Design Document

for

Co-Lancer
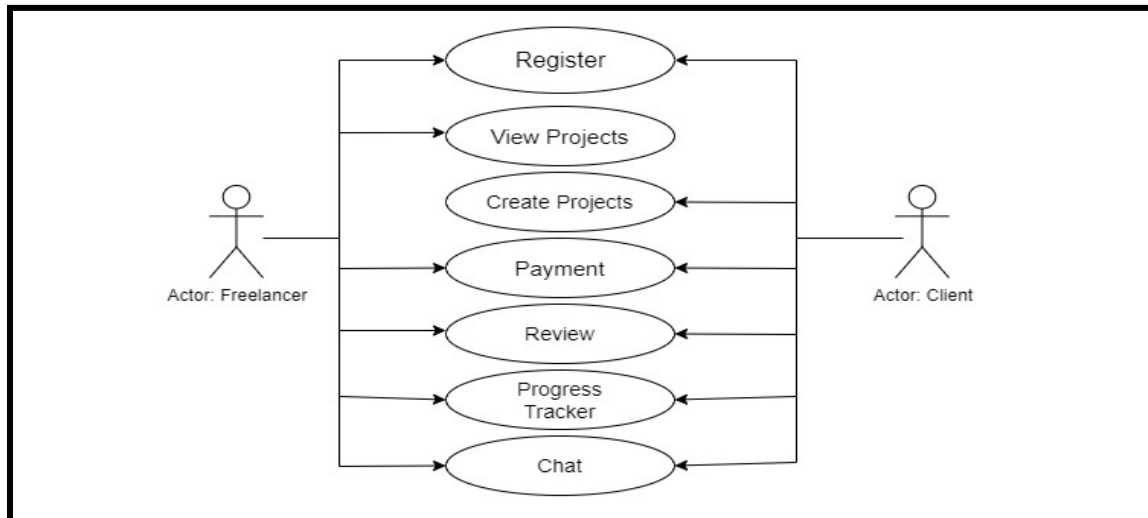
Version 1.0 approved

Prepared by

PES2UG21CS374- POOJA SATHEESH

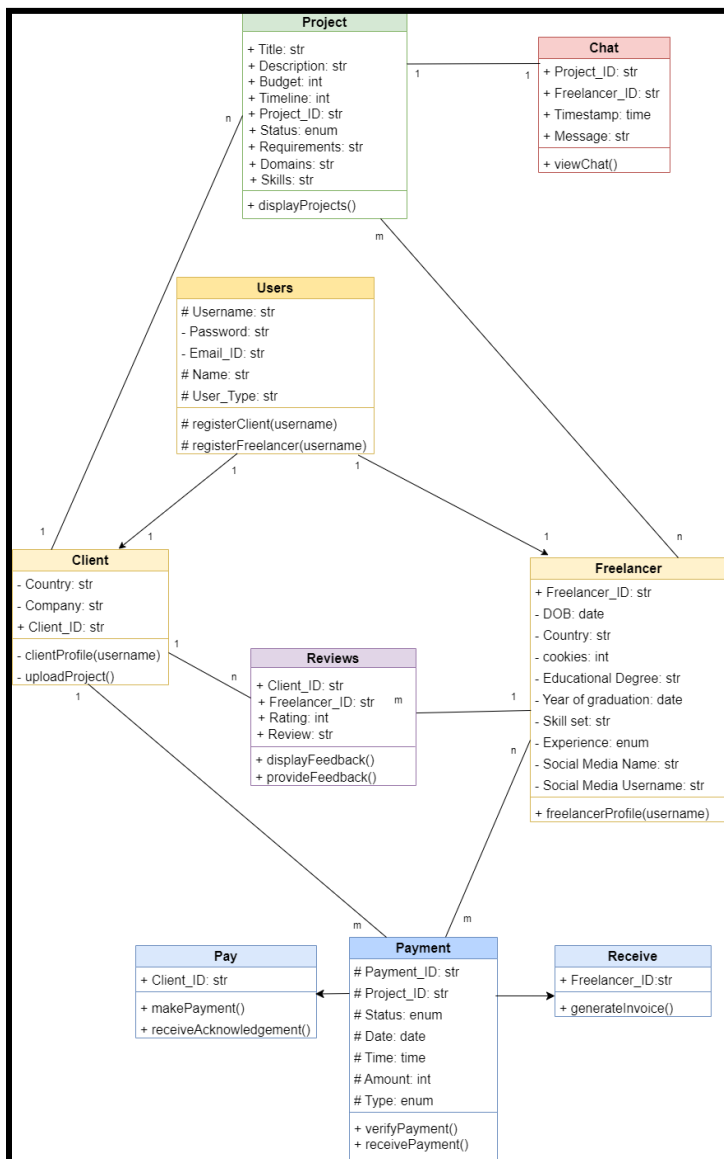PES2UG21CS396 – PREETHI M.

PES UNIVERSITY
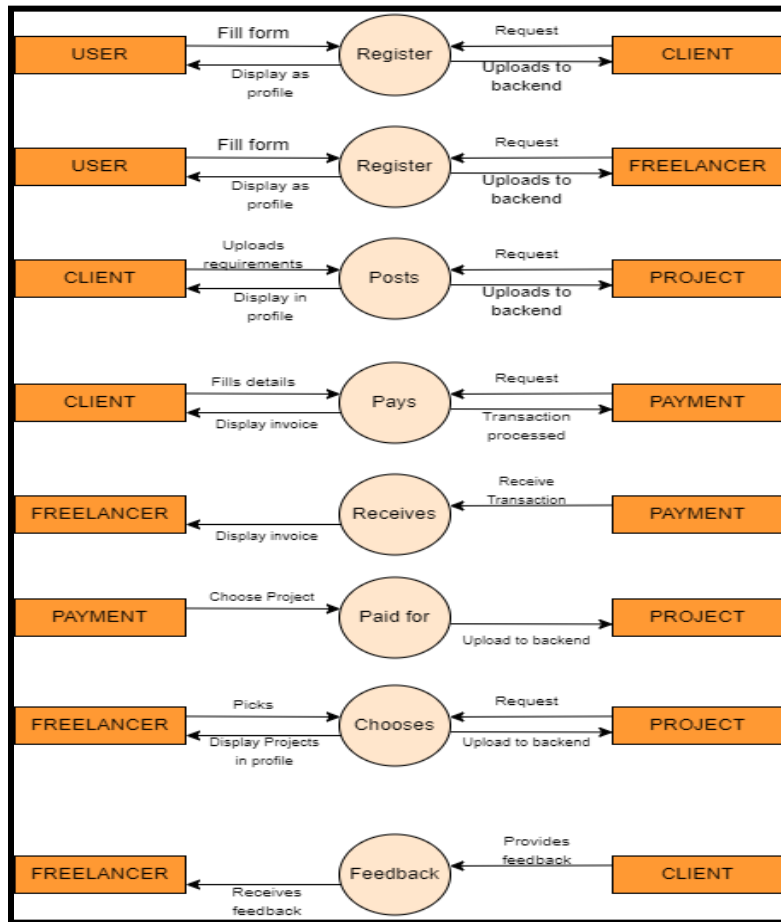
27/10/2023

1. USE CASE DIAGRAM

2. CLASS DIAGRAM:

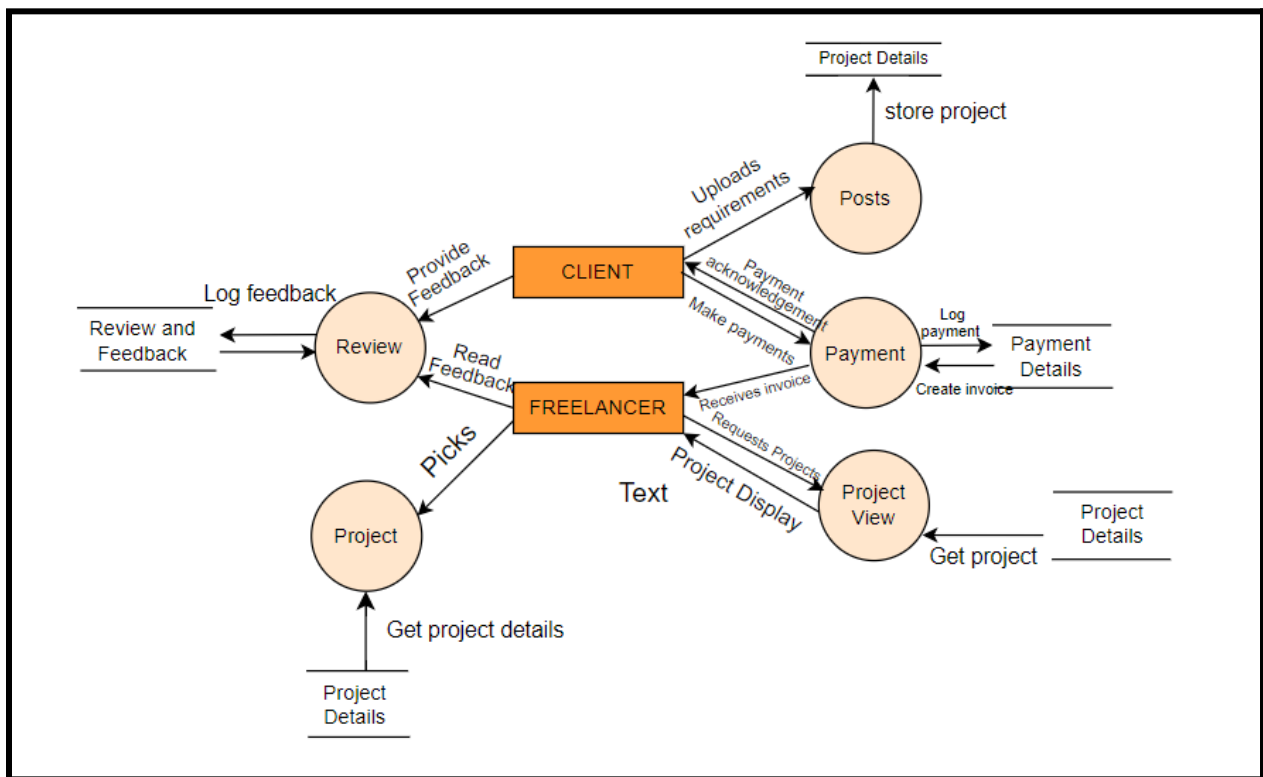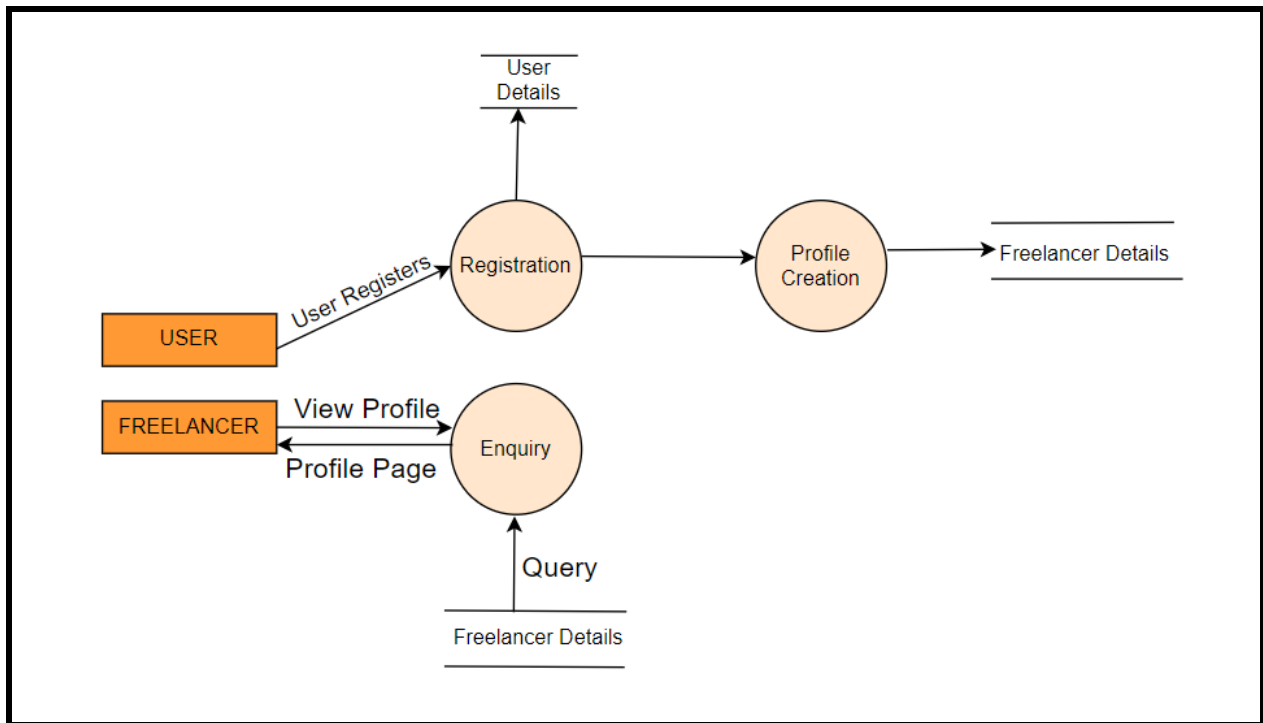# 3. TOP LEVEL DATA FLOW DIAGRAM

4. FIRST LEVEL DFD

Diagram 1

- User Details
- Registration
- Profile Creation → Freelancer Details
- USER — User Registers → Registration
- FREELANCER — View Profile → Enquiry
- Enquiry — Profile Page → FREELANCER
- Query
- Freelancer Details

Diagram 2

- Project Details
- store project
- Posts
- CLIENT — Uploads requirements → Posts
- CLIENT — Provide Feedback → Review
- Review — Log feedback → Review and Feedback
- Payment — Payment acknowledgement → CLIENT
- CLIENT — Make payments → Payment
- Payment — Log payment → Payment Details
- Payment — Create invoice ← Payment Details
- FREELANCER — Read Feedback → Review
- FREELANCER — Receives invoice ← Payment
- FREELANCER — Requests Projects → Project View
- Project View — Project Display → FREELANCER
- Project View — Get project ← Project Details
- Review — Picks → Project
- Project — Get project details ← Project Details
- Text

5. ARCHITECTURAL STYLE INTEGRATION

Client- server architecture will be used in Co-Lancer.

Features of client-server architecture:
Centralized Data Management: In this freelancer collaboration website, data of user profiles, project data, etc are collected. Centralized data storage on the server simplifies data management such that users can access consistent data securely. Enforcing data integrity, backup and security is made easier.

Security: The website deals with sensitive information, such as project details, financial transactions, and personal details. A server centralizes security measures, through which implementation of authentication, authorization, encryption, and other security protocols occur.

Collaboration and Real-Time Features: The website offers features like messaging and project updates. The server facilitates real-time communication and collaboration by acting as a central hub where clients exchange data. This ensures synchronization of messages and updates.

Consistency and Reliability: A server based architecture maintains consistency of user experience. Since clients access the same server, all users access the same data and features.

Maintenance and Updates: It is easier to update, fix bugs and add new features. A client-server architecture makes deployments easier. It allows for modification of the server independently without modifying the clients. It can ensure that all users have access to the latest version of the platform, reducing the need for users to update their individual platforms.

Cross-Platform Compatibility: Freelancers can use various devices and operating systems to access the website . A client-server architecture allows the users to have a consistent experience across any platform as the backend is accessed by the server so no specific client system platform is needed for it, making the website more accessible.

Data Backup and Recovery: A centralized server can be routinely backed up which makes it easier to recover the data in case of loss or system failures.