# 

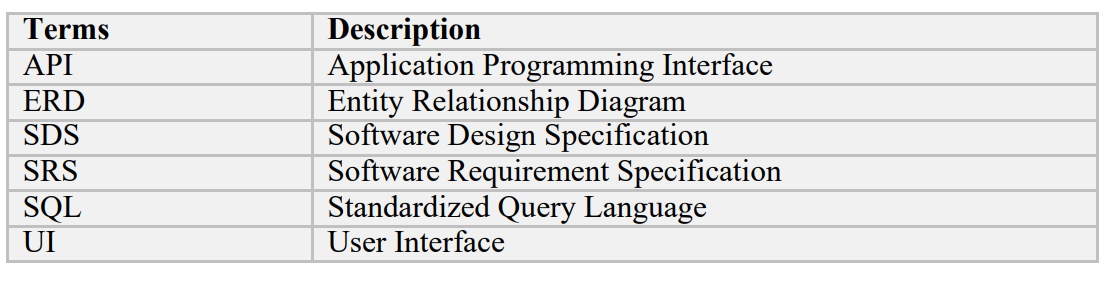
# Software Design Specification

1. **Introduction**
   1. **Purpose of this document**

The purpose of this document is mainly to break down the product into components to describe in detail what the purpose of each component is and how it will be implemented.

* 1. **Scope of the development project**

We aim to create a platform that targets freelancers with much easier policies and regional service charges. On the other hand, to tackle the issue of unprofessional and unskilled freelancers, we aim to provide a customized filtering process to ensure that only professional and skilled freelancers can be allowed to work on the platform. We also provide a customized test related to the job that is being bided by freelancers so that only matched freelancers can bid on outsourced projects by businesses, as this way we can tackle much of the filtering part we can offer to the clients. To tackle the issue of the order policy we will make sure that the freelancer has the choice to accept or decline customer job offers made by the client.

* 1. **Definitions, acronyms, and abbreviations**
  2. **References**

- IEEE Template for System Design Specification Documents: <https://goo.gl/nsUFwy>

* 1. **Overview of document**

This System Design specification document contains the following information:

**Section 1.0**: An introduction to the document by explaining the purpose of the document, the scope providing terms and references, and a brief overview of each section within the document.

**Section 2.0**: An overview of the system architecture, system components and modules, relationships between the various components, and user interface issues without any technical details.

**Section 3.0**: A detailed description of each software component used in our product.

**Section 4.0**: A detailed description of the user interface including screen images.

**Section 5.0**: An explanation of the reusability of existing products and relationships within the product.

**Section 6.0**: A listing of the design decisions and tradeoffs made during the design phase. This section will help readers and users understand the reasoning for these decisions.

**Section 7.0**: Pseudo code implementation for the different algorithms used in the creation of neural networks and in its training and simulation.

**Section 8.0**: It contains an appendix.

1. **System architecture description**

This section provides an overview and rationale for the program's data and architectural design decisions.

* 1. **Section Overview**

This section describes the constraints and usability of the system. Data design are used to explain the major features of the system.

It describes the complete structure of the software and detail description of each individual features. It also describes the architecture design with major components of the software.

* 1. **General Constraints**

Pakwork, the main thing which is to be kept in mind is to provide flow in the performance of the system. The system will be easy to understand, easy to use, responsive, secure and the most important it will be so simple so that the user can use it smoothly. Website does not need any heavy performance computer normal system can also access this website; however, we do require an internet connection in order to use the website. The project is flexible and can modify any part of the project. This software will work on web platform. There is no more external software which is connected to this software directly or indirectly. The database of the website is placed on MySQL and MongoDB which requires internet connectivity to fetch and store data. The Mailing System is based on Gmail so the SMTP service also requires an internet connectivity for its communication.

* **Hardware Interface:**

Since the tool has no designated hardware and it can run on any computing machine it does not have any direct hardware interface. Although for optimal performance and faster results the user is required to use a high-speed internet.

* **Software Interface:**

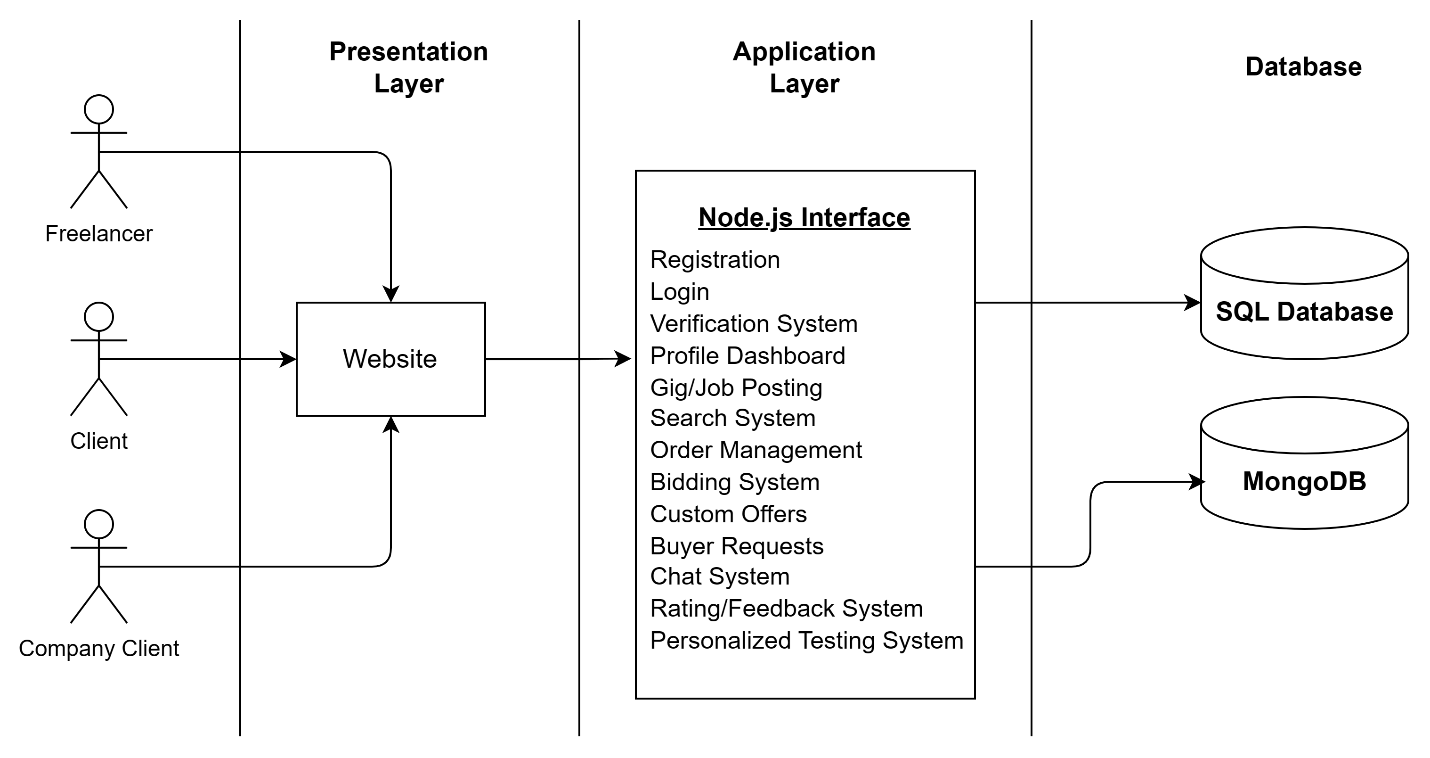
The software interfaces that are used in the tool are as follows:

Microsoft Visual Studio Code: Microsoft Visual Studio Code is a lightweight text editor that is widely-used for any particular programming task, mainly it is used for web development. Due to its performance many developers consider it as their daily go-to text editor when they’re programming scripts or applications.

MySQL Workbench: MySQL is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications— which may run either on the same computer or on another computer across a network.

MongoDB: MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas.

Communication Interface: The communication between the different parts of the system is important since they depend on each other. However, the most important communication happening in the website is between web-based interface used by the user to communicate with the system and that is locally done.

* 1. **Data Design**
* Reference to: SRS Appendix B: Entity Relation Diagram, Domain Model   
  1. **Program Structure**

The major functions the product must perform are listed as follows:

* Freelancer, Client and Company Client would be able to register themselves on the platform
* Freelancer, Client and Company Client would be able to log on to the platform
* Freelancer, Client and Company Client would be able to customize their profile
* Freelancer would be able submit verification documents for verification.
* Freelancer would be able to create, edit or delete Gigs
* Client and Company Client would be able to create, edit or delete custom jobs, requests and offers
* Freelancer, Client and Company Client would be able to chat with each other.
* Freelancer would be able to accept offers sent by Client or Company Client.
* Client and Company Client would be able to start a bid on their project.
* Freelancer would be able to place bids on the projects.
* Freelancer would be able to search for active jobs.
* Client and Company Client would be able to accept the order delivered by Freelancers.
* Freelancers would be able to deliver the order placed by the Client or Company Client.
* Client and Company Client would be able to leave feedback and rate the order.
* Company Client would be able to create any personalized test for the Freelancers.
  1. **Alternatives Considered**

N/A

1. **Detailed description of components**

This section represents the meat of your document. Be as detailed as time allows.

* 1. **Section Overview**

This section describes the complete description of all components in detailed with their function’s dependencies, their relationships with other components. It describes the interface of each component with input and output requirements.

* 1. **Component of Gigs & Jobs Feature:**

**3.2.1 Description:**

Creating Gigs and Jobs is the most important feature of our application because that’s how the whole process of freelancing would work out, the freelancer makes their gig and the buyer puts out a job that matches the suitable gigs.

**3.2.2 Data Members:**

* **Include type**

Gigs allow clients to see potential freelancers for their work according their gig description and jobs allow freelancers to match their appropriate buyer according to their job description

* **Visibility**

Gigs are publicly viewable for all while jobs are only available to freelancers and clients who posted that job.

* **Description**

Freelancer can make gigs and clients can view them and communicate with them, clients can post jobs and freelancers can view them and send particular offer to the client.

**3.2.3 Methods**

|  |  |
| --- | --- |
| Identification | The Gigs/Jobs component can be accessed by the freelancers and clients to interact on the platform. It is located on the dashboard for the appropriate user |
| Type | It is a module having several sub components |
| Purpose | The freelancer makes their gig and the buyer puts out a job that matches the suitable gigs. |
| Function | Freelancer can make gigs and clients can view them and communicate with them, clients can post jobs and freelancers can view them and send particular offer to the client. |
| Subordinates | Gig id, Job id and username are used to differentiate between gigs and jobs creators |
| Dependencies | This component depends on the user provided details and further used in different ways in the system. User can use this component on website. |
| Interfaces | The user interface of course feature is situated in the user profile page. SQL queries is used to connect the interfaces. And the code which is running on back-end implements sessions which means user session will maintain automatically and user can view data according to the gig id or job id. |
| Resources | Database access and External APIs permissions is required to run this component. |
| Processing | Gigs and Jobs will be created and freelancers, clients can interact with them. |
| Data | This Component will store related Gigs and Jobs data inside the database. |

**3.3 Component of Order Management and Rating/Feedback:**

**3.3.1 Description**

Order management and Rating/Feedback are one of the most important features in our application because with their interaction the client and freelancer can have a secured business on the platform without any 3rd Party involvement.

**3.3.2 Data Members**

* **Include type**

Clients will place an order on a particular offer made by the freelancer and the order will be started. Freelancers would have to deliver the order by sending the necessary content within the time or else they will be penalized for that. After the delivery the client has to accept the delivery or request for changes and leave feedback once the order is completed with a rating as well.

* **Visibility**

Order management is only visible to the freelancer and the client, the rating system will be available to the client while the freelancer can only view the given rating and feedback.

* **Description**

Clients can place the order from the offers sent by the freelancers and start the order and mark it complete once it’s delivered by the freelancer.

**3.3.3 Methods**

|  |  |
| --- | --- |
| Identification | The order component can be accessed by the freelancers and clients to interact on the platform. It is located on the dashboard for the appropriate user and the rating/feedback component will be appeared after the completion of the order |
| Type | It is a module having several sub components |
| Purpose | Clients can place the order from the offers sent by the freelancers and start the order and mark it complete once it’s delivered by the freelancer. |
| Function | Clients will place an order on a particular offer made by the freelancer and the order will be started. Freelancers would have to deliver the order by sending the necessary content within the time or else they will be penalized for that. After the delivery the client has to accept the delivery or request for changes and leave feedback once the order is completed with a rating as well. |
| Subordinates | Order id and usernames are used to differentiate between different orders. |
| Dependencies | The Order Component is dependent on the freelancer’s given offer and the feedback/rating component is dependent on the order that has been completed. |
| Interfaces | The user interface of the order component is accessible from the dashboard and then clicking a particular order and the rating/feedback component will be displayed on that order if the client is viewing. |
| Resources | Database access and External APIs permissions is required to run this component. |
| Processing | Orders and Rating/Feedback will be created and freelancers, clients can interact with them. |
| Data | This Component will store related Order and Feedback/Rating data inside the database. |

1. **User Interface Design**
   1. **Section Overview**

This section provides a view into the set of standards which we have made use of to build a user-friendly Pakwork Website. They have been applied in the best possible way to make the interface appear pleasing to the user and not seem too difficult to understand at the same time.

* 1. **Interface Design Rules**

Component Reusability in React.js is to design all web interfaces. Html, CSS and bootstrapping are used to design all website pages. There is no specific standard followed for designing this website. Users should always be informed of system operations with easy to understand and highly visible status displayed on the screen within a reasonable amount of time. Interface designers should ensure that both the graphic elements and terminology are maintained across similar platforms. For example, an icon that represents one category or concept should not represent a different concept when used on a different screen. Designers should assume users are unable to understand technical terminology, therefore, error messages should almost always be expressed in plain language to ensure nothing gets lost in translation.

* 1. **GUI Components**

Pakwork is not planning to use any external APIs for its functionality and plans on building most of its APIs internally. As for its GUI components are concerned, they’re purely built on React, while the email-templates and external verification links are rendered html directly from node-mailer and express-html.

* 1. **Detailed Description**

Registration: Create your account on the platform (freelancer, client, company client)

Login: Sign-in to your accounts (one login for all types of users )

Dashboard: Show profile detail of the user that is logged in according to their user type.

Profile Management: Update Profile Details such as profile picture, description, social links.

Verification System: Upload Documents to verify as a freelancer on the platform.

Gigs Page: Collection of Gigs according to the given searched text.

Gig Detail Page: Show Detail of the selected gig.

Jobs Page: Collection of Jobs according to the given searched text.

Job Detail Panel: Show Detail of the selected job.

Manage Job Panel: Show appropriate options to create, update a job.

Manage Gig Panel: Show appropriate options to create, update a gig.

Search Box: Search for any type of gigs available.

Manage Orders Page: Show details of the orders along with each’s status.

Order Page: Show details of a particular order with appropriate options.

Chat Inbox Message: Interactive chat conversations with specific users.

Live Bidding Page: Shows all active live biddings that are happening.

Live Bidding Room: Show details of a particular bidding with options to interact.

Custom Offer Panel: Create a custom offer with appropriate options.

Buyer Requests Panel: Create a buyer request with appropriate options.

* For Screenshots refer to appendix in the Software Requirement Specifications

1. **Reuse and relationships to other products**

Reusability allows developers to be more efficient because the same code can be developed once and used in many different applications. Second, reliability can be improved by reusing previously developed, and previously tested, components. The development of new code entails additional costs in time and money for testing, validation, and verification. Much of these expenses can be avoided by using already manufactured external components.

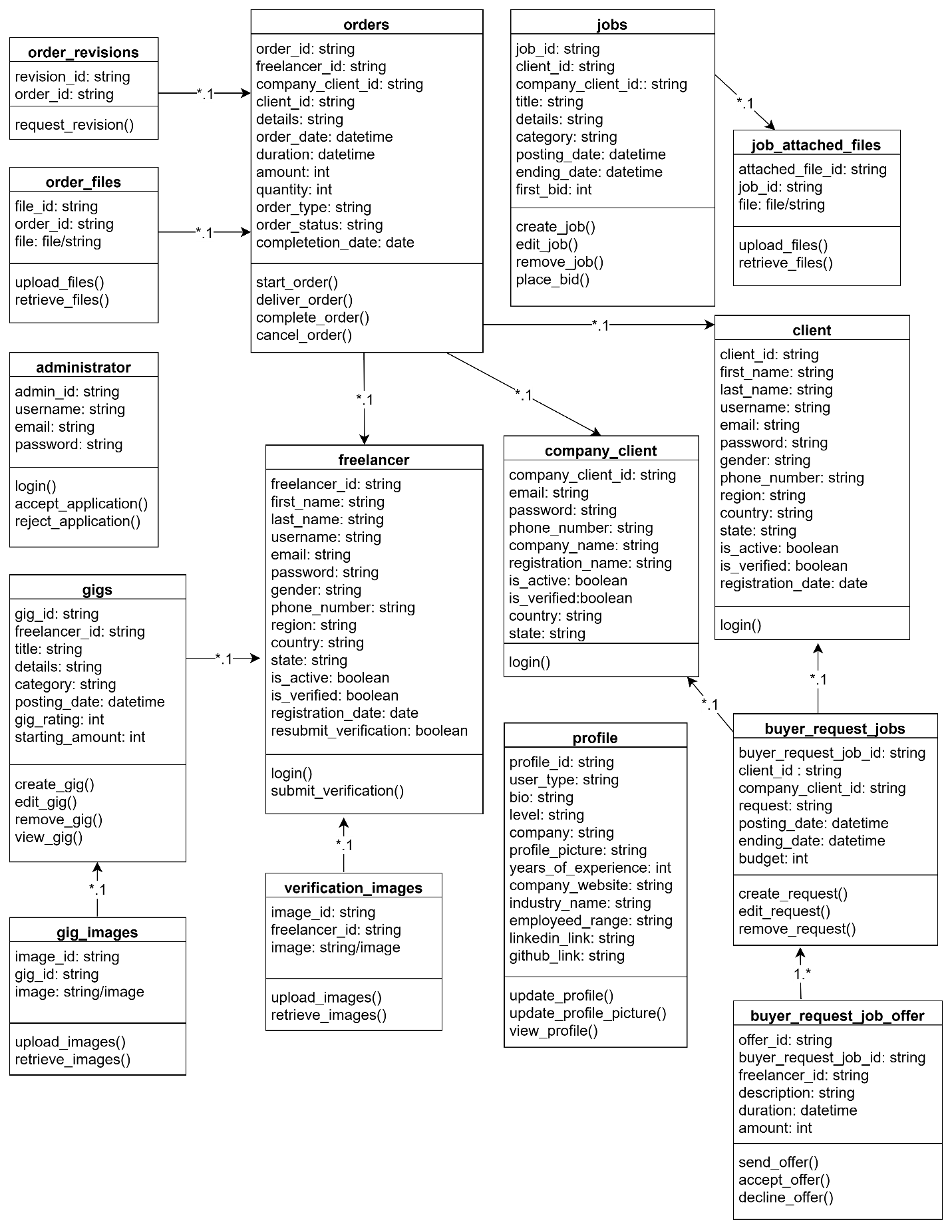
1. **Design decision and tradeoffs**

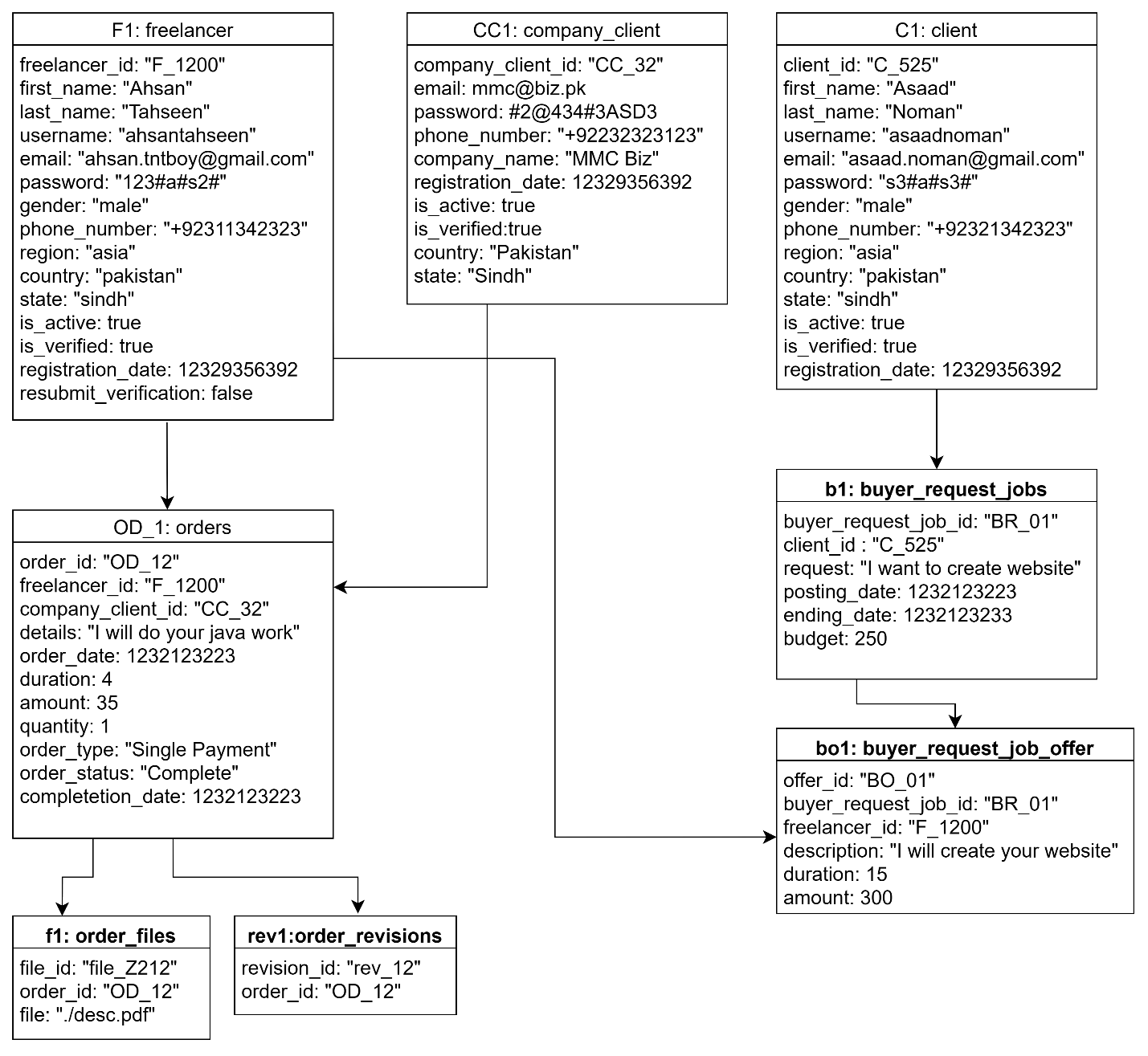
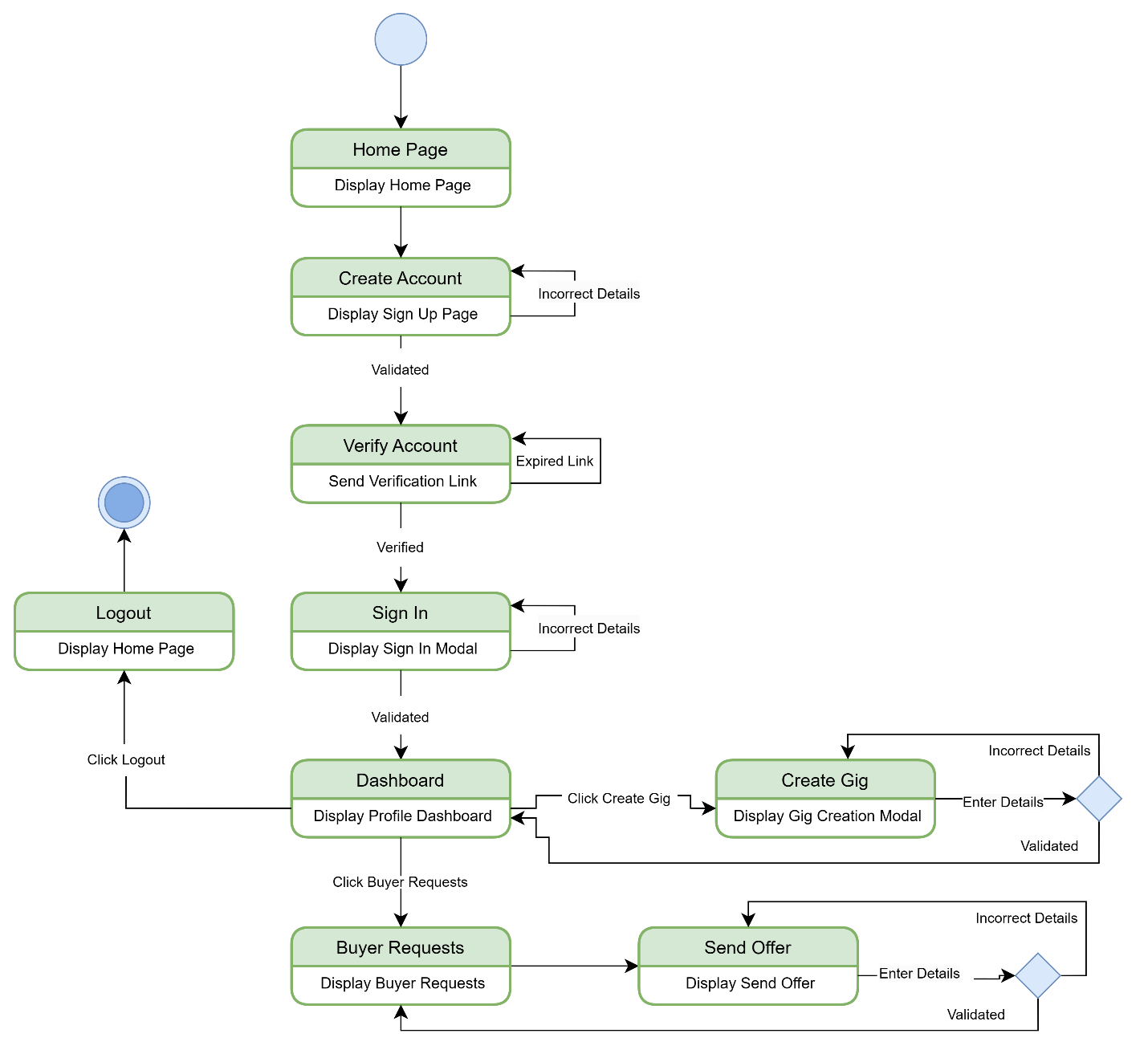
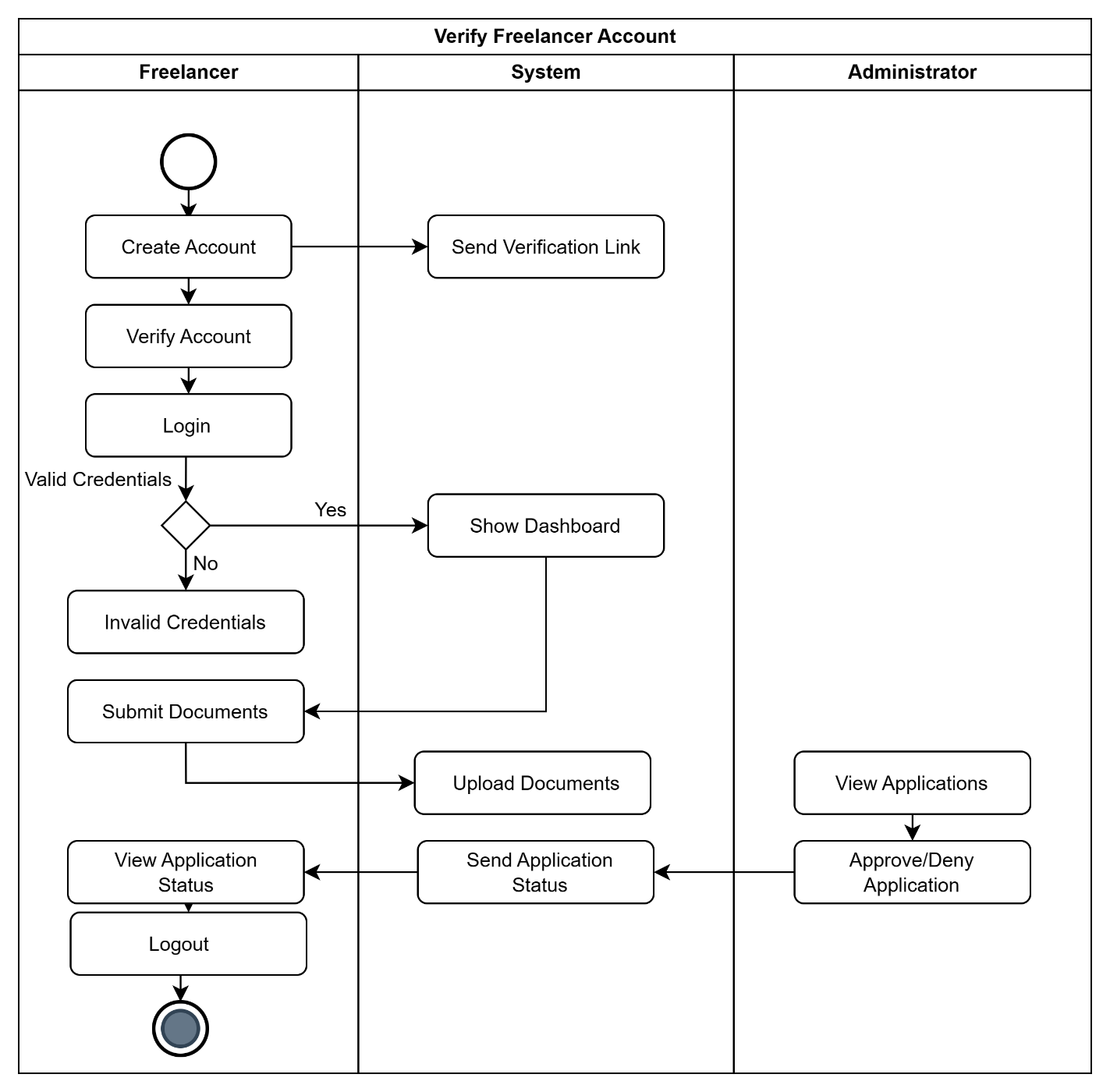
**N/A**

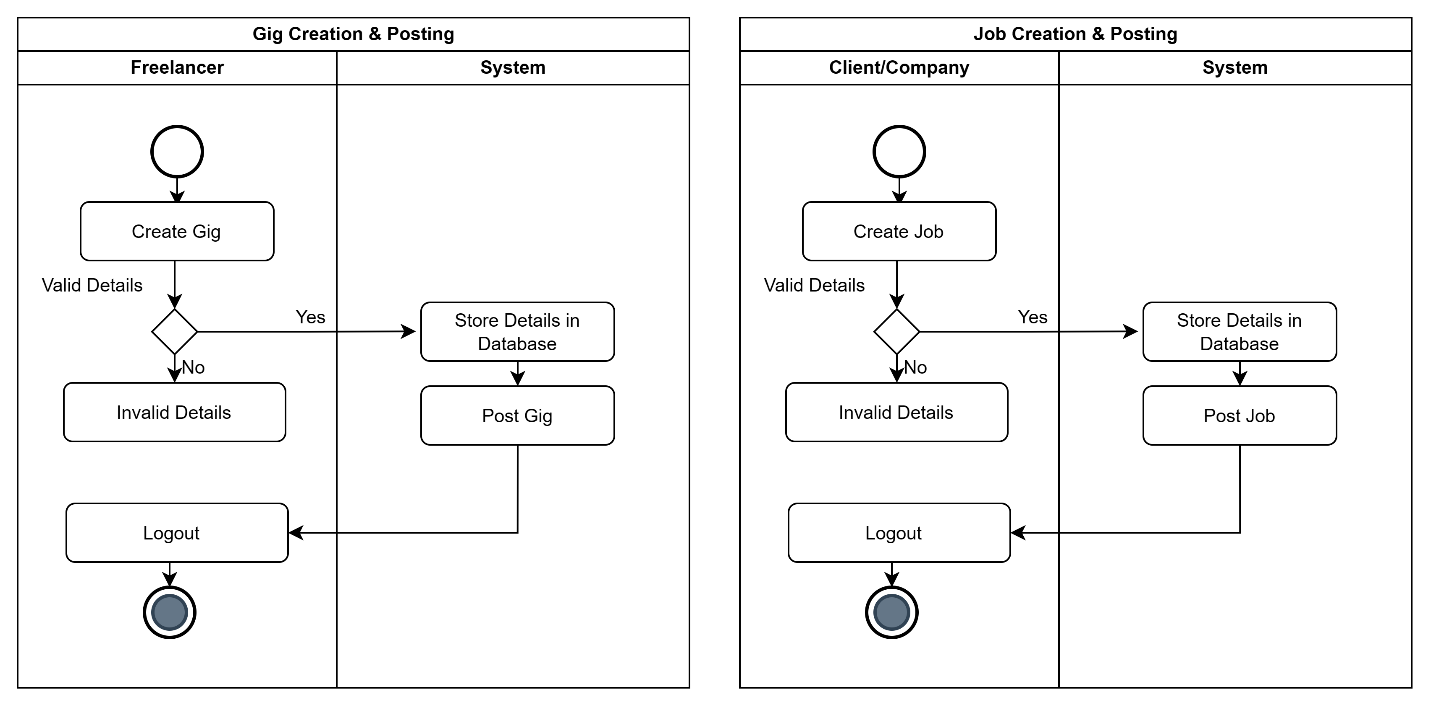
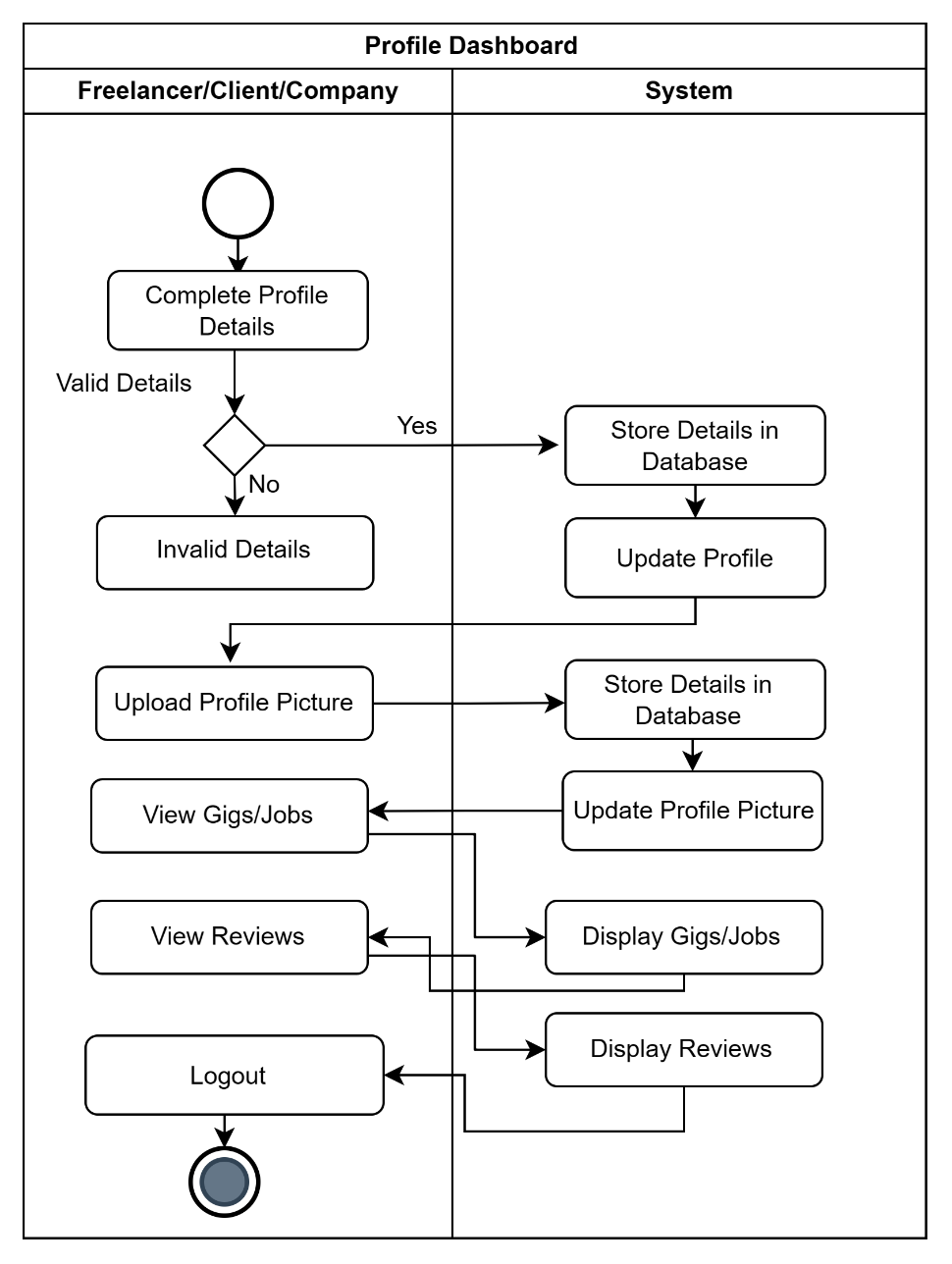
1. **Pseudocode for components**

**N/A**

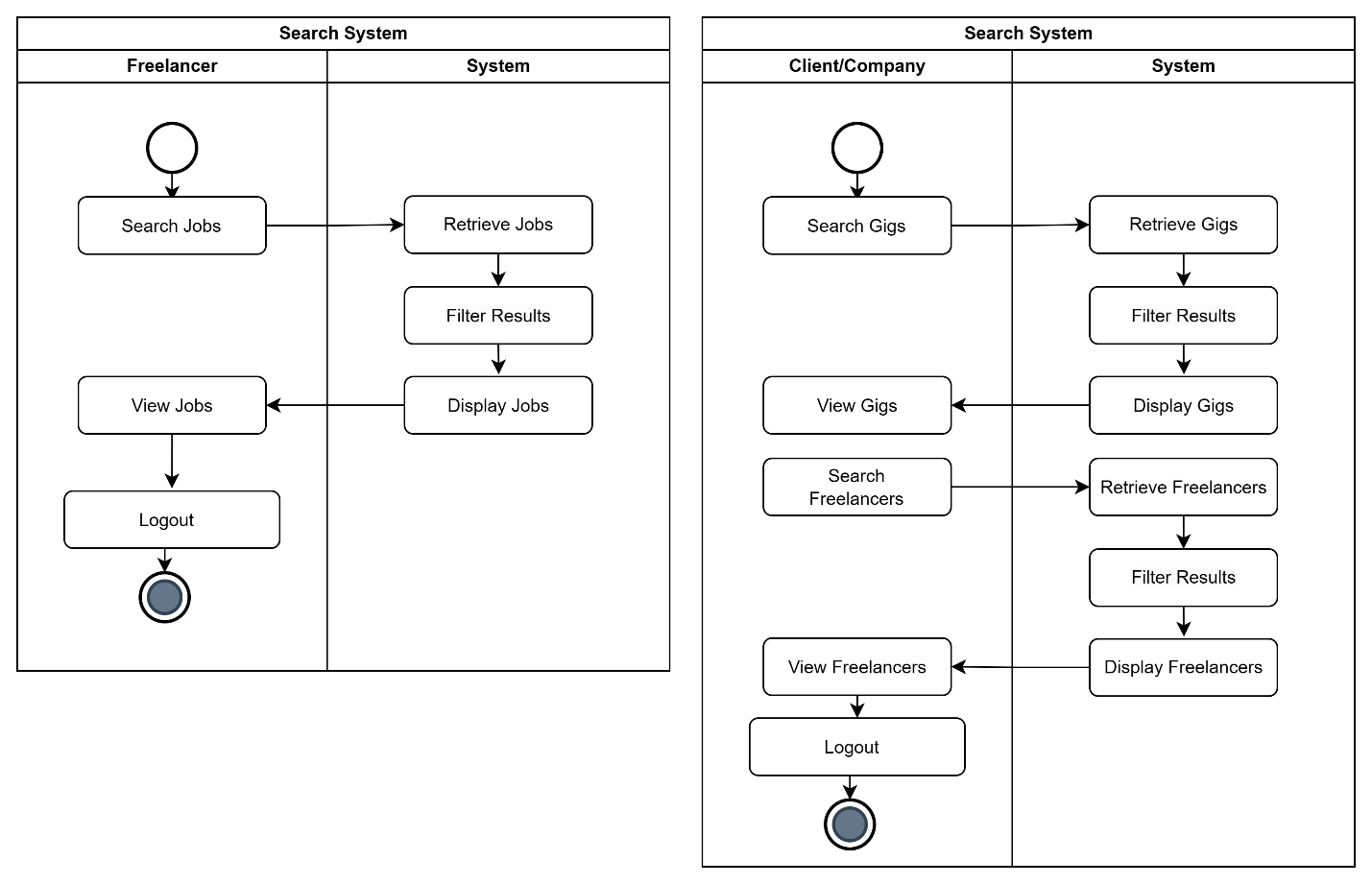
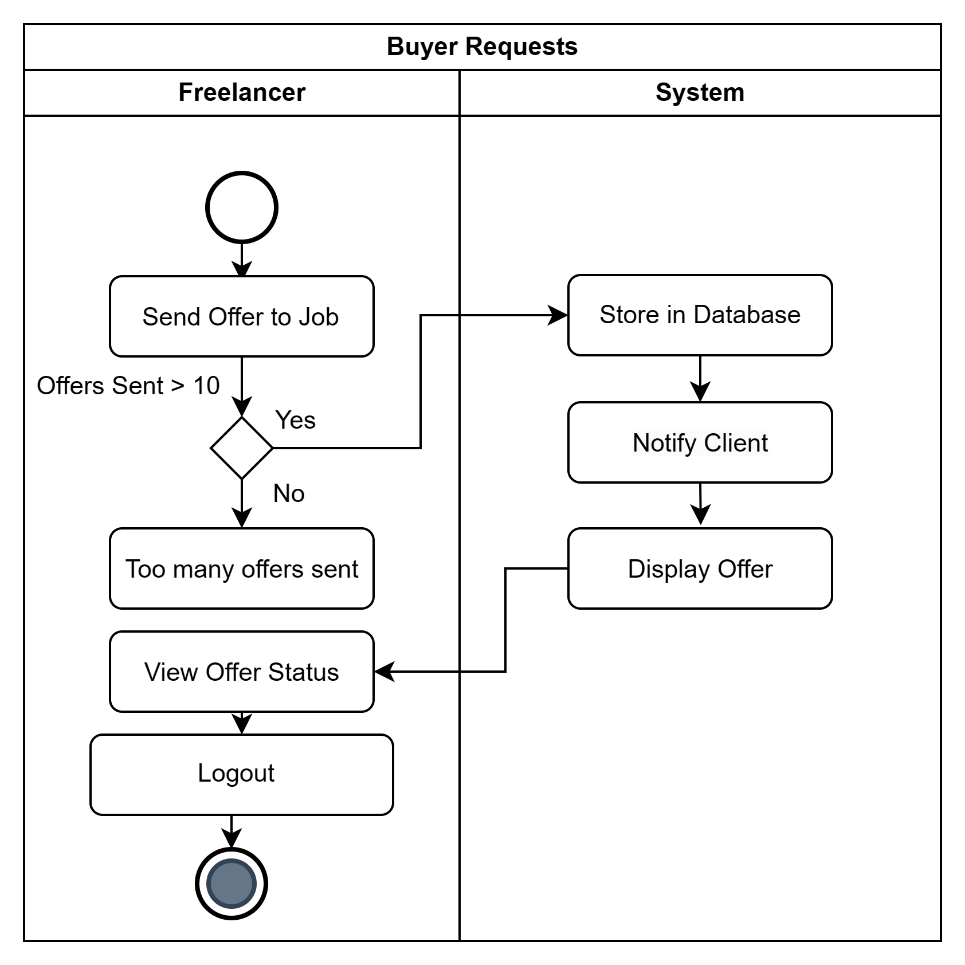
1. **Appendices**
2. **Class Diagram**

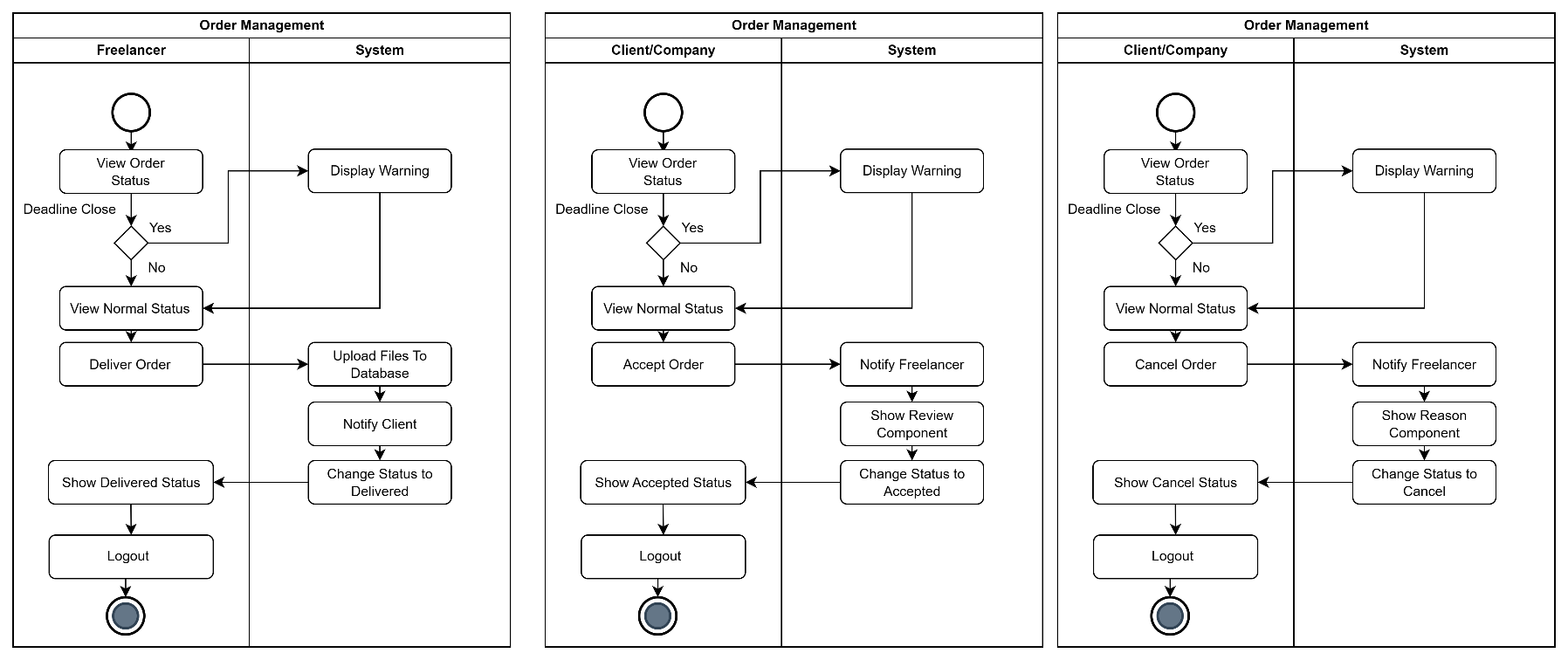
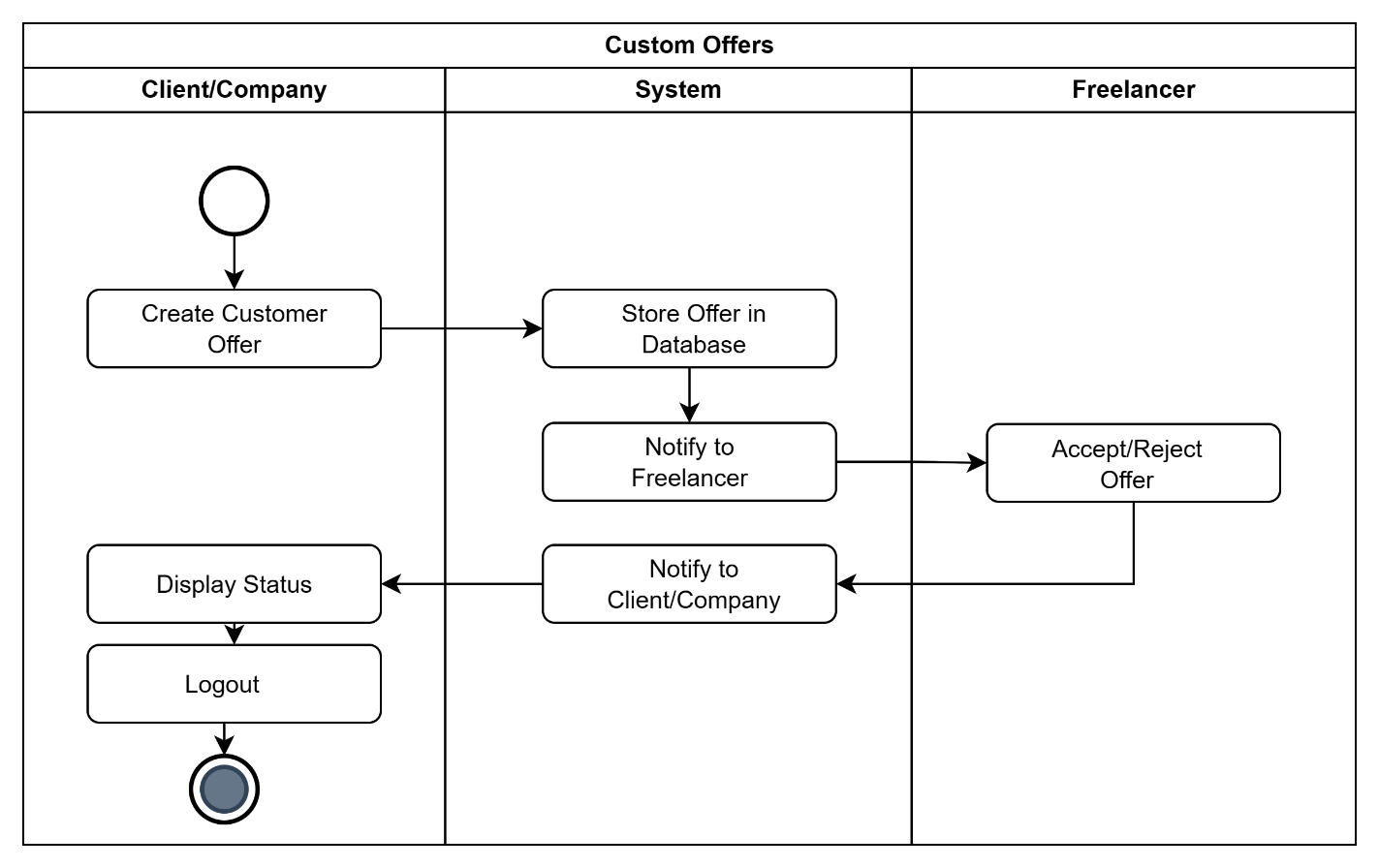
****

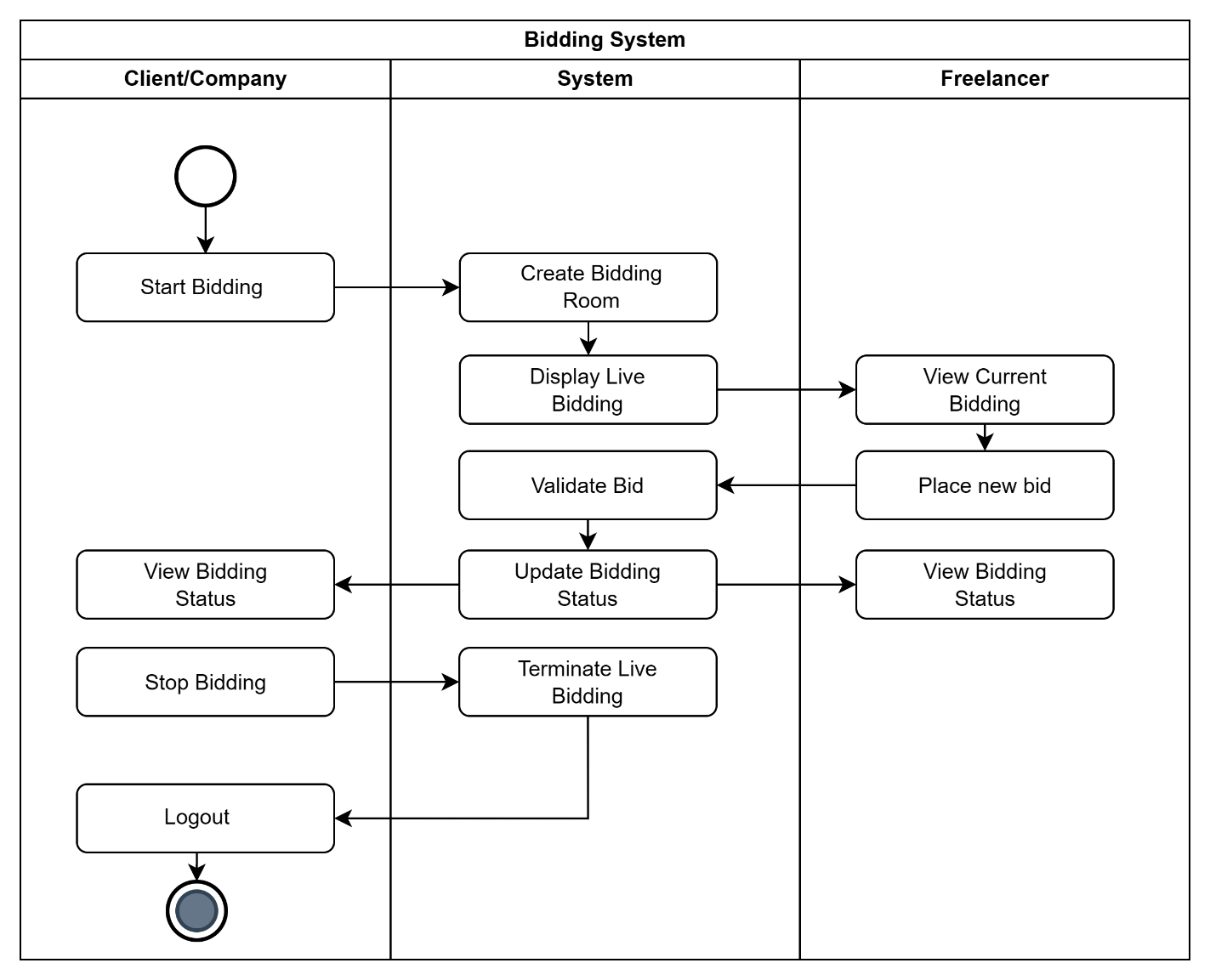
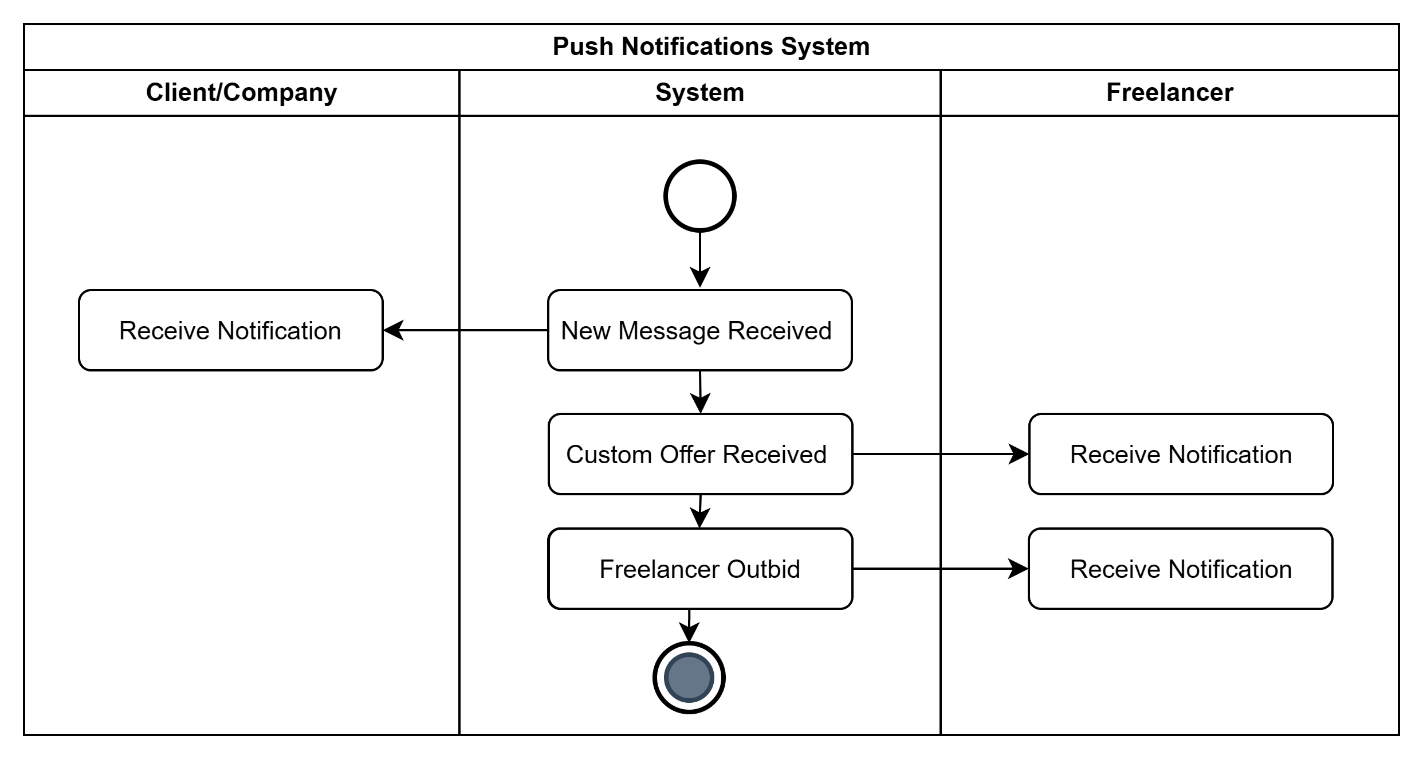
1. **Object Diagram**
2. **Statechart Diagram**
3. **Activity Diagram**

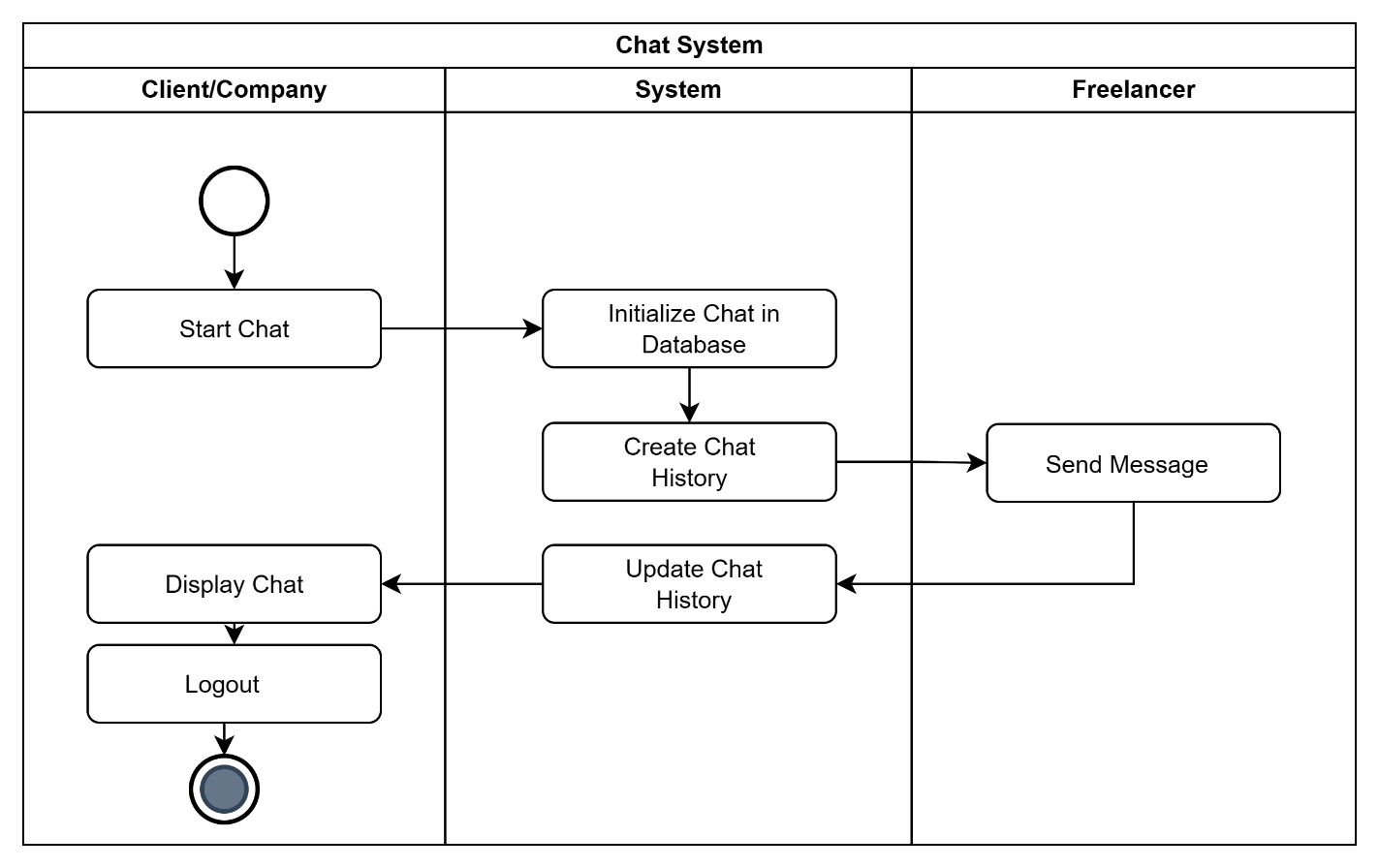
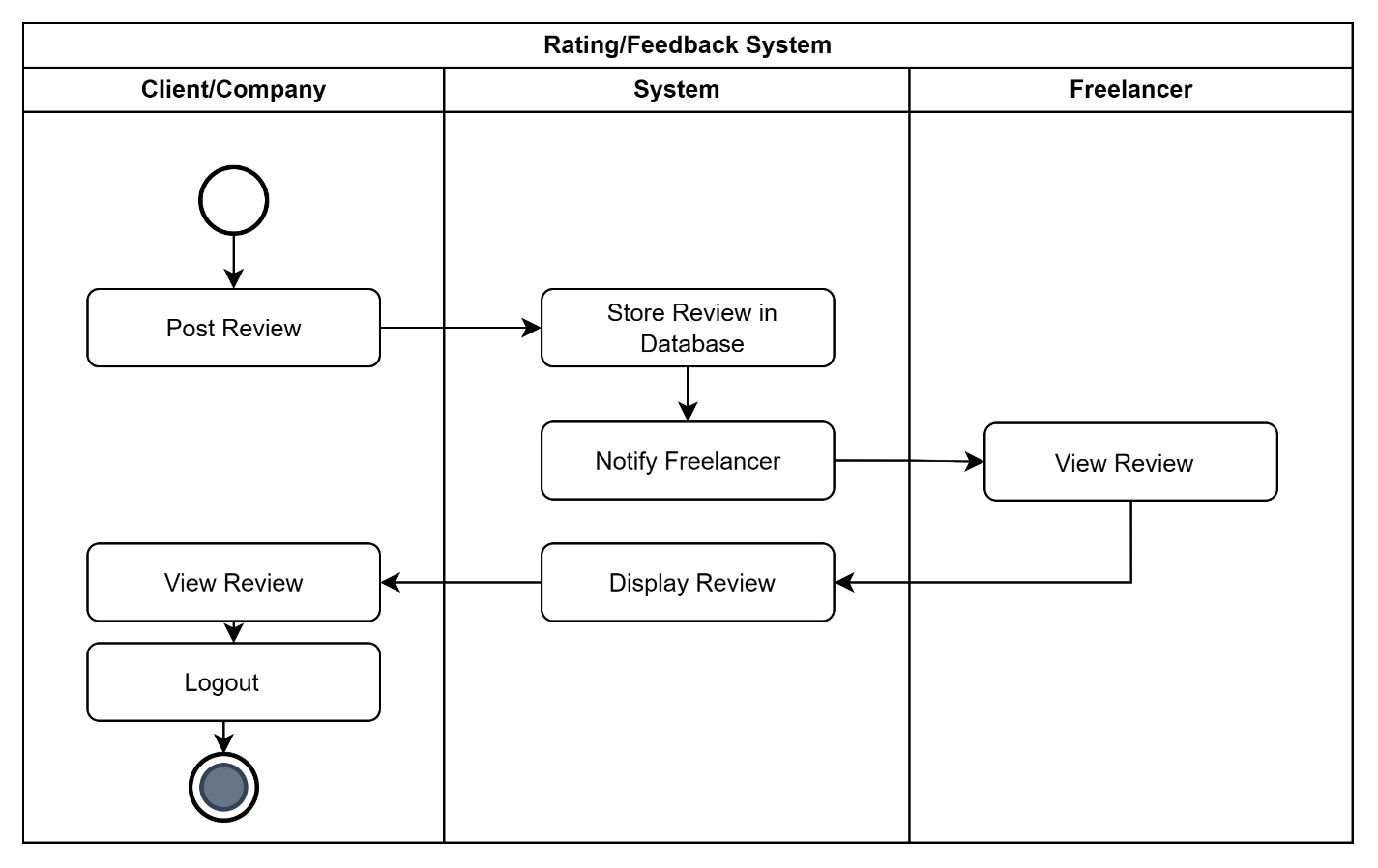
****

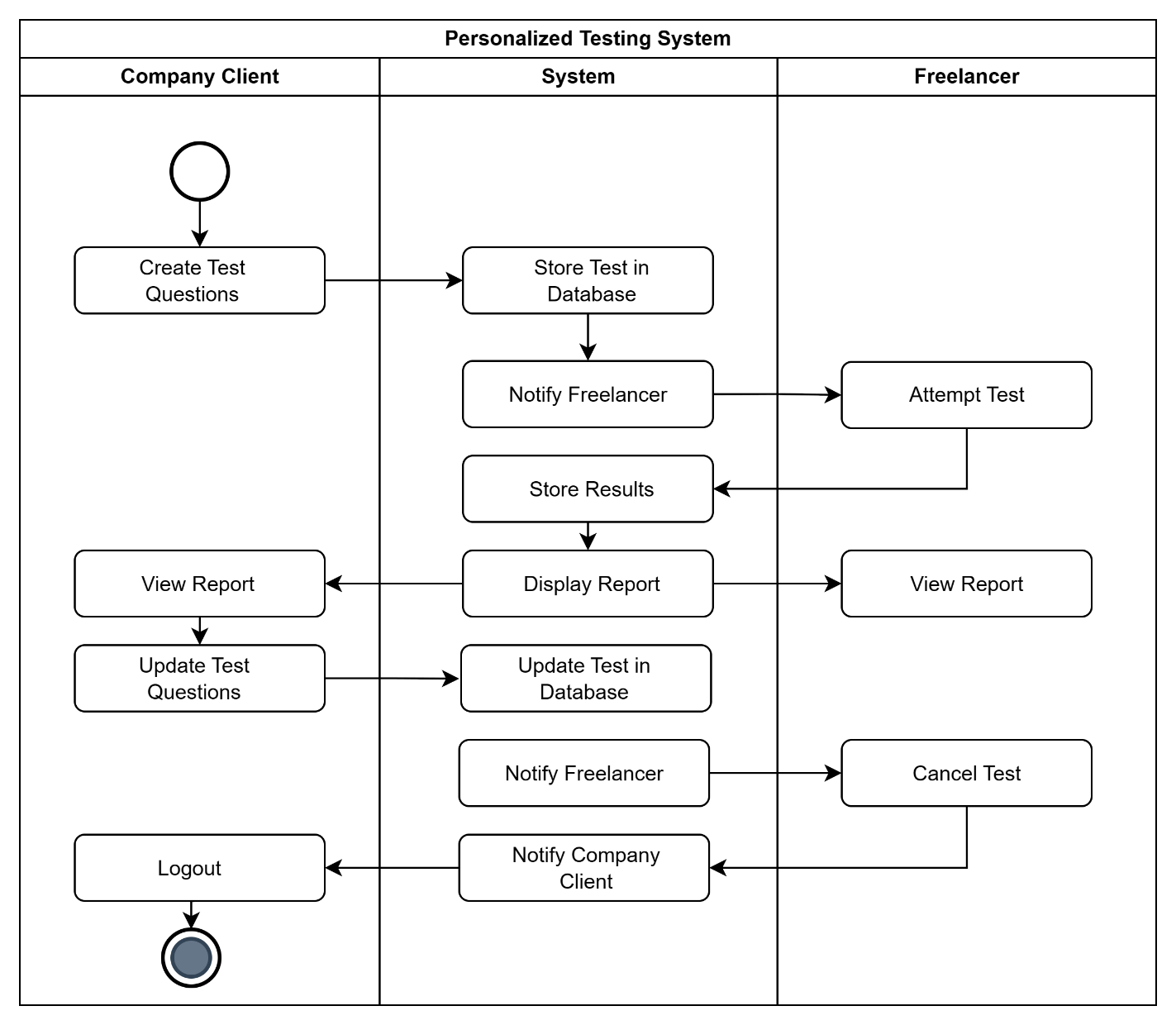
**­**

****

****

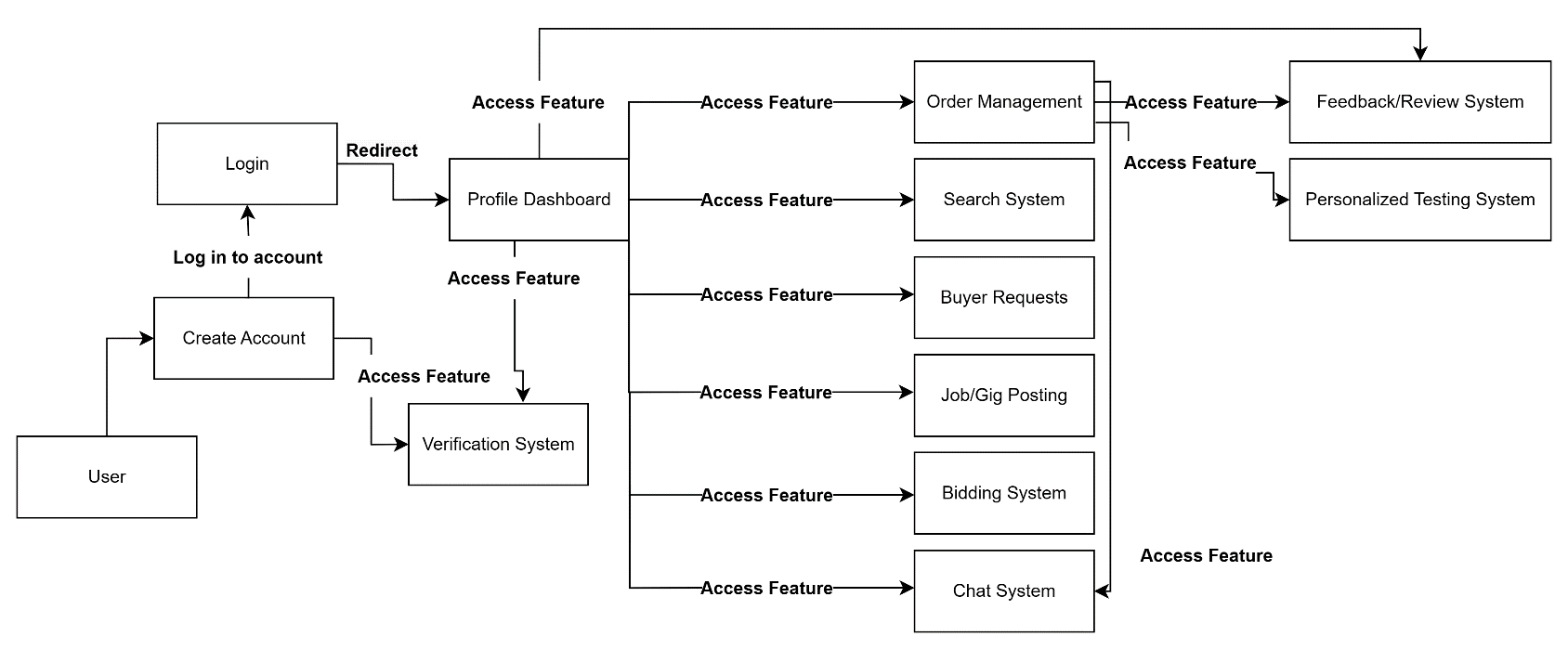
****

****

****

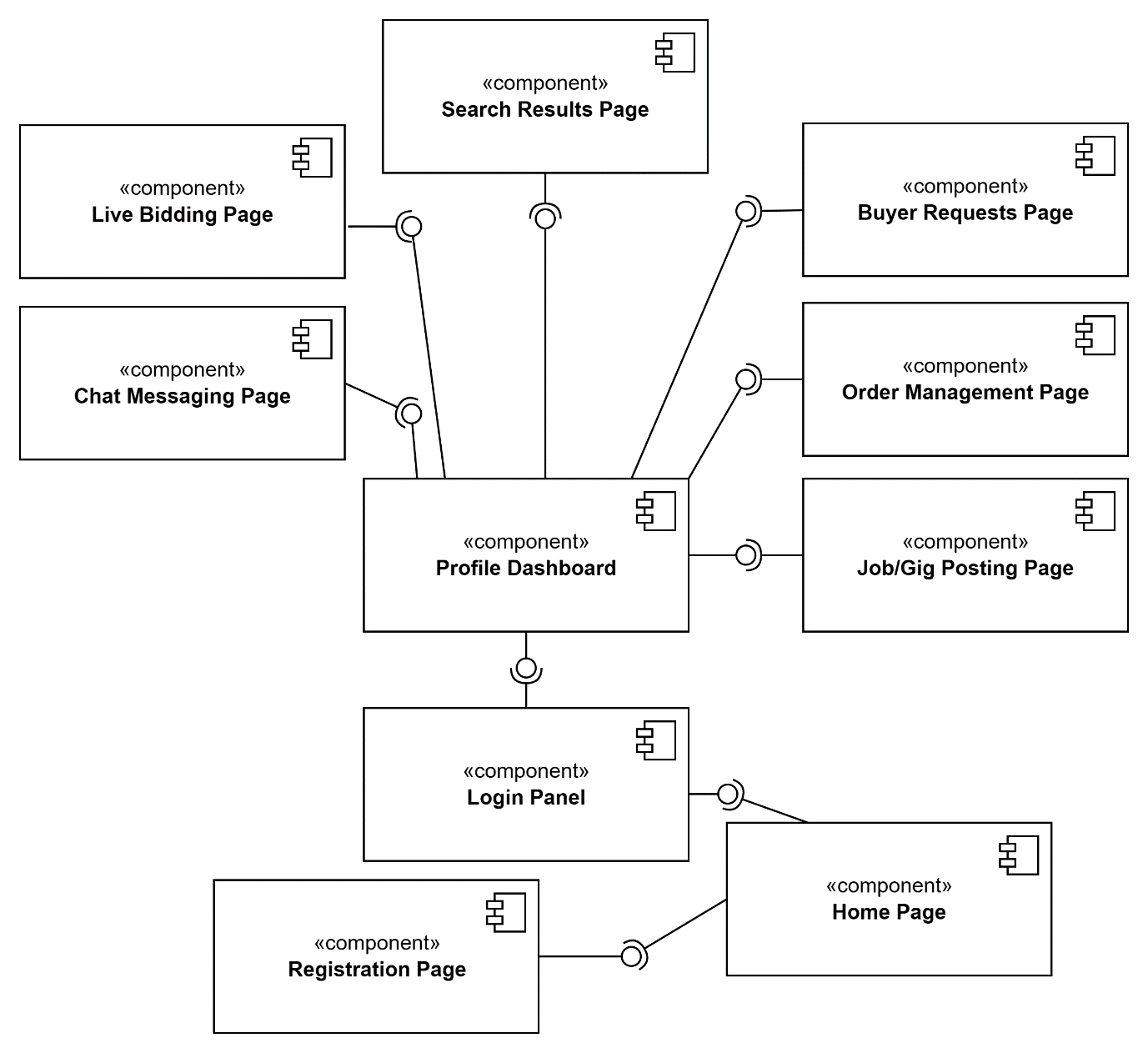
1. **Sequence Diagram**

* Refer to Appendix in System Requirements Specification

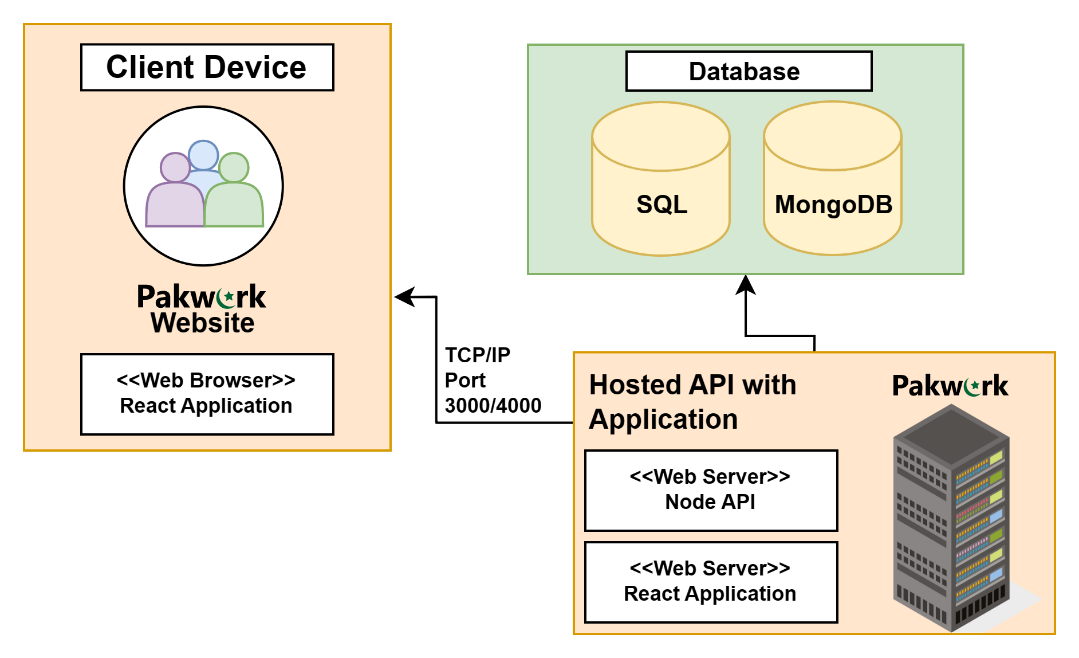
1. **Collaboration Diagram**
2. **Use Case Diagram**

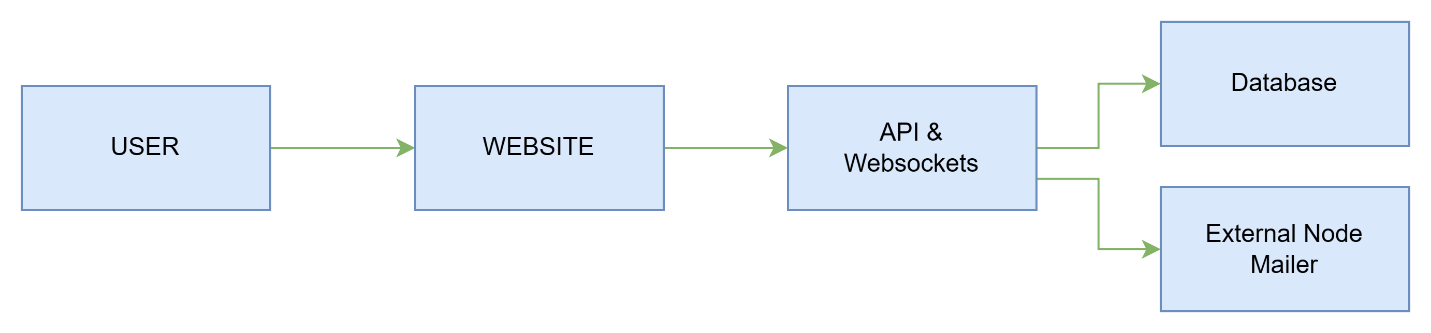
* Refer to Appendix in System Requirements Specification

1. **Component Diagram**

****

**9. Deployment Diagram**

****

**10. System Block Diagram**