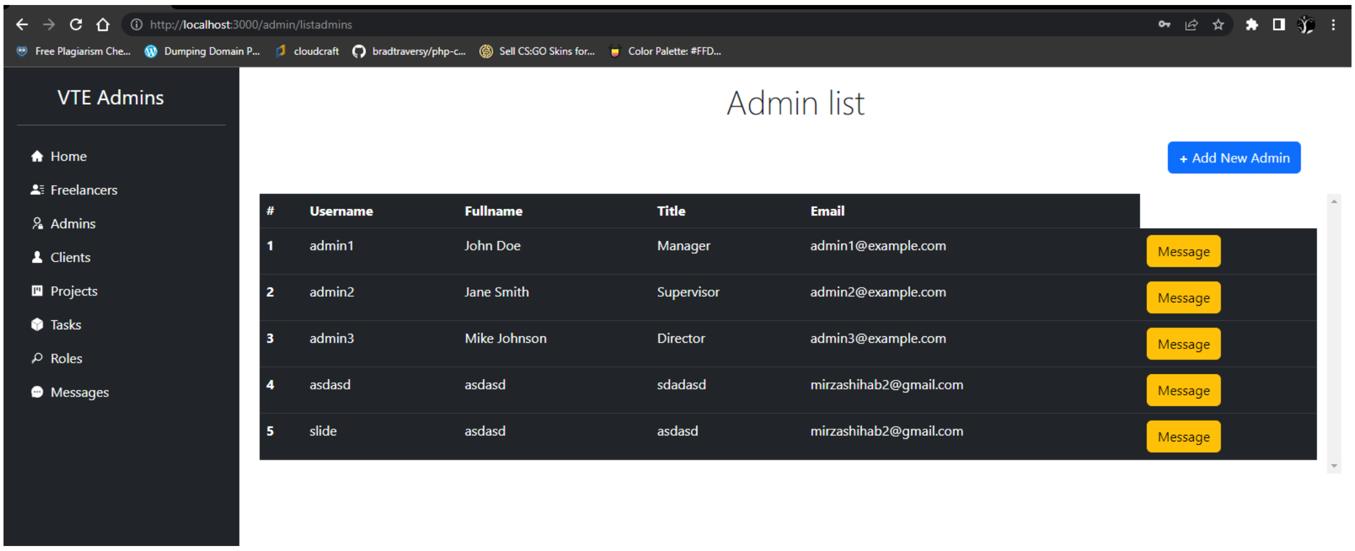
Clients list



Client list

#	Name	Description	Email		
1	Client A	A multinational company in the tech industry	client1@example.com	Edit	Remove
2	Client B	asdsadasd	mirzashihab2@gmail.com	Edit	Remove
3	Client C	A non-profit organization in the education sector	client3@example.com	Edit	Remove

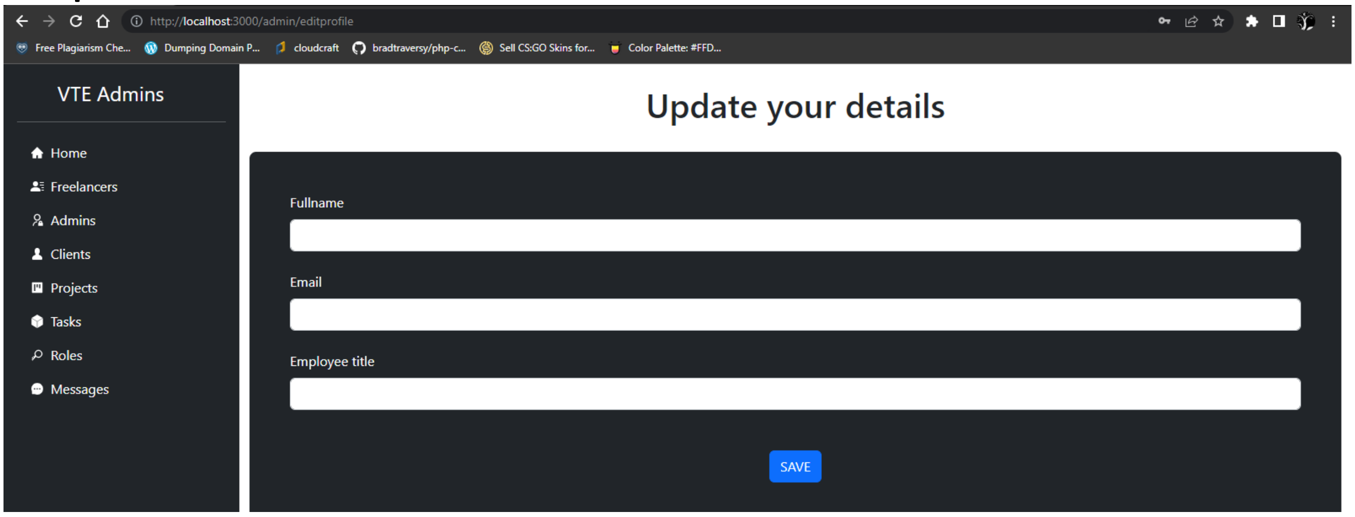
Admin list



Admin list

#	Username	Fullname	Title	Email	
1	admin1	John Doe	Manager	admin1@example.com	Message
2	admin2	Jane Smith	Supervisor	admin2@example.com	Message
3	admin3	Mike Johnson	Director	admin3@example.com	Message
4	asdasd	asdasd	sdadasd	mirzashihab2@gmail.com	Message
5	slide	asdasd	asdasd	mirzashihab2@gmail.com	Message

Edit profile



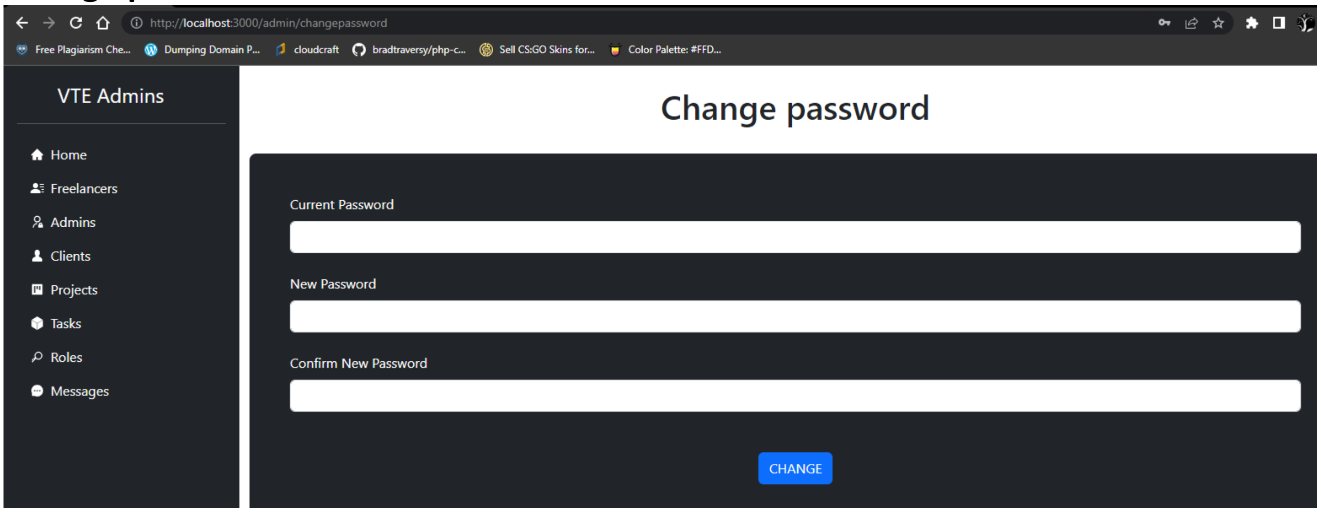
Update your details

Fullname

Email

Employee title

Change password


http://localhost:3000/admin/changepassword

Change password

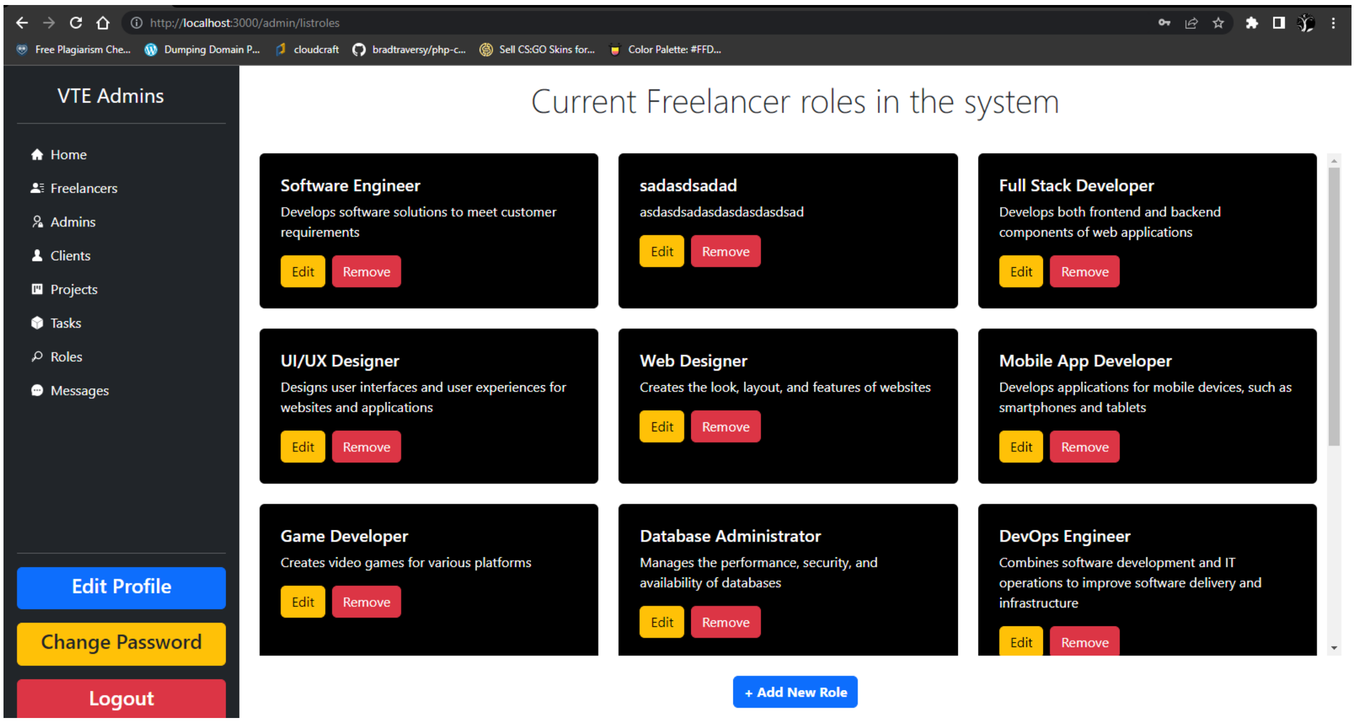
Current Password

New Password

Confirm New Password

CHANGE

List of roles


http://localhost:3000/admin/listroles

Current Freelancer roles in the system

Software Engineer
Develops software solutions to meet customer requirements
Edit Remove

sadasdsadad
asdasdsadasdasdasdsad
Edit Remove

Full Stack Developer
Develops both frontend and backend components of web applications
Edit Remove

UI/UX Designer
Designs user interfaces and user experiences for websites and applications
Edit Remove

Web Designer
Creates the look, layout, and features of websites
Edit Remove

Mobile App Developer
Develops applications for mobile devices, such as smartphones and tablets
Edit Remove

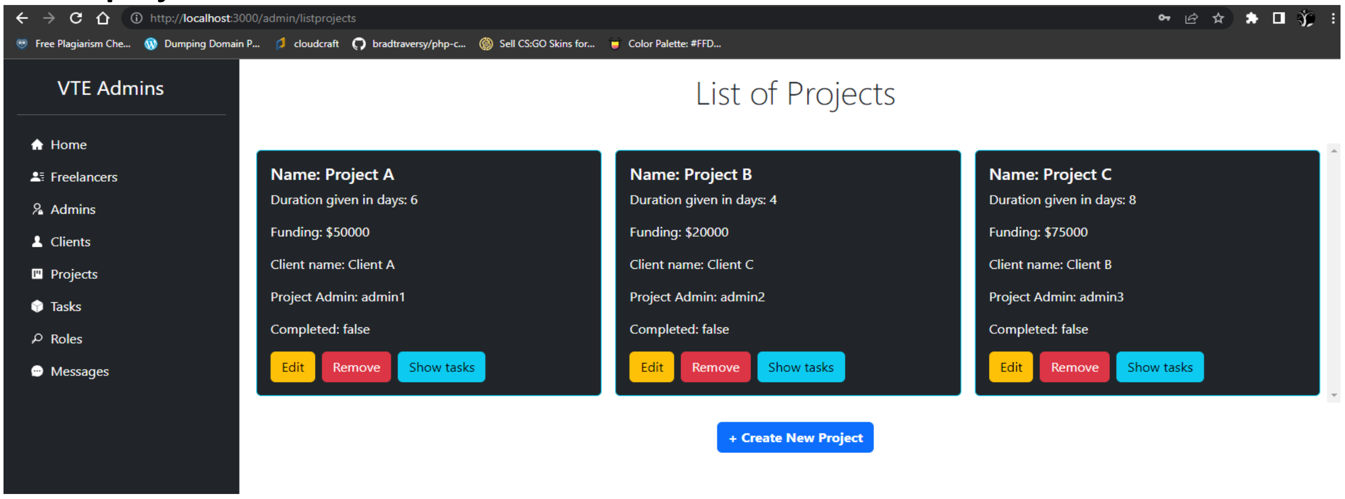
Game Developer
Creates video games for various platforms
Edit Remove

Database Administrator
Manages the performance, security, and availability of databases
Edit Remove

DevOps Engineer
Combines software development and IT operations to improve software delivery and infrastructure
Edit Remove

+ Add New Role

List of projects


http://localhost:3000/admin/listprojects

List of Projects

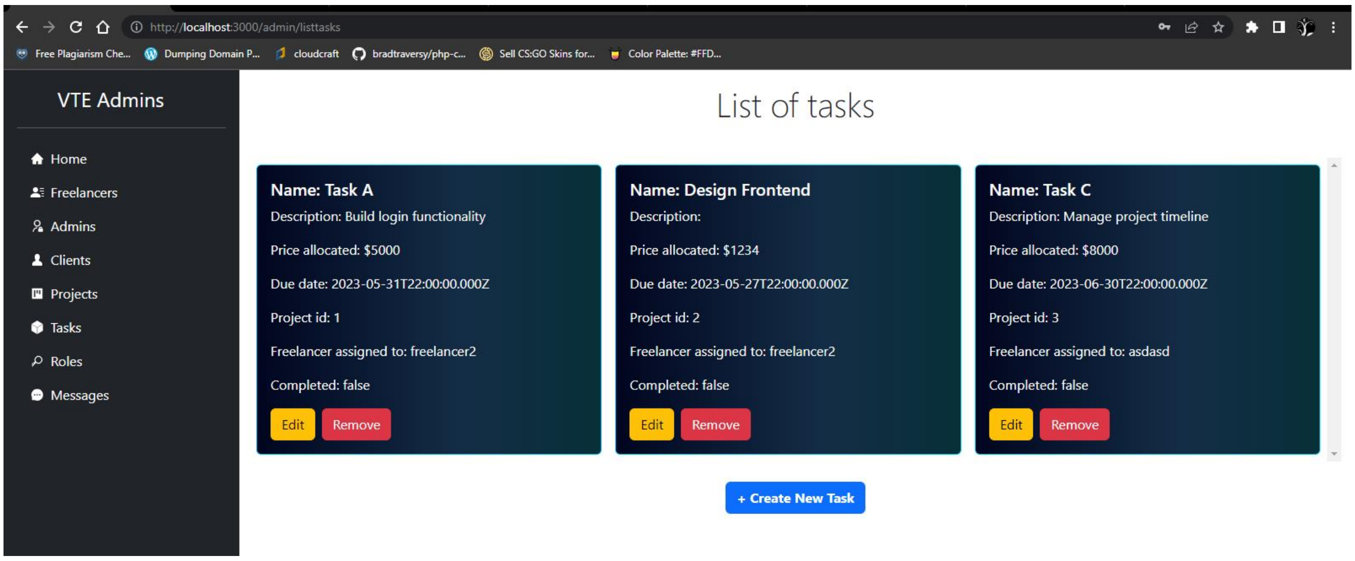
Name: Project A
Duration given in days: 6
Funding: $50000
Client name: Client A
Project Admin: admin1
Completed: false
Edit Remove Show tasks

Name: Project B
Duration given in days: 4
Funding: $20000
Client name: Client C
Project Admin: admin2
Completed: false
Edit Remove Show tasks

Name: Project C
Duration given in days: 8
Funding: $75000
Client name: Client B
Project Admin: admin3
Completed: false
Edit Remove Show tasks

+ Create New Project

List of tasks


http://localhost:3000/admin/listtasks

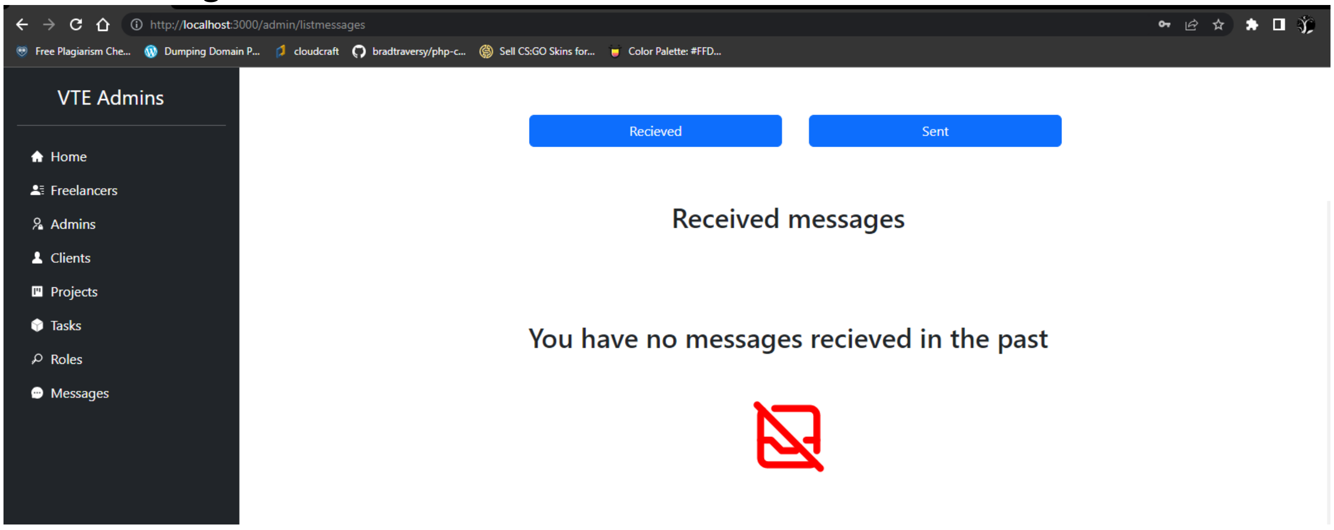
List of tasks

Name: Task A
Description: Build login functionality
Price allocated: $5000
Due date: 2023-05-31T22:00:00.000Z
Project id: 1
Freelancer assigned to: freelancer2
Completed: false

Name: Design Frontend
Description:
Price allocated: $1234
Due date: 2023-05-27T22:00:00.000Z
Project id: 2
Freelancer assigned to: freelancer2
Completed: false

Name: Task C
Description: Manage project timeline
Price allocated: $8000
Due date: 2023-06-30T22:00:00.000Z
Project id: 3
Freelancer assigned to: asdasd
Completed: false

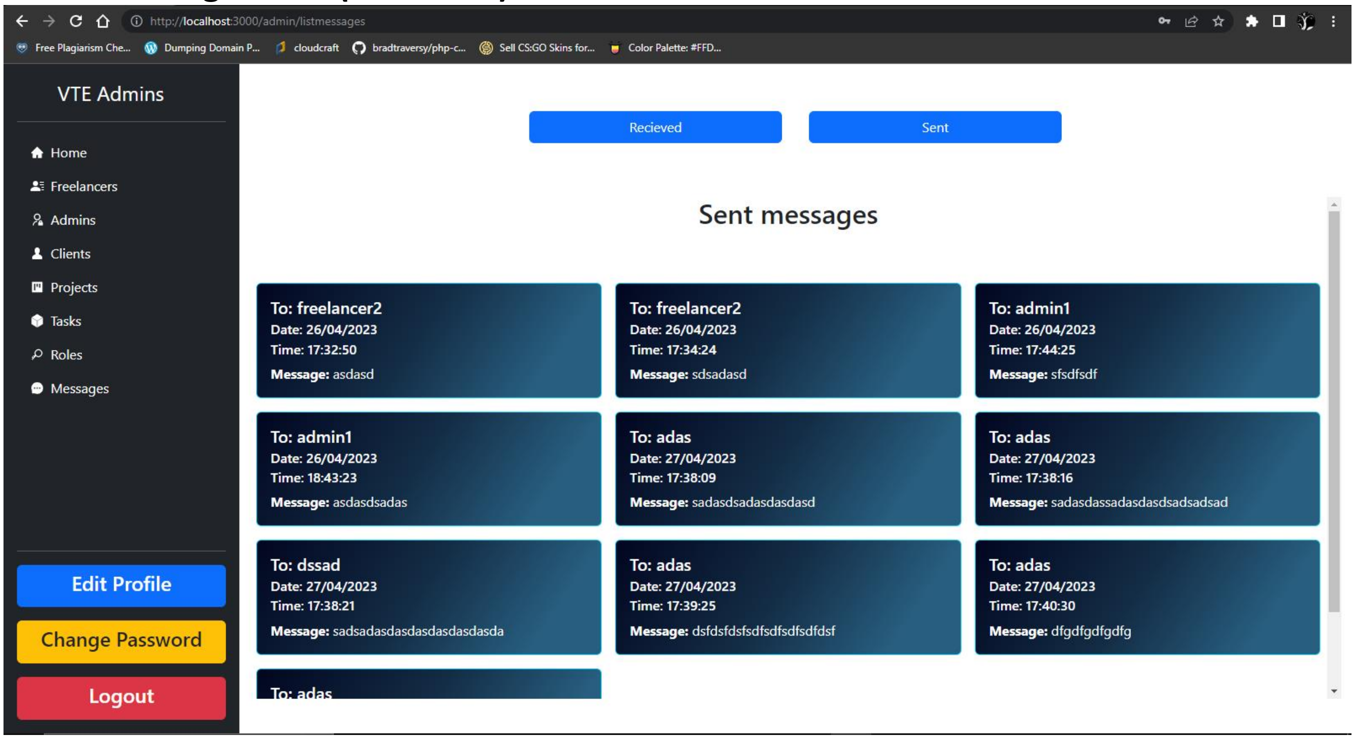
List of messages received


http://localhost:3000/admin/listmessages

Received messages

You have no messages recieved in the past

List of messages sent (scrollable)


http://localhost:3000/admin/listmessages

Sent messages

To: freelancer2
Date: 26/04/2023
Time: 17:32:50
Message: asdasd

To: freelancer2
Date: 26/04/2023
Time: 17:34:24
Message: sdsadasd

To: admin1
Date: 26/04/2023
Time: 17:44:25
Message: sfsdfsdf

To: admin1
Date: 26/04/2023
Time: 18:43:23
Message: asdasdsadas

To: adas
Date: 27/04/2023
Time: 17:38:09
Message: sadasdsadasdasdasd

To: adas
Date: 27/04/2023
Time: 17:38:16
Message: sadasdassadasdasdsadsadsad

To: dssad
Date: 27/04/2023
Time: 17:38:21
Message: sadsadasdasdasdasdasdasda

To: adas
Date: 27/04/2023
Time: 17:39:25
Message: dsfdsfdsfsdfsdfsdfsdfdsf

To: adas
Date: 27/04/2023
Time: 17:40:30
Message: dfgdfgdfgdfg

To: adas

9.6 User guide

9.6.1 installation guide (how to set up the system)

Before we start make sure you have node.js version 16.18.1 installed on your machine as well as mongo db.

1. download and extract the folder from the zip file
2. it would be best to open the folder in visual studio code
3. there will be two directories i.e., client directory and server directory.
4. Open a terminal or command prompt in the same folder
5. Change directory to client using the command “cd client”.
6. Next run the following command “npm install”.
7. After the packages are installed, run “npm run dev”
8. Leave the current terminal window open and open a new terminal window.
9. Run “mongod” in the new terminal window. Open one more new terminal window.
10. And finally go back to the main directory by running “cd ..”
11. Now run “cd server”.
12. Run “npm install”.
13. and finally run “node server.js”
14. now the client and server are running and you can go to http://localhost:3000 to access the website.

9.6.1 Admin guide.

First page:
The first page you will see will be asking you to choose if you’re a freelancer or admin Choose admin and it will take you to log in page

Log in page:
Enter you credentials here. For the sake of testing, you can use these credentials:

username: slide   password: slide123

home page:
Here you will see a general overview and some statistics about the data in the system.

Freelancers page:
Here you will see a list of the current freelancer that are in the system and you will have options to edit remove and message each one. You will also have a button that allows you to add a new one.

Client’s page:
Here you will see a list of the current clients that are in the system and you will have options to edit remove and message each one. You will also have a button that allows you to add a new one.

Admins page:
Here you will see a list of the current admins that are in the system and you will have the option to message each one. You will also have a button that allows you to add a new one using an admin key.

Freelancer Role's page:
Here you will see a list of the current freelancer roles that are in the system and you will have options to edit and remove each one. You will also have a button that allows you to add a new one.

Projects page:
Here you will see a list of the projects that are in the system and you will have options to edit remove and list the tasks for each one. You will also have a button that allows you to add a new one or mark one as complete.

Tasks page:
Here you will see a list of the tasks for each project that are active and you will have options to edit and remove each one. You will also have a button that allows you to add a new one or mark one as complete.

Messages page:
Here you will see a list of the received and sent messages in your inbox and the date you sent them or you received them.

Change password page:
here you will have the ability to change your password.

Edit profile page:
Here you will have the ability to edit your profile.

The overall features for admins

- view statistics
- manage freelancers
- manage client
- manage projects
- manage tasks
- manage admin
- message admins and freelancer
- edit profile
- change password
- login
- logout

9.6.2 Freelancer guide

First page:
The first page you will see will be asking you to choose if you're a freelancer or admin
Choose freelancer and it will take you to log in page

Log in page:
Enter you credentials here. For the sake of testing, you can use these credentials:

username: slide_freelancer
password: slide_freelancer123

home page:
Here you will see a general overview and some statistics about your performance as a freelancer in VTE.

Tasks page:
Here you will see the tasks you are assigned to do and will have the option to mark them as complete.

Messages page:
Here you will see a list of admins that you can send a message to. And you will also see your inbox.

Change password page:
here you will have the ability to change your password.

Profile details page:
Here you will see your account details such as name username and the roles you are assigned etc.

Edit profile page:
Here you will have the ability to edit your profile.

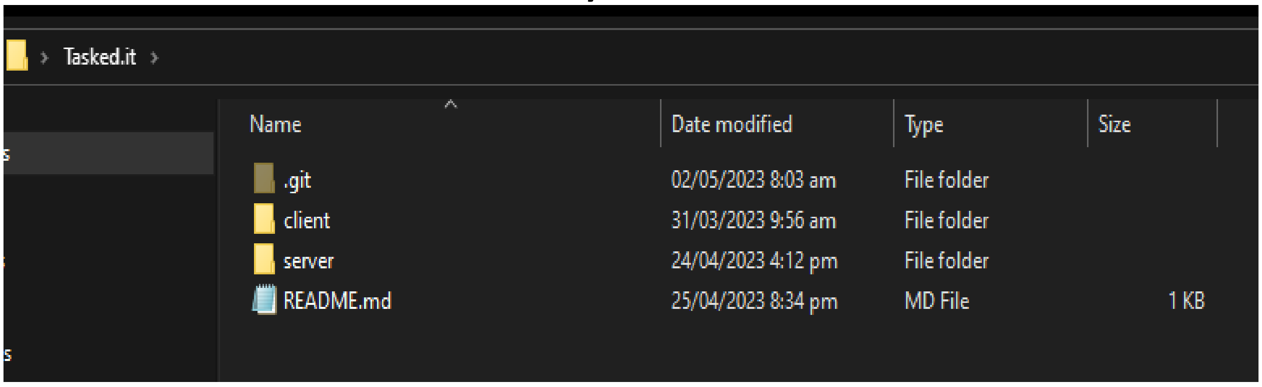
Overall features for freelancers:

- log in
- view tasks
- mark as complete
- see and send messages
- change password
- edit profile
- view profile
- log out

9.7 Configuration management screenshots

I used git and GitHub to track all and any changes I made to the system and track all the features.

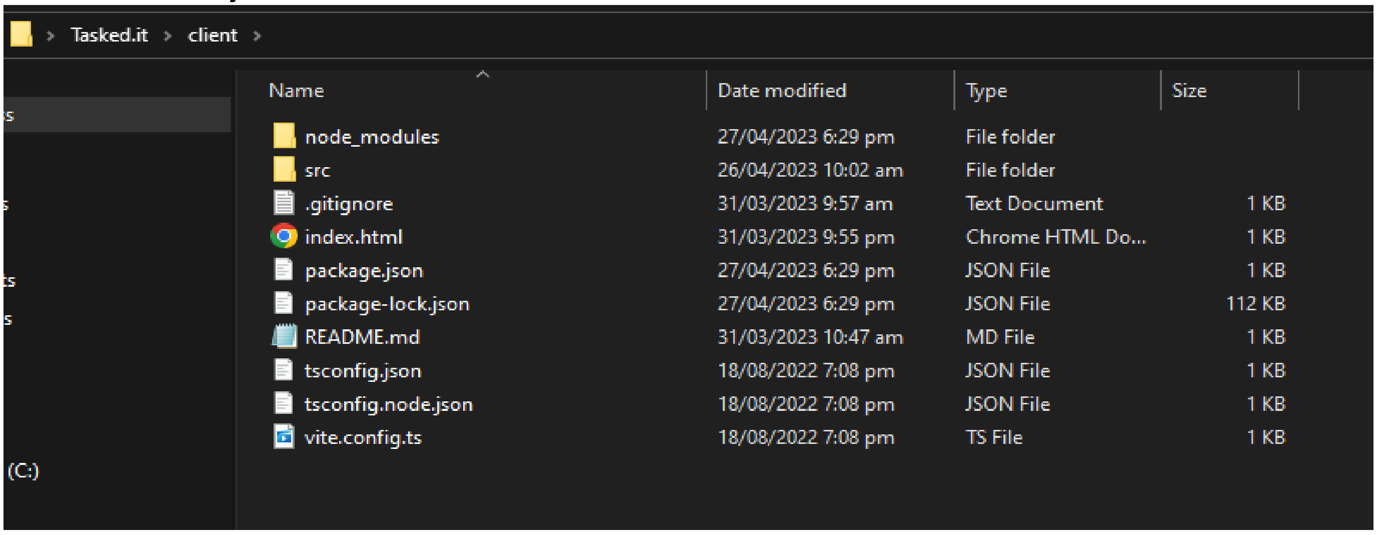
Screenshot evidence for root directory



Tasked.it >

Name	Date modified	Type	Size
.git	02/05/2023 8:03 am	File folder	
client	31/03/2023 9:56 am	File folder	
server	24/04/2023 4:12 pm	File folder	
README.md	25/04/2023 8:34 pm	MD File	1 KB

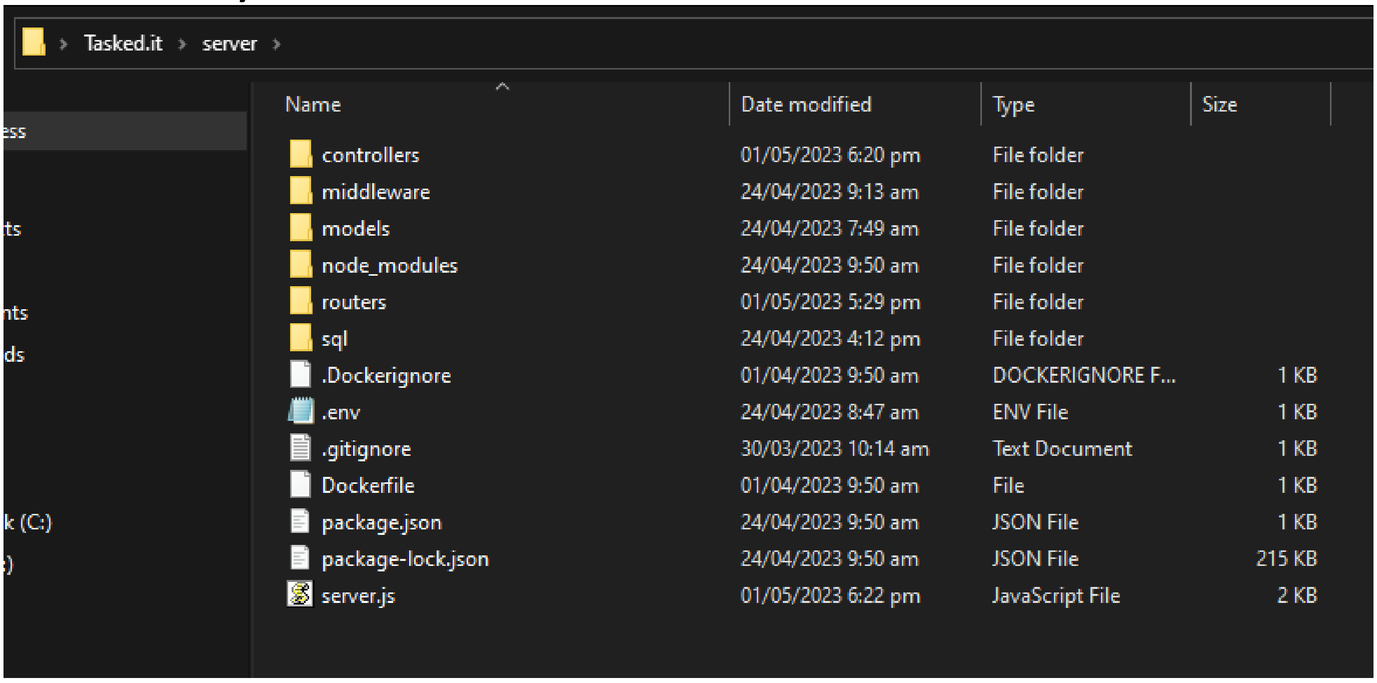
Client directory



Tasked.it > client >

Name	Date modified	Type	Size
node_modules	27/04/2023 6:29 pm	File folder	
src	26/04/2023 10:02 am	File folder	
.gitignore	31/03/2023 9:57 am	Text Document	1 KB
index.html	31/03/2023 9:55 pm	Chrome HTML Do...	1 KB
package.json	27/04/2023 6:29 pm	JSON File	1 KB
package-lock.json	27/04/2023 6:29 pm	JSON File	112 KB
README.md	31/03/2023 10:47 am	MD File	1 KB
tsconfig.json	18/08/2022 7:08 pm	JSON File	1 KB
tsconfig.node.json	18/08/2022 7:08 pm	JSON File	1 KB
vite.config.ts	18/08/2022 7:08 pm	TS File	1 KB

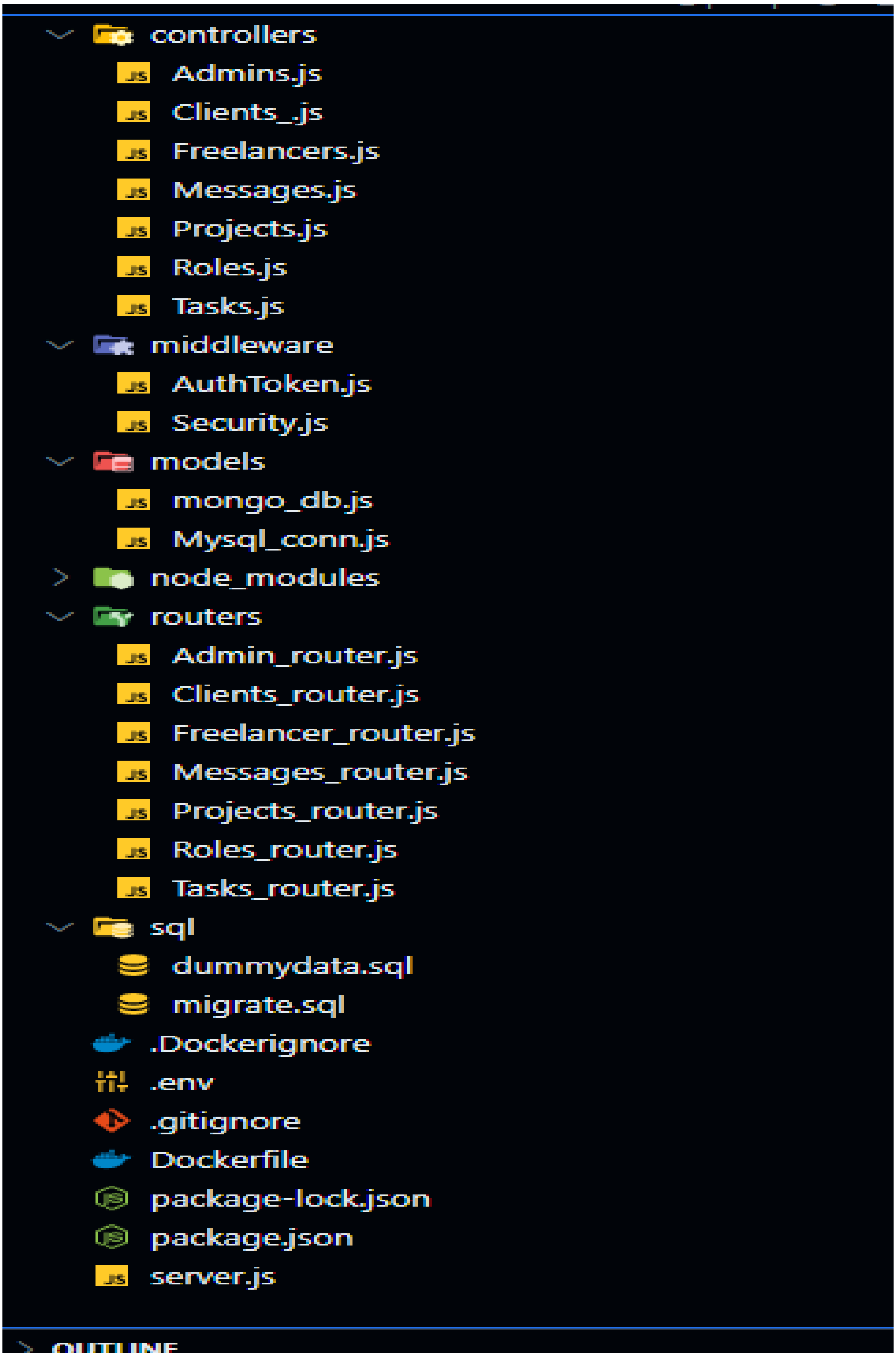
Server directory



Tasked.it > server >

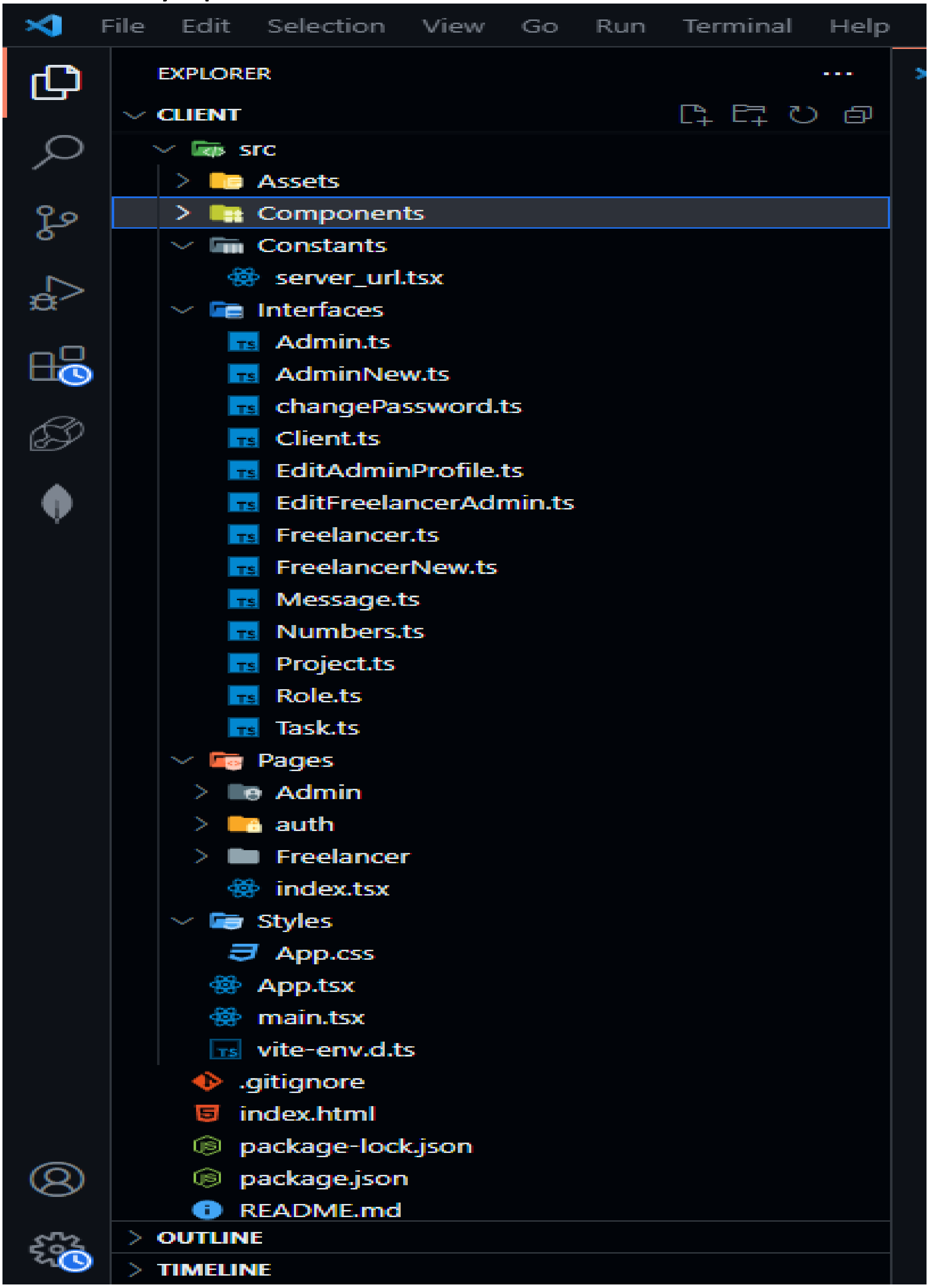
Name	Date modified	Type	Size
controllers	01/05/2023 6:20 pm	File folder	
middleware	24/04/2023 9:13 am	File folder	
models	24/04/2023 7:49 am	File folder	
node_modules	24/04/2023 9:50 am	File folder	
routers	01/05/2023 5:29 pm	File folder	
sql	24/04/2023 4:12 pm	File folder	
.Dockerignore	01/04/2023 9:50 am	DOCKERIGNORE F...	1 KB
.env	24/04/2023 8:47 am	ENV File	1 KB
.gitignore	30/03/2023 10:14 am	Text Document	1 KB
Dockerfile	01/04/2023 9:50 am	File	1 KB
package.json	24/04/2023 9:50 am	JSON File	1 KB
package-lock.json	24/04/2023 9:50 am	JSON File	215 KB
server.js	01/05/2023 6:22 pm	JavaScript File	2 KB

Server directory expanded



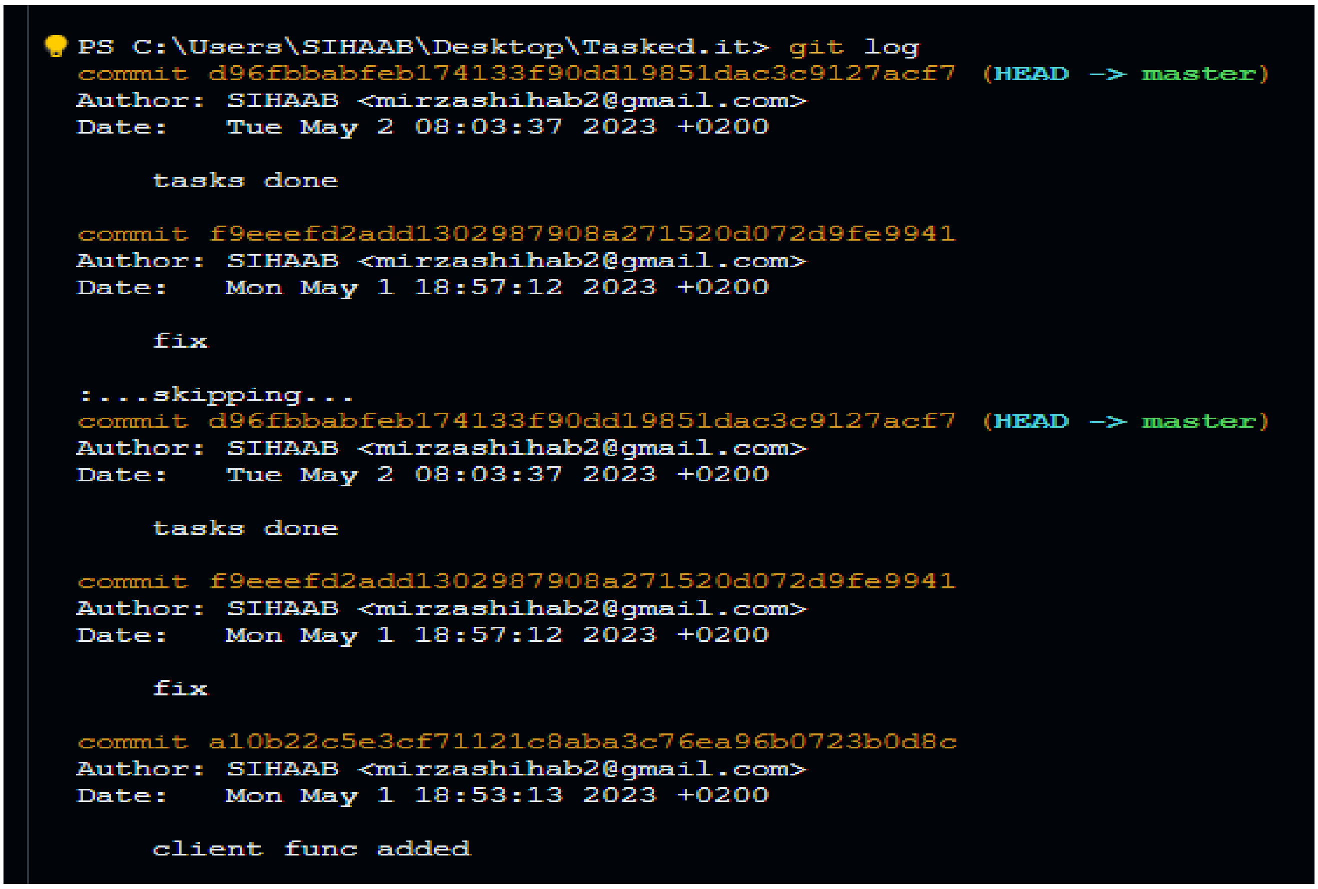
```
controllers
    Admins.js
    Clients_.js
    Freelancers.js
    Messages.js
    Projects.js
    Roles.js
    Tasks.js
middleware
    AuthToken.js
    Security.js
models
    mongo_db.js
    Mysql_conn.js
node_modules
routers
    Admin_router.js
    Clients_router.js
    Freelancer_router.js
    Messages_router.js
    Projects_router.js
    Roles_router.js
    Tasks_router.js
sql
    dummydata.sql
    migrate.sql
.Dockerignore
.env
.gitignore
Dockerfile
package-lock.json
package.json
server.js
```

Client directory expanded



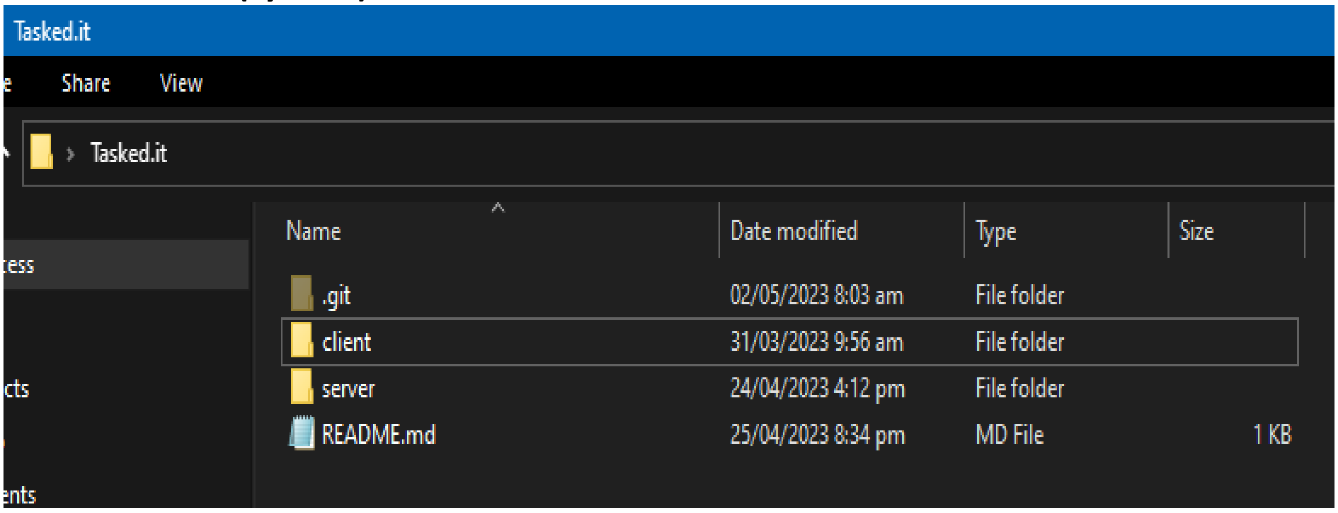
```
CLIENT
  src
    Assets
    Components
    Constants
      server_url.tsx
    Interfaces
      Admin.ts
      AdminNew.ts
      changePassword.ts
      Client.ts
      EditAdminProfile.ts
      EditFreelancerAdmin.ts
      Freelancer.ts
      FreelancerNew.ts
      Message.ts
      Numbers.ts
      Project.ts
      Role.ts
      Task.ts
    Pages
      Admin
      auth
      Freelancer
      index.tsx
    Styles
      App.css
    App.tsx
    main.tsx
    vite-env.d.ts
  .gitignore
  index.html
  package-lock.json
  package.json
  README.md
```

Screenshot of a few previous commits



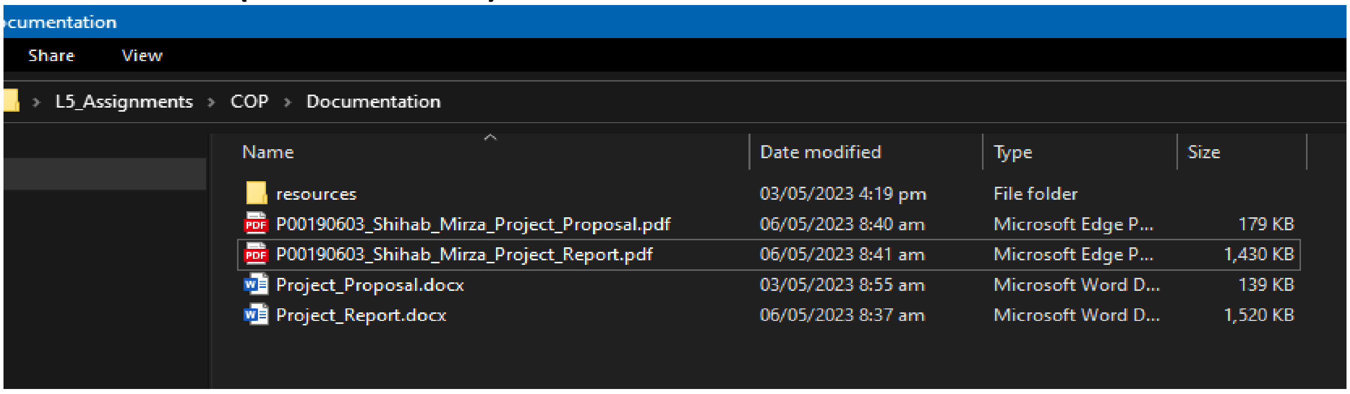
```
PS C:\Users\SIHAAB\Desktop\Tasked.it> git log
commit d96fbbabfeb174133f90dd19851dac3c9127acf7 (HEAD -> master)
Author: SIHAAB <mirzashihab2@gmail.com>
Date:   Tue May 2 08:03:37 2023 +0200

    tasks done

commit f9eeefd2add1302987908a271520d072d9fe9941
Author: SIHAAB <mirzashihab2@gmail.com>
Date:   Mon May 1 18:57:12 2023 +0200

    fix

:...skipping...
commit d96fbbabfeb174133f90dd19851dac3c9127acf7 (HEAD -> master)
Author: SIHAAB <mirzashihab2@gmail.com>
Date:   Tue May 2 08:03:37 2023 +0200

    tasks done

commit f9eeefd2add1302987908a271520d072d9fe9941
Author: SIHAAB <mirzashihab2@gmail.com>
Date:   Mon May 1 18:57:12 2023 +0200

    fix

commit a10b22c5e3cf71121c8aba3c76ea96b0723b0d8c
Author: SIHAAB <mirzashihab2@gmail.com>
Date:   Mon May 1 18:53:13 2023 +0200

    client func added
```

Final structure (system)



Name	Date modified	Type	Size
.git	02/05/2023 8:03 am	File folder	
client	31/03/2023 9:56 am	File folder	
server	24/04/2023 4:12 pm	File folder	
README.md	25/04/2023 8:34 pm	MD File	1 KB

Final structure (documentation)



Name	Date modified	Type	Size
resources	03/05/2023 4:19 pm	File folder	
P00190603_Shihab_Mirza_Project_Proposal.pdf	06/05/2023 8:40 am	Microsoft Edge P...	179 KB
P00190603_Shihab_Mirza_Project_Report.pdf	06/05/2023 8:41 am	Microsoft Edge P...	1,430 KB
Project_Proposal.docx	03/05/2023 8:55 am	Microsoft Word D...	139 KB
Project_Report.docx	06/05/2023 8:37 am	Microsoft Word D...	1,520 KB

9.8 System code

It is important to note that not all the code can be shown here as it is too much so it is best to open the system code in visual studio code and review it from there. Nevertheless, here is the main server file (server.js) from the backend. Without this, the application would be rendered useless.

```javascript
const express = require('express');
const security = require('./middleware/Security');
const cors = require('cors');
require('dotenv').config();
const app = express();
const PORT = process.env.PORT;

app.use(express.json());
app.use(security.securityMiddleware);
app.use(cors({origin: "*"}));
const adminRouter = require("./routers/Admin_router");
const freelancerRouter = require("./routers/Freelancer_router");
const MessagesRouter = require("./routers/Messages_router");
const RolesRouter = require("./routers/Roles_router");
const TasksRouter = require("./routers/Tasks_router");
const ProjectsRouter = require("./routers/Projects_router");
const ClientsRouter = require("./routers/Clients_router");

app.use("/admins", adminRouter);
app.use("/freelancers", freelancerRouter);
app.use("/messages", MessagesRouter);
app.use("/roles", RolesRouter);
app.use("/tasks", TasksRouter);
app.use('/projects', ProjectsRouter);
app.use('/clients', ClientsRouter);

app.get('/test', (req, res) => {
    return res.send("server running");
})
app.listen(PORT || 4455, () => {
    console.log('\nserver running on port 4455');
})
/* This is a Node.js server application using the Express framework. It
imports necessary modules such as `express`, `cors`, and `dotenv`. It sets
up middleware for security and parsing JSON data. It defines routes for
`/admins` and `/freelancers` using separate routers. It also defines a test
route
at `/test`. Finally, it listens on a specified port or a default port of
4455. */
```

9.8 External client documents

MEMORANDUM OF UNDERSTANDING

This Memorandum of Understanding ("MOU") is made and entered into on this day of **3rd April 2023** between SHIHAB MIRZA, hereafter referred to as the "Student," and Visionary Technologies Enterprise ("VTE"), hereafter referred to as the "Company."

Purpose

The purpose of this MOU is to establish an understanding between the Student and VTE for the completion of a computing project.

Project Description

The student will be developing an admin dashboard style website to solve a problem that VTE faces in the company. The project is being completed as a school assignment, and the student will not receive any compensation for their work, unless VTE decides otherwise. VTE will provide the student with all the research information they need to complete the project.

Responsibilities

The student will:

The student should create an admin dashboard style website which meets all of the requirements specified in the project. They should complete their work with diligence and professionalism. Confidential information provided by VTE must only be used in association with this project and not for any other purpose.

VTE will:

Supply the student with all of the knowledge for the research required for the project. Offer assistance and advice as necessary throughout the project. Refrain from using any work finished by the student for any activity outside of the academic task. The student will hold full rights to their intellectual property and VTE will not claim possession to any project created by the student without their authorization. The product is like no other on the market
This item is not similar to anything else that is currently available.

Confidentiality

The student promises to remain discreet about any details given to them by VTE and not discuss it with anyone else unless VTE has specifically given their authorization in writing.

Termination

Either party may terminate this MOU with written notice to the other party if the other party breaches any material provision of this MOU and fails to remedy such breach within thirty (30) days of receiving written notice of such breach.

Governing Law

This MOU shall be governed by and construed in accordance with the laws of the republic of Zambia.

IN WITNESS WHEREOF, the parties have executed this MOU as of the date first written above.

SHIHAB SAJID MIRZA (student)	CEO of Visionary technologies enterprise (VTE)
Signature:	Signature:

PROBLEM STATEMENT:

The current system:
Visionary technologies enterprise (VTE) is a technology products and services company register with PACRA. It builds products for various clients that are in need of a technology services or product. VTE's approach is to hire freelancers who are interested in working on projects that offer a significant payout. The company assesses the skills of these freelancers to ensure that the quality of VTE's products is maintained. After being hired or partnered with VTE, freelancers are assigned projects to work on. Depending on the project's complexity, it may be completed by a single freelancer or split among multiple freelancers, each with a specific skill set required for building specific parts of the project. For example, a graphic designer may design graphics and UI/UX, while a front-end developer may focus on the front end, and a backend developer may work on the back end. Upon project completion and payment from the client, VTE takes its cut and then distributes the remaining funds to the freelancers based on the amount of work they contributed to the project.

The current system operation:
Currently, it keeps record of its hired freelancers and the projects they are assigned to do, on paper. It also communicates to freelancers about the project specifications via WhatsApp, or other social media applications.

Business operation:
VTE looks for clients or a client sends VTE a message to do build a website, app or design an advert etc. sends a quotation, if client agrees on the price:
VTE sends a message in the WhatsApp group to notify freelancers about the project.
Interested freelancers will communicate with VTE.
VTE chooses the best freelancers among them, if the they are occupied/busy on another project, it then shifts priority to the next best freelancers and so on.
Depending on the complexity of the project, VTE may split it up amongst 2 or more freelancers.
Once project is completed, client will pay and VTE will take its cut which is x % of the initial amount. The rest of the funds are transferred to the freelancer's bank accounts.

Note: VTE interviews freelancers before hiring, to maintain the integrity and quality of their products.

Proposed system:
The proposed system is a web-based application designed in an admin dashboard style. Freelancers and admins will have their own separate dashboards, accessible via the index route and /admins route respectively. Users will need to log in to access their respective dashboards.

Once logged in, freelancers will be able to view a page showing the current projects they are assigned to. Clicking on a project will provide further details such as the due date, specific components of the project they are responsible for, the compensation they will receive, and information about the other freelancers working on the same project. Freelancers will have the ability to mark a task as complete or choose to unlink themselves from a project if they are unable to complete the given task.
Admins, on the other hand, will have a page to add new freelancers and view a list of all current freelancers, with options to remove, block, or ban a freelancer as needed. There will also be a page where admins can update a freelancer's details.
Finally, admins will be able to assign projects to freelancers and mark projects or tasks as complete, when necessary, all within their dashboard.