

American International University Bangladesh (AIUB)

**Department of Computer Science**

**Faculty of Science & Technology (FST)**

**Workify - Online Job Management System**

A Software Engineering Project Submitted

By

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Semester: Summer 24-25** | | **Section:** | **Group Number:** |  |
| **SL** | **Student Name** | **Student ID** | **Contribution (CO3+CO4)** | **Individual**  **Marks** |
| 1 | MD RAFIT RAHAD | 23-52476-2 | 33.33 |  |
| 2 | KAZI AYESHA AFIA RAYEESA | 23-52794-2 | 33.33 |  |
| 3 | SAFWAN YESFEE FAHIM | 23-50877-1 | 33.33 |  |

The project will be evaluated for the following Course Outcomes

|  |  |  |
| --- | --- | --- |
| ***CO3 (PO-g-1)***  ***Select appropriate software engineering models, project management roles and their associated skills for the complex software engineering project and evaluate the sustainability of developed software, taking into consideration the societal and environmental aspects*** | Total Marks | |
|  | |
| Selection of Software Engineering Models: Process model selection and presents sufficient evidence to support argument for the model selection | [5 Marks] |  |
| Role identification and Responsibility Allocation: Well-planned project with proper role identification and responsibility allocation in the project management activities | [5Marks] |  |
| Formatting and Submission: Submission, Defense, Completeness, Spelling, grammar,  and Organization of the Project report | [5Marks] |  |
| ***CO4 (PO-k-1)***  ***Apply engineering management principles and economic decision making to develop software engineering project management plan.*** | Total Marks | |
|  | |
| Project WBS and Testcases: Relevant WBS (project task list) and testcases for the proposed project are stated properly. | [5Marks] |  |
| Effort Estimation and Scheduling: Project estimation was described using proper effort estimation or schedules based on available project resources | [5Marks] |  |
| Risk Management: Sufficient and appropriate risks are identified, analyzed, and properly categorized or prioritized. | [5Marks] |  |

## Background to the Problem

## A group of people standing around a large screen AI-generated content may be incorrect.In today’s rapidly evolving digital era, freelancing has become one of the most popular work models worldwide, offering professionals the flexibility to work from anywhere while allowing businesses to access a global pool of talent. Many individuals are turning to freelancing as a way to build independent careers, explore diverse projects, and achieve a better work-life balance. At the same time, companies, startups, and entrepreneurs are increasingly looking for skilled freelancers to complete short-term projects, meet urgent deadlines, and reduce operational costs compared to hiring full-time staff. This growing demand for freelance services has led to the emergence of various online platforms designed to connect freelancers and clients, enabling collaboration in a virtual environment. However, despite the availability of popular platforms such as Upwork and Fiverr, the current market faces persistent challenges that make freelancing more complicated than it should be.

One of the major challenges with current freelance marketplaces is the high service fees that reduce freelancers’ earnings and make projects costly for clients. Complicated registration processes discourage new users from joining quickly, while limited communication tools make it difficult to share project requirements and updates clearly. Payment systems often lack transparency, leading to uncertainty about when freelancers will be paid and whether clients’ funds are secure. In addition, weak dispute resolution systems leave both parties without proper support when conflicts arise. These issues create barriers that harm trust, delay projects, and reduce the overall freelancing experience.

Our project, “Workify,” was developed by Ayesha, Rafit, and Safwan, who worked with the freelancer, client, and admin roles respectively to ensure a balanced and inclusive design. The platform focuses on simplicity and transparency, featuring an easy registration process, a clear and reliable payment system, and strong communication tools for effective collaboration. A structured dispute resolution process has also been designed to protect the interests of both clients and freelancers. Workify aims to create an affordable, trustworthy, and user-friendly freelance marketplace that allows clients to find skilled freelancers easily and freelancers to access quality projects with confidence.

## 1.2 Solution to the Problem and Process Model Selection

**a) Describe what your project scopes and features are.**

Workify is a web-based freelance marketplace that connects clients with freelancers to enable smooth hiring, collaboration, and payment.

Our main features include:

* User Management: Registration and profile creation for clients, freelancers, and admins.
* Job Posting: Clients can post both hourly and fixed-price jobs.
* Job Browsing: Freelancers can search and filter jobs based on their skills.
* Proposal System: Freelancers can submit proposals with their rates and timeline.
* Communication: Built-in chat system for project discussions.
* Contract Management: Formal hiring and contract acceptance process.
* Milestone Tracking: Progress monitoring for both clients and freelancers.
* Payment System: Secure payment release and receipt.
* Review System: Mutual feedback between clients and freelancers.
* Admin Controls: Platform moderation, user verification, and dispute resolution.

**b) User Story Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **As a/An** | **I want to** | **So that** | **Acceptance Criteria** |
| Client | register and create a profile | I can post jobs and hire freelancers | Client can successfully sign up with valid email and password, verify account, and create a complete profile with name, description, and payment method. |
| Client | post jobs (hourly or fixed) | freelancers can apply for them | Client can post a job with title, description, budget or hourly rate, and category. Job must be visible on the job board and open for proposals. |
| Client | review freelancer proposals | I can choose the best candidate | Client can view all proposals for a job, compare freelancer profiles, and filter or sort based on skills, price, and ratings. |
| Client | chat with freelancers | I can discuss project details before hiring | Client can start a secure chat with freelancers who have applied, exchange text and file attachments, and receive real-time notifications. |
| Client | hire freelancers | they can start working on my project | Client can send an offer to selected freelancer, freelancer can accept, and the contract becomes active. |
| Client | release payments | freelancers get paid after work is done | Client can review submitted work or milestones, approve them, and release funds from escrow to the freelancer’s account. |
| Client | leave reviews | I can provide feedback on freelancers’ work | Client can leave a rating (1–5 stars) and a written review after project completion, which becomes visible on freelancer’s profile. |
| Freelancer | register and create a profile | I can find and apply to jobs | Freelancer can successfully sign up, verify email, and create a profile with skills, experience, and payment details. |
| Freelancer | browse jobs | I can find work relevant to my skills | Freelancer can view all open jobs, filter by category, budget, and experience level, and see job details before applying. |
| Freelancer | submit proposals | I can get considered for projects | Freelancer can write and send a proposal with a cover letter, price, and delivery time. Proposal appears in client’s dashboard. |
| Freelancer | chat with clients | I can clarify job requirements | Freelancer can initiate or respond to chat messages from clients, send files, and receive instant notifications. |
| Freelancer | accept contracts | I can start working officially | Freelancer can review job offer details, accept or reject it, and gain access to the project workspace once accepted. |
| Freelancer | deliver milestones | client can track and review my progress | Freelancer can upload completed milestones, attach deliverables, and mark them as ready for client approval. |
| Freelancer | receive payments | I get paid for my work | Freelancer sees approved payments credited to their wallet/balance and can withdraw to a chosen payment method. |
| Freelancer | leave reviews | I can give feedback about the client | Freelancer can leave a star rating and short review about the client after job completion, which appears on client’s profile. |
| Admin | verify accounts | the platform remains secure | Admin can review user registration details, approve or reject suspicious accounts, and request additional verification documents. |
| Admin | monitor job posts and user profiles | I can ensure quality and compliance | Admin can view, edit, or remove job posts and profiles that violate policies. Notifications are sent to users when changes are made. |
| Admin | resolve disputes | I can help clients and freelancers fairly | Admin can view dispute details, review communication and files, and decide whether to refund client or release payment to freelancer. |
| Admin | manage platform fees | the platform earns revenue appropriately | Admin can set or update platform commission percentage, and system automatically applies it to transactions. |
| Admin | suspend or ban users | I can enforce platform rules | Admin can temporarily suspend or permanently ban users, blocking them from logging in and notifying them of the reason. |

**c) User Story Board (Trello)**

For our Trello board, we organized our user stories into the following columns:

* Backlog: All user stories waiting to be worked on.
* To Do: Authentication and basic setup features.
* In Progress: Dashboard and admin features.
* Testing: Job management, hiring, and communication features.
* A screenshot of a computer

  AI-generated content may be incorrect.Done: Payment and review features.

Trello link 🡪

<https://trello.com/invite/b/688addd0358c55f60856271a/ATTI7e4399406930e3aa48bb8c1faad299b2EFC55EA0/online-job-management>

**d. What are the existing software solutions that are available to solve mentioned problem?**

Several platforms already exist in the freelance marketplace:

1. Upwork: The most popular platform but charges high fees (3-20% commission)
2. Fiverr: Good for small gigs but limited for long-term projects
3. Freelancer.com: Similar to Upwork but with complex bidding system
4. Guru: Focuses on professional services but has limited user base
5. 99designs: Specialized for design work only

These platforms have common problems like high fees, complex interfaces, and poor customer support. Our solution addresses these issues by offering lower fees, simpler design, and better user experience.

**e. Select one software development process model from the models taught in the course that best suits your project.**

We selected the Incremental Process Model, specifically Extreme Programming (XP), for our Workify project. XP allows us to develop the software in small, frequent increments, delivering working features early and continuously. This approach emphasizes customer feedback, pair programming, and test-driven development, ensuring high code quality, adaptability to changing requirements, and faster delivery of value to users.

**f. Provide an analysis of the project environment, including the nature of requirements and whether they are stable or likely to change.**

* Requirements Nature & Stability:

Core requirements like user registration, job posting, and payments are well-defined and stable. However, UX-related features (UI design, notifications, filtering options) are expected to evolve as we gather user feedback. Third-party integration (e.g., payment gateways) may change based on business needs.

* Technology Environment:  
  The project is a web-based platform, built with modern web technologies (HTML, CSS, JavaScript, Node.js/PHP). Database design is stable, but APIs and deployment setups allow flexibility for scaling and adding features in future iterations.
* Team Environment:  
  We have a small team (4–5 members) with mixed experience levels. XP practices like pair programming, stand-up meetings, and continuous integration will help maintain communication, share knowledge, and keep everyone aligned.

**g. Explain how the selected model supports your team size, communication and coordination of tasks. Is the solution feasible to meet the business objective?**

Extreme Programming (XP) supports our team size, communication, and task coordination effectively, making the solution feasible to meet business objectives.

Team Size (4–5 people):  
XP works best with small teams, making it ideal for our group. Practices like pair programming and collective code ownership allow 1–2 developers to handle each increment efficiently. This reduces conflicts, keeps everyone aligned, and ensures consistent code quality.

Communication:  
XP emphasizes constant communication through daily stand-ups, on-site customer involvement, and frequent feedback loops. Regular iteration reviews keep progress transparent, while continuous integration ensures everyone sees updates in real time.

Task Coordination:  
XP uses small iterations with clearly defined user stories and deliverables, making task distribution simple and manageable. Test-Driven Development (test driven development) and continuous testing reduce rework and catch issues early, keeping dependencies between features under control.

Business Feasibility:  
XP’s incremental delivery ensures that we produce a usable version of Workify after every iteration. This means stakeholders get value early, and we can adjust features based on feedback, reducing the risk of building something misaligned with business needs. Even under time constraints, a minimal but functional version will be ready.

**h. Evaluate how flexible the model is in adapting to changes in scope, technology, or user requirements.**XP adapts quickly through short iterations and reprioritization of user stories. New technologies or tools can be introduced between iterations, and customer feedback shapes each release, keeping the product aligned with evolving needs.

**i. Provide deep insight that demonstrate and present a creative solution to the real-life problem.**We go beyond a typical freelance platform by using skill-based job suggestions, clear upfront pricing, and milestone tracking to build trust. Dispute resolution is kept simple, and freelancers can join without any bidding fees reducing barriers for new talent.

**j. Describe the target group of users of your solution? And how will they benefit from your proposed solution to the problem?**

* Small Businesses: Get affordable talent without long-term hiring
* Freelancers: Earn more due to lower platform fees.
* Students: Gain flexible, skill-building projects.  
   Overall, users save time, reduce costs, and work with transparency.

**k. Describe the contribution of your project to the development of scientific results that are identified and well documented.**Our project tests XP in a real academic setting, documenting its efficiency for small teams. We collect data on user behavior, performance under load, and matching accuracy, turning them into case studies for future research in software engineering.

**l. Present enough evidence to support argument for your model selection in developing your proposed solution.**XP matches our project’s incremental nature, allows rapid feedback cycles, and supports frequent testing. This ensures each feature is usable and reduces rework. Other models like Waterfall or Spiral would slow down delivery or overcomplicate planning.

**m. Discuss how the process model manages project risks and uncertainties at different stages.**XP reduces risks by focusing on early delivery of high-value features, using pair programming and test-driven development to prevent defects, and running continuous integration to catch issues immediately. Iteration reviews keep requirements aligned and minimize surprises.

**n. Relate the process model to the project schedule, showing how it supports timely delivery and meeting deadlines.**Each XP iteration delivers working software within a few weeks, ensuring visible progress. If delays occur, less critical stories can be moved to later iterations without affecting core functionality, keeping deadlines realistic.

**o. Present a justification that explains clearly why the chosen model is more suitable than other alternatives.**XP balances structure and speed, letting us ship quality software quickly while responding to change. It avoids Waterfall’s rigidity, Agile’s overhead, and Prototyping’s lack of completeness making it the most practical choice for our team size and timeline.

## 1.3 Project Role Identification and Responsibilities:

**a) Identify the main roles involved in the project, covering both development and management aspects.**Our project team consists of four main roles: Project Manager, Developer, Tester/QA, and UI/UX Designer. These roles collectively manage planning, implementation, testing, and delivery.

**b) Describe the responsibility of each role in key stages such as requirements gathering, design, implementation, testing and deployment.**The Project Manager leads requirements gathering, schedules iterations, and oversees deployment. Developers focus on coding, integration, and database management. Testers perform continuous testing to ensure quality, while the UI/UX Designer works on mockups, user flows, and interface design before development starts.

**c) Specify which roles are responsible for decision making, quality assurance and resource management.**Decision-making is led by the Project Manager, with input from developers during planning sessions. Quality assurance is handled by the Tester/QA, while developers use TDD to support it. Resource management, including workload balancing and time tracking, is also managed by the Project Manager.

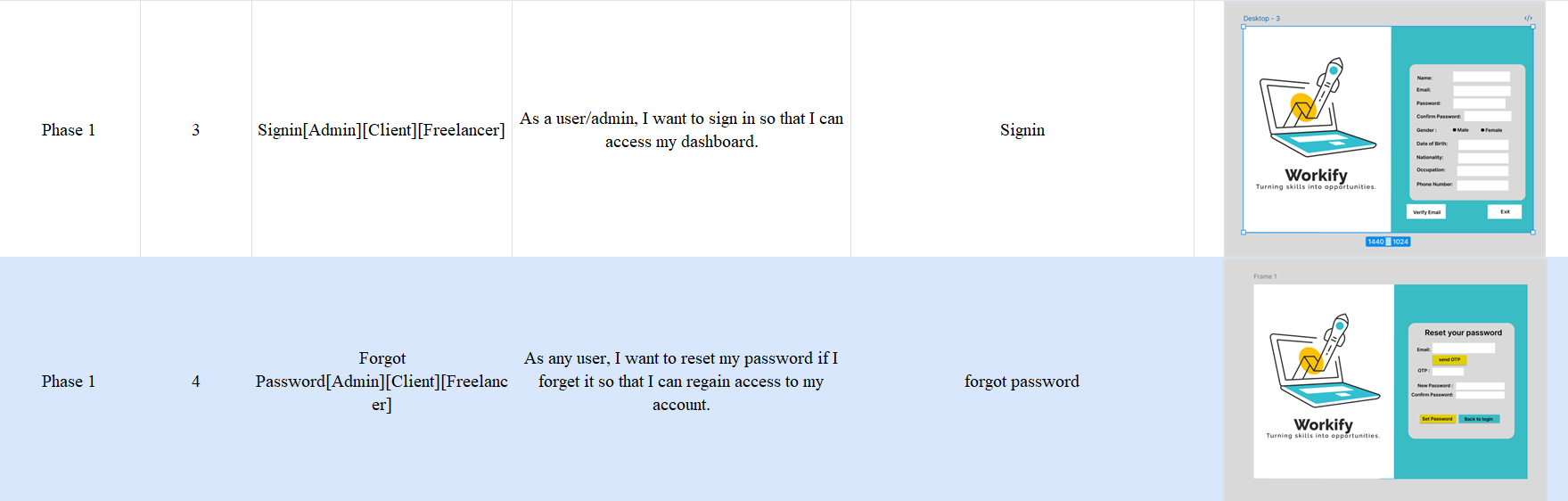
**d) Explain how responsibilities are distributed among the team members and justify the allocation based on skills and expertise.**Tasks are divided based on strengths developers handle technical implementation, the designer ensures usability, and the tester maintains product quality. This distribution minimizes rework and leverages each member’s expertise. XP practices like pair programming ensure shared knowledge across the team.

## 2. SOFTWARE REQUIREMENTS SPECIFICATIONS (SRS) / PRODUCT REQUIREMENTS DOCUMENT (PRD)

## 2.1 Functional Requirements

**PRD TABLE LINK🡪**

<https://aiubedu60714-my.sharepoint.com/:x:/g/personal/23-52794-2_student_aiub_edu/EeVzQ1M2VdhKutvcAQSMkHEBS1NafnzF5zPNU5aUFT0INg?e=BWOeL6&nav=MTVfezAwMDAwMDAwLTAwMDEtMDAwMC0wMDAwLTAwMDAwMDAwMDAwMH>

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

## 2.2 Non-Functional Requirements

**a) Performance:**The platform must load pages within 2–3 seconds and support at least 100 concurrent users without noticeable lag.

**b) Reliability:**System uptime should be 99% or higher, ensuring job postings, chats, and payments are consistently available.

**c) Integrity/Security:**

User passwords must be encrypted, data should be stored securely, and financial transactions must be processed over HTTPS with proper authentication and authorization controls.

**d) Usability:**The UI should be intuitive, mobile-responsive, and allow job posting and proposal submission within minimal steps to enhance user experience.

**e) Maintainability:**The system should use clean, modular code with documentation so future teams can easily add new features or fix bugs.

**f) Scalability:**The architecture should support growth by allowing additional servers, databases, or microservices without major redesigns to handle more users and projects in the future.

## 3.1 Effort and Cost Estimation

The project scopes for clients and freelancers are registration, job posting, proposal submission, hiring, messaging, payments and reviews. Additionally, admin scopes are monitoring jobs, verifying clients and users.

**LOC Estimation:**

Total lines of Code for the entire project are 7,500 LOC

Effort Estimation=E = [LOC x B0.333/P]3 x (1/t4)

E= [7500x 1^0.333/600] ^3 x (1/3^4)

E=24.11

**Using COCOMO:**  
SLOC = 7500  
P = 1.12  
t = 0.35  
Coefficient = 3  
The project type is Semi ditached.

Effort (PM) = Coefficient × (SLOC/1000) ^P  
PM = 3\*(7500/1000) ^1.12  
PM = 28.654

Development time (DM) = 2.50 × (PM)^t  
DM = 2.5\*(28.654) ^0.35  
DM = 8.09

Required number of people (ST) = PM / DM  
ST = 28.654/8.09  
ST = 3.54≈4

**Make Buy Decision Tree:**

A diagram of a job application system

AI-generated content may be incorrect.

Computing expected cost from decision tree:

Expected cost= (path probability \* estimated path cost)

The expected cost to build is: expected cost= 0.35 x 350k +0.65 x 450k = 415k

The expected cost to reuse is: expected cost= 0.2 x 250k +0.8 x 480k = 434k

The expected cost to buy is: expected cost= 0.4 x 230k +0.60 x 370k = 314k

The expected cost to contract is: expected cost= 0.6 x 320k +0.4 x 480k = 384k

Assumptions and possible variation in results, since estimation always involves some level of uncertainty are mentioned bellow:

Assumptions: LOC Estimation assumes that the productivity rate (P) of 600 LOC per person per hour remains constant throughout the project. The formula used assumes uniform effort distribution and does not account for potential spikes in workload during critical phases. COCOMO Model assumes the project is semi-detached meaning that the project has moderate complexity and requires a moderate level of coordination between different team members. Make/Buy Decision Tree assumes that all costs provided are accurate and up-to-date, and that the project will follow the selected path with the given probability distributions.

Possible Variations in Results: LOC estimation can vary depending on team efficiency the complexity of tasks and the actual number of LOC that will be written in the actual development process. COCOMO model can vary based on the project’s actual complexity. If the project turns out to be more complex requires additional resources so the coefficients might need to be adjusted and the actual development time will be longer or shorter. Make-buy decision tree cost predictions depend on market conditions and assumptions about the probability distribution of costs. So price may changes and risk factors could lead to higher or lower costs than estimated.

## 3.2 Project Scheduling

Tasks are broken down into smaller, manageable user stories. These stories are prioritized based on customer value, and development happens in short iterations**.** The tasks are connected and responsibilities assigned also . These are shown in the bellow :

Gathering & User Stories

Task 1.1: Collect user stories from the client and team.

Task 1.2: Prioritize user stories for the next iteration.

Responsible: Customer, Product Owner.

Design and Architecture

Task 2.1: Design a simple architecture that can evolve.

Task 2.2: Create initial wireframes for the UI.

Responsible: System Architect, UX/UI Designer.

Development (XP Iterations)

Task 3.1: Implement highest-priority user stories.

Task 3.2: Code in pairs (pair programming).

Task 3.3: Write automated tests for all new code.

Responsible: Developers, Tester.

Testing

Task 4.1: Perform continuous integration and testing.

Task 4.2: Run automated tests after every small change.

Responsible: Developer, Tester.

Deployment and Feedback

Task 5.1: Deploy after every iteration for feedback.

Task 5.2: Gather customer feedback and adjust.

Responsible: Product Owner.

Refactoring and Continuous Improvement

Task 6.1: Refactor code continuously based on feedback.

Responsible: Developers.

**A diagram of a diagram

AI-generated content may be incorrect.**

**Time Line chart:**

**A screenshot of a computer

AI-generated content may be incorrect.**

***Major deliverables tied to milestones and ensure outputs are clearly defined bellow:***

Milestone 1: Complete Requirements and Analysis (3 weeks)

Deliverable: Requirements Document ,User Stories and prioritize user stories

Milestone 2: Finalize System and UI Design (4 weeks)

Deliverable: Architecture diagrams, wireframes, and design specifications.

Milestone 3: Development Phase Completion (5 weeks)

Deliverable: A working and integrated version of the platform .

Milestone 4: Testing Completion (3 weeks)

Deliverable: Test case results, quality assurance reports, and any necessary fixes.

Milestone 5: Deployment (2 week)

Deliverable: Fully deployed and operational website

Milestone 6: Maintenance and Ongoing Support (1 week)

Deliverable: Ongoing Bug Fixes and User Feedback Integration

To ensure the project stays on track:

Weekly Team Meetings: Discuss progress, roadblocks, and next steps.

Bi-weekly Milestone Reviews: Review progress at each milestone to ensure deliverables are met.

Schedule Checks: Review timelines regularly, ensuring the project is meeting deadlines or adjusting as needed.

**Earned Value Analysis (EVA) Table:**

|  |  |  |
| --- | --- | --- |
| Task No | Planned Effort | Actual Effort |
| 1 | 8.5 | 7.4 |
| 2 | 7 | 6.66 |
| 3 | 7.5 | 7 |
| 4 | 7.8 | 8 |
| 5 | 5 | 4.65 |
| 6 | 6.66 | 7.52 |
| 7 | 6 | 5.45 |
| 8 | 5 | 4.34 |
| 9 | 5 | 6 |
| 10 | 4 | 4.5 |
| 11 | 5.7 | 6.5 |
| 12 | 7.7 | 8 |
| 13 | 6 | 5.85 |
| 14 | 7.54 | 7 |
| 15 | 6 | 5.7 |
| 16 | 7.7 | 6.7 |
| 17 | 8 | 8.9 |
| 18 | 7.34 | 8 |
| 19 | 5 | 4.33 |
| 20 | 6 | 6.7 |
| 21 | 6 | 6.16 |
| 22 | 8 | 7.6 |
| 23 | 4.5 | 4 |
| 24 | 7 | 7.8 |
| 25 | 6.5 | 6.7 |
| 26 | 6 | 7.5 |
| 27 | 5 | 6 |
| 28 | 8 | 7.9 |
| 29 | 7.5 | 8 |
| 30 | 6 | 5.57 |
| 31 | 7.8 | 8.75 |
| 32 | 8 | 7.8 |
| 33 | 5 | 4.34 |
| 34 | 7 | 6.66 |
| 35 | 7.8 | 7 |
| 36 | 6 | 5 |
| 37 | 5 | 5.75 |
| 38 | 8 | 7.44 |
| 39 | 6 | 6.45 |
| 40 | 4 | 3.77 |
| 41 | 7.77 | 8 |
| 42 | 8.85 | 8 |
| 43 | 7.8 | 9.44 |
| 44 | 8 | 8.7 |
| 45 | 7 | 6.15 |
| 46 | 6.66 | 7.66 |
| 47 | 6 | 5 |
| 48 | 6 | 5.8 |
| 49 | 6.5 | 6.36 |
| 50 | 5 | 4.76 |
| 51 | 7.77 | 8.3 |
| 52 | 6 | 6.8 |
| 53 | 5.55 | 6 |
| 54 | 7 | 6.66 |
| 55 | 7.88 | 8 |
| 56 | 7.44 | 7 |
| 57 | 6 | 5.3 |
| 58 | 7 | 5.65 |
| 59 | 5.66 | 6 |
| 60 | 6 | 6.54 |
| 61 | 6.77 | 6 |
| 62 | 5 | 4 |
| 63 | 6.7 | 7.4 |
| 64 | 7 | 7.45 |
| 65 | 6.66 | 5.5 |
| 66 | 5.55 | 7 |
| 67 | 6 | 5.55 |
| 68 | 6 | 6.33 |
| 69 | 7 | 8 |
| 70 | 7 | 6.78 |
| 71 | 5 | 5.55 |
| 72 | 6 | 5.44 |
| 73 | 6.78 | 6 |
| 74 | 6.66 | 8.2 |
| 75 | 5 | 5 |
| 76 | 8 | 9 |
| 77 | 8 | 6.8 |
| 78 | 6 | 5 |
| 79 | 6.64 | 5.77 |
| 80 | 6.5 | 6.7 |
| 81 | 5 | 4.44 |
| 82 | 6.66 | 7 |
| 83 | 8 | 7.89 |
| 84 | 6.22 | 8 |
| 85 | 6.33 | 5 |
| 86 | 5 | 6.44 |
| 87 | 7 | 8 |
| 88 | 5.4 | 6.5 |
| 89 | 7.8 | 7.5 |
| 90 | 5.9 | 6 |
| 91 | 8 | 8.5 |
| 92 | 5.75 | 4 |
| 93 | 6.9 | 5.06 |
| 94 | 6.4 | 8.23 |
| 95 | 6.7 | 7.34 |
| 96 | 7.5 | 8.7 |
| 97 | 7 | 7 |
| 98 | 7 | 6.66 |
| 99 | 5.55 | 4.77 |
| 100 | 6 | 6.8 |
| 101 | 8 | 7 |
| 102 | 5.55 | 4 |
| 103 | 5.55 | 5.89 |
| 104 | 7.7 | 5.55 |
| 105 | 6 | 7.6 |
| 106 | 7.52 | 6.33 |
| 107 | 6.64 | 7.5 |
| 108 | 7.7 | 6.88 |
| 109 | 7.16 | 6.7 |
| 110 | 4 | 4.88 |
| 111 | 5 | 6 |
| 112 | 6 | 5.57 |
| 113 | 7 | 8.4 |
| 114 | 5.55 | 5.55 |
| 115 | 6.5 | 5.6 |
| 116 | 5.8 | 6.78 |
| 117 | 7.89 | 8 |
| 118 | 5.57 | 5.7 |
| 119 | 6.6 | 6 |
| 120 | 7.54 |  |
| 121 | 7.3 |  |
| 122 | 5.41 |  |
| 123 | 6.4 |  |
| 124 | 4.5 |  |
| 125 | 6 |  |
| 126 | 5.65 |  |
| 127 | 8 |  |
| 128 | 8.5 |  |
| 129 | 7.65 |  |

|  |
| --- |
|  |
| Effort Estimated=960 Person Day |
| Total task=160 |
| BCWP=776.74 |
| BCWS=841.47 |
| ACWP=774.52 |
| BAC=960 |
| SPI=BCWP/BCWS  =776.74/841.47  =1.0029 |
| SV=BCWP-BCWS  =776.74-841.47  =-64.73 |
| CPI=BCWP/ACWP  =776.74/774.52  =0.973 |
| CV=BCWP-ACWP  =776.74-774.52  =2.22 |
| %schedule for completion=BCWS/BAC  = (841.47/960) x100  =87.65% [ % of work scheduled to be done at this time ] |
| %complete=BCWP/BAC  = (776.74/960) x100  =80.91% [ % of work completed at this time ] |
|  |

Common causes of delays were recognized and plans were done accordingly, these issues are written bellow:

Unrealistic Deadlines: Iterative progress means deadlines are adjustable and estimates are recalibrated after each iteration.

Changing Requirements: For XP welcomes changes from user and stories are continuously prioritized and adjusted.

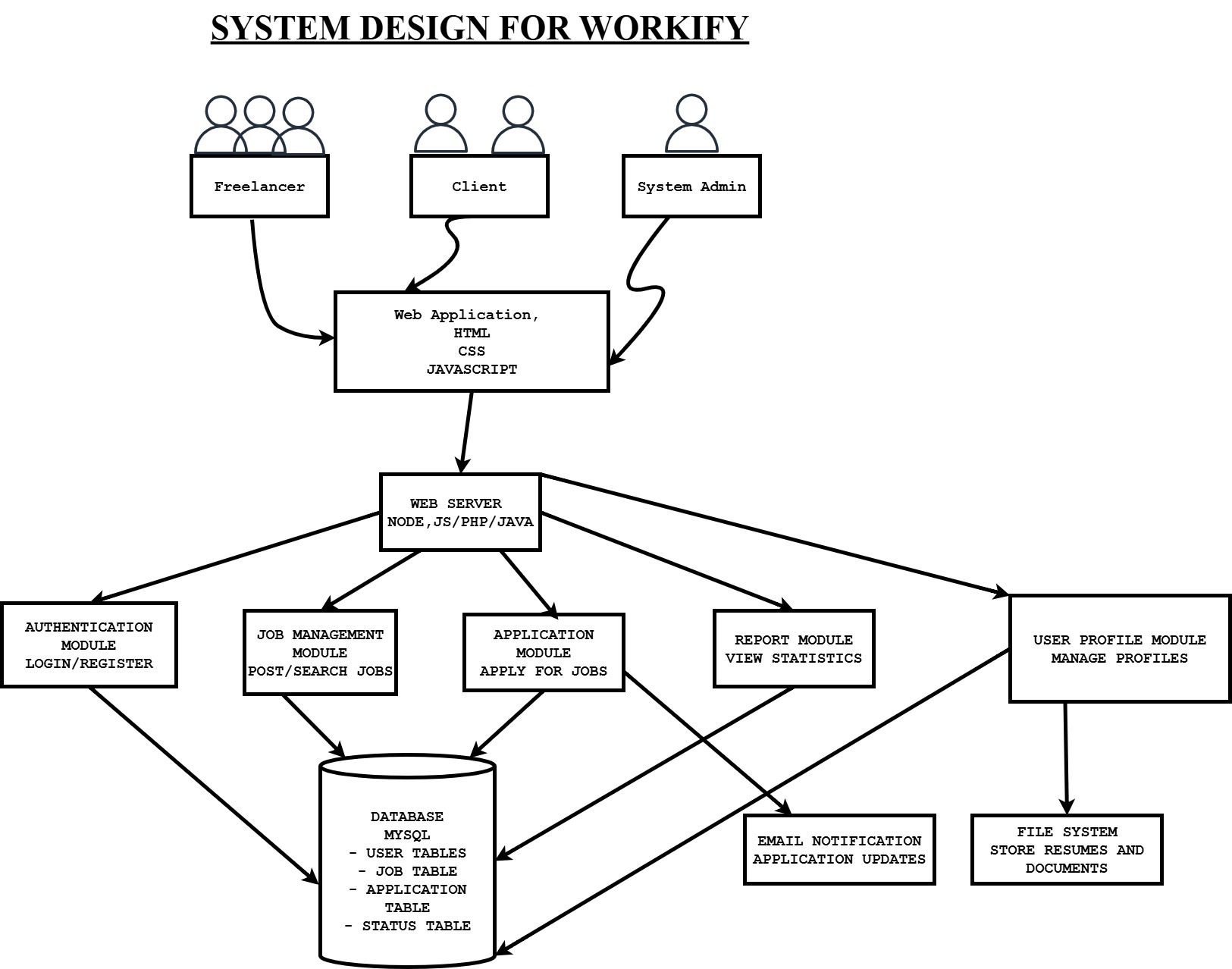
Risks: Risks are tracked continuously through regular feedback from clients and developers.

Technical or Human Issues: Pair programming and continuous testing help identify issues early.

Miscommunication: Daily standups and constant client feedback ensure clear communication throughout.

## 4.1 System Design Diagrams

**Data Flow Diagram**



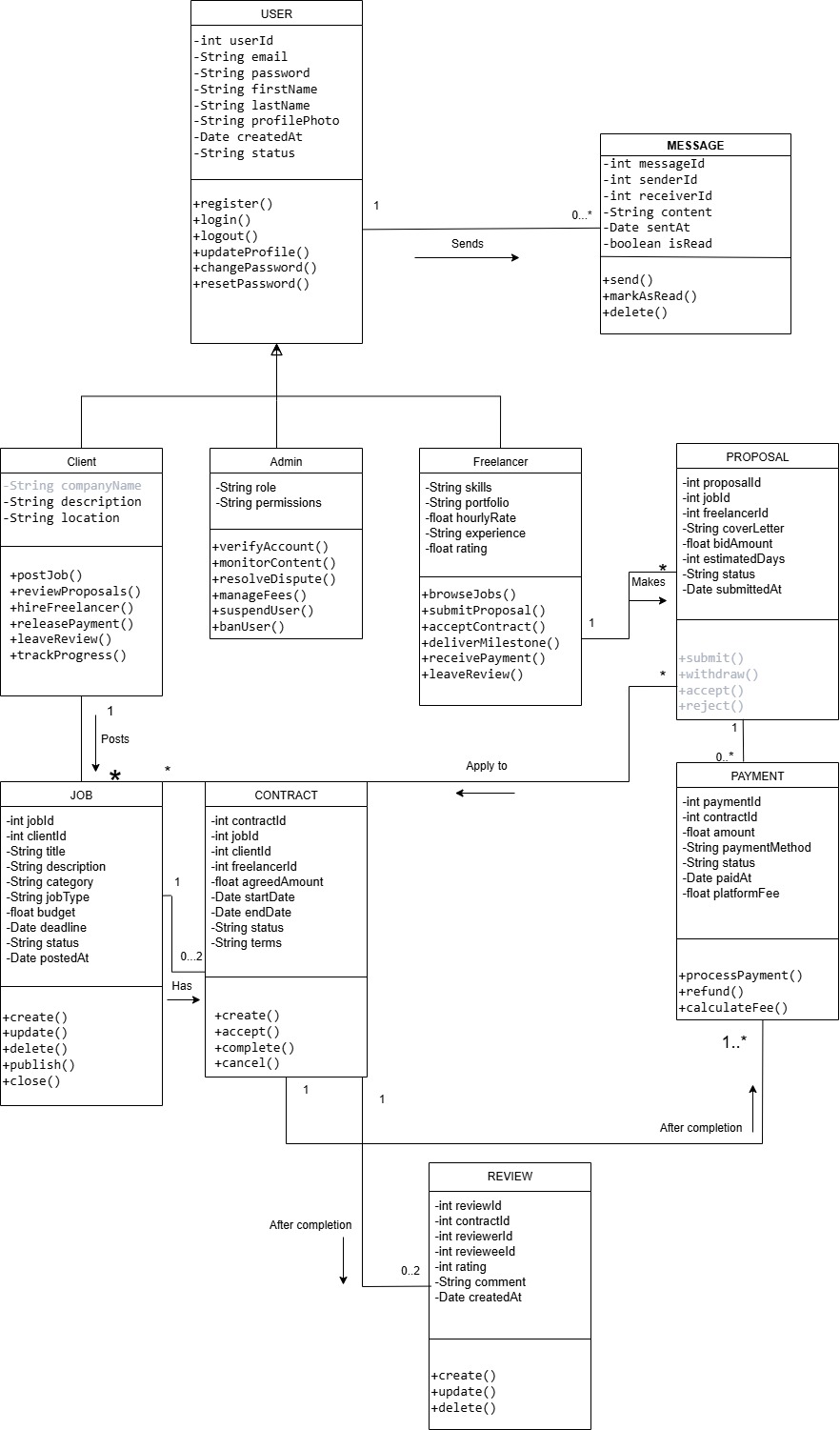
Draw.io Link 🡪 <https://drive.google.com/file/d/1PxLY89C4auXlVU-6LrSR0oggVwKpS9_R/view?usp=sharing>

A diagram of a diagram

AI-generated content may be incorrect.

Draw.io Link 🡪 <https://drive.google.com/file/d/1SRZi2mwkA4ZPPOvo0VTqBhtIo6w6tHjb/view?usp=sharing>

**Use Case Diagram**



Draw.io Link 🡪 <https://drive.google.com/file/d/13xmhkutNoxLyaBQyabvX3ChR1lgbg06y/view?usp=sharing>

**Class Diagram**

## A diagram of a flowchart AI-generated content may be incorrect. 4.2 [Figma UI]

Draw.io Link 🡪 <https://drive.google.com/file/d/1QqO1iGbHu9tE-FJmG6JkOHKec1no3g6S/view?usp=sharing>

**Activity Diagram**

Figma link 🡪 <https://www.figma.com/design/3U1iV7WDkfossE46AMkOEp/Workify?node-id=0-1&t=lJglMWgDEnPHug13-1>

|  |  |
| --- | --- |
|  |  |

A screenshot of a chat

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Testing

**As a Client:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| * **Project Name:** Online Job Management System | | | **Test Designed by:** Md Rafit Rahad | | |
| **Test Case ID:** TC\_1 | | | **Test Designed date**: 26/09/2025 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Md Rafit Rahad | | |
| **Module Name:** Client – Signup Session | | | **Test Execution date:** 26/09/2025 | | |
| **Test Title:** Verify signup with valid client details | | |  | | |
| **Description:** Test the client signup page and profile creation flow. | | |  | | |
| **Precondition:** User is on the signup page; email is not already registered. | | | | | |
| **Dependencies:** Email service for OTP/verification | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Go to the site 2. Navigate to Client signup 3. Enter full name, email, password, company (optional) 4. Agree to Terms and click Create Account 5. Verify email (if prompted) and auto-login to dashboard | Full name: Client name Email: client@example.com Password: StrongP@ssw0rd Company: N/A | Client accounts are created and users are redirected to login page. | | As expected | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by:** Md Rafit Rahad | | |
| **Test Case ID:** TC\_2 | | | **Test Designed date**: 26/09/2025 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Md Rafit Rahad | | |
| **Module Name:** Client –Post Job | | | **Test Execution date:** 26/09/2025 | | |
| **Test Title:** Verify validations for posting a job | | |  | | |
| **Description:** Ensure required fields like Title, Description, Job type, Budget are validated for jobs. | | |  | | |
| **Precondition:** Client is logged in and post from job page. | | | | | |
| **Dependencies:** All data’s are input. | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Open Post Job form 2. Enter Title and Description 3. Select Job Type = Fixed 4. Leave Budget empty 5. Click Post Job | Title: Build landing page Description: 3-section responsive landing page Job Type: Fixed Budget: (empty) | Form shows validation error for Budget and prevents submission. | | Job is posted successfully without a budget (BUG). | Fail |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by:** Md Rafit Rahad | | |
| **Test Case ID:** TC\_3 | | | **Test Designed date**: 26/09/2025 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Md Rafit Rahad | | |
| **Module Name:** Client –Post Job | | | **Test Execution date:** 26/09/2025 | | |
| **Test Title:** Verify validations for posting a job | | |  | | |
| **Description:** Ensure required fields like Title, Description, Job type, Budget are validated for jobs. | | |  | | |
| **Precondition:** Client is logged in and post from job page. | | | | | |
| **Dependencies:** All data’s are input. | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Open Post Job form 2. Enter Title and Description 3. Select Job Type = Fixed 4. Budget cannot be empty 5. Click Post Job | Title: Build landing page Description: 3-section responsive landing page Job Type: Fixed Budget: Range is entered. | Job post is created and client is in client dashboard. | | As expected. | Pass. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by:** Md Rafit Rahad | | |
| **Test Case ID:** TC\_4 | | | **Test Designed date**: 26/09/2025 | | |
| **Test Priority (Low, Medium, High):** Medium | | | **Test Executed by:** Md Rafit Rahad | | |
| **Module Name:** Client – Review Proposals | | | **Test Execution date:** 26/09/2025 | | |
| **Test Title:** Verify client can verify, view and filter proposals for a job | | |  | | |
| **Description:** Client can open a posted job, view proposals and filter the proposals only for the job post done by him/her. | | |  | | |
| **Precondition:** At least one job has proposals submitted by freelancers. | | | | | |
| **Dependencies:** Job post has to be done by the client. | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Navigate to My Jobs 2. Open a job with proposals 3. View proposal list with freelancer details and bid 4. Apply filter: Sort by Lowest Bid 5. Open a proposal to view full details | Job: Build landing page Filter: Sort by Lowest Bid | Proposals load with correct details; filter applies and reorders list accordingly. | | As expected | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by:** Md Rafit Rahad | | |
| **Test Case ID:** TC\_5 | | | **Test Designed date**: 26/09/2025 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Md Rafit Rahad | | |
| **Module Name:** Client Hire Freelancer | | | **Test Execution date:** 12/08/2025 | | |
| **Test Title:** Verify client can hire a selected freelancer from a proposal | | |  | | |
| **Description:** Client should be able to start contract by sending an offer and receiving acceptance. | | |  | | |
| **Precondition:** Client has at least one proposal, payment method has been settled. | | | | | |
| **Dependencies:** Contract for the job service. | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Open selected proposal 2. Send offer to freelancer 3. Set contract terms scope, amount, milestones 4. Verify contract. | Contract Type: Fixed Amount: $300 Milestones: Final Delivery | Offer is sent; upon freelancer acceptance, contract shows under Active Contracts. | | As expected | Pass |

**As a Freelancer:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name: Workify** | | | **Test Designed by: Kazi Ayesha** | | |
| **Test Case ID: TC\_1** | | | Test Designed date: 23/09/2025 | | |
| **Test Priority : High** | | | Test Executed by: Tester | | |
| **Module Name: Freelancer Registration** | | | Test Execution date: 26/09/2025 | | |
| **Test Title: Freelancer Registration with Valid Data** | | |  | | |
| **Description: Test freelancer signup process** | | |  | | |
| **Precondition: The user has a valid email** | | | | | |
| **Dependencies: Database connection** | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status** |
| 1. Go to registration page | N/A | Page loads | | Page loaded successfully | Pass |
| 2. Enter name | John Smith | Field accepts input | | Name entered correctly | Pass |
| 3. Enter email | john@email.com | Valid email format | | Email accepted | Pass |
| 4. Enter password | Pass123! | Strong password accepted | | Password accepted | Pass |
| 5. Select skills | Web Development | Skills selected | | Skills added to profile | Pass |
| 6. Click Register | N/A | Account created successfully | | Registration successful, welcome email sent | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name: Workify** | | | **Test Designed by: Kazi Ayesha** | | |
| **Test Case ID: TC\_2** | | | Test Designed date: 23/09/2025 | | |
| **Test Priority : High** | | | Test Executed by: Tester | | |
| **Module Name: Job Application** | | | Test Execution date: 26/09/2025 | | |
| **Test Title: Browse Jobs and Submit Proposal** | | |  | | |
| **Description: Test job browsing and proposal submission** | | |  | | |
| **Precondition: Freelancer is logged in** | | | | | |
| **Dependencies: Active job postings exist** | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status** |
| 1. Click Browse Jobs | N/A | Job listings display | | 15 jobs displayed | Pass |
| 2. Filter by Web Development | Filter: Web Dev | Filtered results show | | 8 relevant jobs shown | Pass |
| 3. Click job title | Job: E-commerce Site | Job details open | | Full job description visible | Pass |
| 4. Click Submit Proposal | N/A | Proposal form opens | | Form displayed with all fields | Pass |
| 5. Enter bid amount | $800 | Amount accepted | | Bid amount set successfully | Pass |
| 6. Write cover letter | "I have 3 years..." | Text entered | | Cover letter saved | Pass |
| 7. Submit Proposal | N/A | Proposal sent | | Proposal submitted, confirmation shown | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name: Workify** | | | **Test Designed by: Kazi Ayesha** | | |
| **Test Case ID: TC\_3** | | | Test Designed date: 23/09/2025 | | |
| **Test Priority : Medium** | | | Test Executed by: Tester | | |
| **Module Name: Client Communication** | | | Test Execution date: 26/09/2025 | | |
| **Test Title: Chat with client** | | |  | | |
| **Description: Test messaging Functionality** | | |  | | |
| **Precondition: Chat conversation exists** | | | | | |
| **Dependencies: Client has sent message** | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status** |
| 1. Go to Messages | N/A | Messages page loads | | Inbox displayed with 3 conversations | Pass |
| 2. Select client chat | Client: ABC Corp | Chat thread opens | | Conversation history visible | Pass |
| 3. Read client message | N/A | Message visible | | "Can you start Monday?" message shown | Pass |
| 4. Type response | "Yes, I can start" | Text appears in box | | Message typed successfully | Pass |
| 5. Send message | N/A | Message sent | | Message delivered, timestamp shown | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name: Workify** | | | **Test Designed by: Kazi Ayesha** | | |
| **Test Case ID: TC\_4 [failed]** | | | Test Designed date: 23/09/2025 | | |
| **Test Priority : High** | | | Test Executed by: Tester | | |
| **Module Name: Contrast and Milestone** | | | Test Execution date: 26/09/2025 | | |
| **Test Title: Accept Contract and Submit Milestone** | | |  | | |
| **Description: Test contract acceptance and work delivery** | | |  | | |
| **Precondition: Contract offer received** | | | | | |
| **Dependencies: Payment system active** | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status** |
| 1. Go to Contracts | N/A | Contracts page loads | | 1 pending contract visible | Pass |
| 2. Review contract terms | Contract: $800 Website | Terms displayed | | All details clear, payment terms visible | Pass |
| 3. Click Accept Contract | N/A | Contract becomes active | | Status changed to "Active", start date set | Pass |
| 4. Go to Active Projects | N/A | Project list shows | | Project appears in active list | Pass |
| 5. Click Submit Milestone | N/A | Milestone form opens | | Upload form with description field | Pass |
| 6. Upload files | homepage.html | File uploads | | Error: File upload failed - server timeout | Fail |
| 7. Add description | "Homepage complete" | Text saved | | Description added | Pass |
| 8. Submit milestone | N/A | Milestone submitted | | Cannot submit - missing file attachment | Fail |
| **Project Name: Workify** | | | **Test Designed by: Kazi Ayesha** | | |
| **Test Case ID: TC\_4 [passed]** | | | Test Designed date: 23/09/2025 | | |
| **Test Priority : High** | | | Test Executed by: Tester | | |
| **Module Name: Contrast and Milestone** | | | Test Execution date: 26/09/2025 | | |
| **Test Title: Accept Contract and Submit Milestone** | | |  | | |
| **Description: Test contract acceptance and work delivery** | | |  | | |
| **Precondition: Contract offer received** | | | | | |
| **Dependencies: Payment system active** | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status** |
| 1. Go to Contracts | N/A | Contracts page loads | | 1 pending contract visible | Pass |
| 2. Review contract terms | Contract: $800 Website | Terms displayed | | All details clear, payment terms visible | Pass |
| 3. Click Accept Contract | N/A | Contract becomes active | | Status changed to "Active", start date set | Pass |
| 4. Go to Active Projects | N/A | Project list shows | | Project appears in active list | Pass |
| 5. Click Submit Milestone | N/A | Milestone form opens | | Upload form with description field | Pass |
| 6. Upload files | homepage.html | File uploads | | File uploaded successfully (2MB) | Pass |
| 7. Add description | "Homepage complete" | Text saved | | Description added | Pass |
| 8. Submit milestone | N/A | Milestone submitted | | Milestone sent for client review | Pass |

**As a Admin:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by: Safwan Yasfee Fahim** | | |
| **Test Case ID:** TC\_Admin\_01 | | | Test Designed date: 02/7/2025 | | |
| **Test Priority (Low, Medium, High):** Medium | | | Test Executed by: Safwan Yasfee Fahim | | |
| **Module Name:** Admin Dashboard-Verify Accounts | | | Test Execution date: 22/08/2025 | | |
| **Test Title:** Verify accounts | | |  | | |
| **Description:** Ensure that Admin can approve pending accounts to maintain platform security | | |  | | |
| **Precondition:** Admin login to platform and pending accounts exist | | | | | |
| **Dependencies:** User registration must be complete | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Navigate verify accounts. 2. Select a pending account. 3. Click approve. | Pending account:user101 | Account status changes from pending to verified | | As expected | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by:** Safwan Yasfee Fahim | | |
| **Test Case ID:** TC\_Admin\_02 | | | Test Designed date: 02/7/2025 | | |
| **Test Priority (Low, Medium, High):** Medium | | | Test Executed by: Safwan Yasfee Fahim | | |
| **Module Name:** Admin Dashboard-Monitor job posts | | | Test Execution date: 22/08/2025 | | |
| **Test Title:** Monitor job post agreement**.** | | |  | | |
| **Description:** Ensure that Admin can review inappropriate job posts. | | |  | | |
| **Precondition:** Admin log in and job posts exist. | | | | | |
| **Dependencies:** User must create job post**.** | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1.Navigate to monitor job post.  2. Select a doubtful job post.  3. Click flag | Job Post ID:JP\_1000 | Job post is dropped for review | | As expected | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by: Safwan Yasfee Fahim** | | |
| **Test Case ID:** TC\_Admin\_03 | | | Test Designed date: 02/7/2025 | | |
| **Test Priority (Low, Medium, High):** Medium | | | Test Executed by: Safwan Yasfee Fahim | | |
| **Module Name:** Admin Dashboard-Resolve Disputes | | | Test Execution date: 22/08/2025 | | |
| **Test Title:** Resolve a dispute between client and freelancer | | |  | | |
| **Description:** Ensure that Admin can review disputes and make a fair resolution | | |  | | |
| **Precondition:** Dispute must exist between two users**.** | | | | | |
| **Dependencies:** Dispute raised from a project | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Navigate to Resolve Disputes 2. Select an open dispute. 3. Review client and freelancer evidence. 4. Select resolve in Client’s favor | Disputes ID: A\_109 | Disputes status changes to resolved. | | As expected | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by: Safwan Yasfee Fahim** | | |
| **Test Case ID:** TC\_Admin\_04 | | | Test Designed date: 02/7/2025 | | |
| **Test Priority (Low, Medium, High):** High | | | Test Executed by: Safwan Yasfee Fahim | | |
| **Module Name:** Admin Dashboard-Ban users | | | Test Execution date: 22/08/2025 | | |
| **Test Title:** Suspend users | | |  | | |
| **Description:** Ensure that Admin can suspend users violating platform rules | | |  | | |
| **Precondition:** Fraud account must exist | | | | | |
| **Dependencies:** User activity logs**.** | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Navigate to Ban users 2. Search for fraud account 3. Click suspend | User ID: Culpit\_420 | User cannot log in | | As expected | Fail |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Online Job Management System | | | **Test Designed by: Safwan Yasfee Fahim** | | |
| **Test Case ID:** TC\_Admin\_05 | | | Test Designed date: 02/7/2025 | | |
| **Test Priority (Low, Medium, High):** High | | | Test Executed by: Safwan Yasfee Fahim | | |
| **Module Name:** Admin Dashboard-Ban users | | | Test Execution date: 22/08/2025 | | |
| **Test Title: Suspend users** | | |  | | |
| **Description:** Ensure that Admin can suspend users violating platform rules | | |  | | |
| **Precondition:** Fraud account must exist | | | | | |
| **Dependencies:** User activity logs | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Navigate to Ban users 2. Search for fraud account 3. Click suspend | User ID: Culpit\_420 | User cannot log in | | As expected | Pass |

## Function-Based Metrics (Function Points Analysis)

***Function Point Calculation for Workify***

Based on the identified features and user stories, we analyze the system components:

External Inputs (EI)

* User registration forms (Client, Freelancer, Admin): 3
* Login authentication: 1
* Job posting form: 1
* Proposal submission form: 1
* Profile update forms: 2
* Chat message input: 1
* Milestone submission: 1
* Payment processing: 1
* Review submission: 2 Total EI: 13

External Outputs (EO)

* User dashboards (3 types): 3
* Job listings display: 1
* Proposal lists: 1
* Contract documents: 1
* Payment receipts: 1
* User profiles display: 1
* Chat message display: 1
* Reports (admin): 2 Total EO: 11

External Inquiries (EQ)

* Search jobs by filters: 1
* View user profiles: 1
* Check application status: 1
* Track project progress: 1
* View payment history: 1
* Admin user verification: 1 Total EQ: 6

Internal Logical Files (ILF)

* User accounts database: 1
* Jobs database: 1
* Proposals database: 1
* Contracts database: 1
* Messages database: 1
* Payments database: 1
* Reviews database: 1 Total ILF: 7

External Interface Files (EIF)

* Payment gateway integration: 1
* Email service integration: 1
* File storage service: 1 Total EIF: 3

Function Point Calculation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Component | Count | Simple | Average | Complex | Weighted Score |
| EI | 13 | 8×3=24 | 4×4=16 | 1×6=6 | 46 |
| EO | 11 | 7×4=28 | 3×5=15 | 1×7=7 | 50 |
| EQ | 6 | 4×3=12 | 2×4=8 | 0×6=0 | 20 |
| ILF | 7 | 3×7=21 | 3×10=30 | 1×15=15 | 66 |
| EIF | 3 | 2×5=10 | 1×7=7 | 0×10=0 | 17 |

**Object-Oriented and Class Metrics**

System Architecture Analysis

Weighted Methods per Class (WMC)

* User classes: Average 8 methods each
* Job management classes: Average 12 methods each
* Payment classes: Average 6 methods each
* Communication classes: Average 7 methods each Average WMC across system: 8.25

Depth of Inheritance Tree (DIT)

* Base User class → Client/Freelancer/Admin: DIT = 2
* Job class hierarchy: DIT = 2
* Payment processing: DIT = 3 Maximum DIT: 3, Average DIT: 2.3

Number of Children (NOC)

* User base class: NOC = 3 (Client, Freelancer, Admin)
* Job class: NOC = 2 (HourlyJob, FixedJob)
* Payment class: NOC = 2 (DirectPayment, EscrowPayment) Average NOC: 2.3

Coupling Between Objects (CBO)

* User management: CBO = 5
* Job management: CBO = 7
* Payment system: CBO = 6
* Messaging system: CBO = 4 Average CBO: 5.5

Lack of Cohesion of Methods (LCOM)

Based on method interactions within classes:

* High cohesion classes: 15 classes (LCOM = 0-2)
* Medium cohesion classes: 8 classes (LCOM = 3-5)
* Low cohesion classes: 2 classes (LCOM = 6+) Average LCOM: 2.8

**Operation-Oriented Metrics**

Method/Operation Analysis

Average Operation Size

* Lines of code per method analysis:
  + Simple operations (getters/setters): 3-5 LOC
  + Business logic operations: 15-25 LOC
  + Complex operations (search, matching): 30-50 LOC Average operation size: 18.5 LOC

Operation Complexity

Using Cyclomatic Complexity (CC):

* Simple operations: CC = 1-3 (60% of operations)
* Medium complexity: CC = 4-7 (30% of operations)
* High complexity: CC = 8+ (10% of operations) Average Cyclomatic Complexity: 3.8

Parameters per Operation

* No parameters: 25% of operations
* 1-2 parameters: 45% of operations
* 3-4 parameters: 25% of operations
* 5+ parameters: 5% of operations Average parameters per operation: 2.1

**Maintenance Metrics**

**Software Maturity Index (SMI)**

Based on development phases and changes:

Version Analysis

* Version 1.0 (Initial Release):
  + Total modules (Mt): 25
  + Modules changed (Fc): 0
  + Modules added (Fa): 25
  + Modules deleted (Fd): 0
* Version 1.1 (First Update):
  + Total modules (Mt): 27
  + Modules changed (Fc): 3
  + Modules added (Fa): 2
  + Modules deleted (Fd): 0
* Version 1.2 (Bug Fixes):
  + Total modules (Mt): 28
  + Modules changed (Fc): 5
  + Modules added (Fa): 1
  + Modules deleted (Fd): 0

**SMI Calculations**

SMI = [Mt - (Fa + Fc + Fd)] / Mt

* SMI v1.0 = [25 - (25 + 0 + 0)] / 25 = 0.0 (Expected for initial release)
* SMI v1.1 = [27 - (2 + 3 + 0)] / 27 = 0.815
* SMI v1.2 = [28 - (1 + 5 + 0)] / 28 = 0.786

Current SMI: 0.786 (78.6% stability)

This indicates good software maturity with approximately 79% of the system remaining stable between versions.

**Metrics Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| Metric Category | Key Measure | Value | Assessment |
| Function Points | Adjusted FP | 233 | Medium-large system |
| Class Complexity | Average WMC | 8.25 | Acceptable complexity |
| Inheritance | Max DIT | 3 | Good design depth |
| Coupling | Average CBO | 5.5 | Moderate coupling |
| Cohesion | Average LCOM | 2.8 | Good cohesion |
| Operation Size | Average LOC | 18.5 | Well-sized methods |
| Complexity | Cyclomatic Complexity | 3.8 | Low-medium complexity |
| Maturity | SMI | 0.786 | Good stability |

**Conclusion and Future Work**

The Workify online job management system has been successfully designed and developed as a complete and reliable freelance marketplace that connects clients and freelancers with efficiency, transparency, and ease of use. The project successfully addresses major challenges faced by existing platforms, such as high fees, complicated registration processes, limited communication channels, and unclear payment systems. By focusing on a clean and intuitive interface, clear workflows, and a fair transaction process, Workify creates a seamless experience for both clients and freelancers. The system architecture has been carefully structured with scalability and maintainability in mind, ensuring that the platform can continue to evolve as user demand grows. The modular design makes it easy to introduce new features, resolve issues efficiently, and guarantee smooth operation. Overall, the project has established a strong foundation that not only solves current problems in the freelancing market but also provides a positive, trustworthy, and professional experience for its users.

Future development of Workify will build upon this foundation with a focus on innovation, user engagement, and technological advancement. One of the immediate priorities is the creation of dedicated mobile applications for both Android and iOS so that users can manage their projects conveniently from any location. The platform will also incorporate stronger security measures such as multi-step authentication and real-time fraud detection to protect users and build confidence in online transactions. Artificial intelligence will be introduced to improve job and freelancer matching, helping clients find the most suitable talent more quickly while ensuring freelancers are recommended projects that align with their skills and expertise.

In the longer term, Workify aims to become a truly global platform by expanding its reach to international markets and supporting multiple currencies and languages. Additional features such as interactive analytics dashboards, detailed reporting tools, and advanced collaboration systems will help clients and freelancers track progress and communicate effectively. Plans also include the integration of educational resources to support skill development and career growth, encouraging more people to join the freelancing economy. As technology continues to evolve, the platform envisions exploring blockchain solutions to enhance transparency and trust in payments and contracts. With these future developments, Workify is positioned to grow into a leading platform in the gig economy, offering a secure, efficient, and empowering environment for all users.

Notion Link 🡪 <https://www.notion.so/Workify27adce9bdb5c80c484edd8f84a31624e?source=copy_link>