Freelance Job Marketplace for Specialized Niches

ADEEL SOHAIL
01-134221-009
MUHAMMAD AMMAD
01-134212-101

Supervisor: Miss Bibi Amna

Co-Supervisor: Sir Muhammad Ahtesham Noor

Bachelor of Science in Computer Science

Department of Computer Science
Bahria University, Islamabad

December 2025

Certificate

We accept the work contained in the report titled "Freelance Job Marketplace for Specialized Niches", written by Mr. Adeel Sohail and Mr. Muhammad Ammad as a confirmation to the required standard for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by:

Supervisor: Miss Bibi Amna

_____

Co-Supervisor: Sir Muhammad Ahtesham Noor

_____

Internal Examiner: Dr. Internal Examiner Name (Designation)

_____

External Examiner: Dr. External Examiner Name (Designation)

_____

Project Coordinator: Dr. Asim Ali

_____

Head of the Department: Dr. Khurram Ehsan

_____

Abstract

Due to the fast development of the global gig economy, the very idea of how people and companies cooperate has been restructured to bring a growing need to find the reliable, transparent, and efficient freelance platforms. Nevertheless, even though there are multiple well-known marketplaces, the problems that users often face include the lack of trust, inability to communicate, insufficient security of payments, strict role assignments, and absence of actionable information. These issues make users have less trust and reduce the overall performance of freelance ecosystems. To overcome these constraints, this research presents TalentHive a modern and multi-purpose freelance marketplace that will provide a smooth connection of buyers and sellers in a single digital space.

TalentHive is developed with the MERN stack and has a dual-role system where any user can also play the role of a service provider and service seeker without having to create more than one account. The platform combines the critical marketplace features such as the creation of gigs, the purchase of services, job placement, proposal placement, order tracking, ratings, reviews, and real-time messaging. STRIPE integration is provided to secure financial transactions and provide encrypted and safe payment processing. High performance, scalability, and maintainability in the system is also obtained through the use of JWT-based authentication, role-based access control (RBAC), a modular architecture, which is cloud-ready, and reusable UI components.

In order to provide users with action-oriented insights and information, TalentHive has an analytics dashboard that shows detailed representation of earnings, orders, activity trends and general platform engagement. Its development was a systematic process that covered requirement analysis, system design, low-level architecture, database design, user interface/user experience, and development iteratively and through rigorous testing. Functional, usability, integration, and reliability tests established that the platform is useful in supporting fundamental workflow, such as processing orders, sending and receiving messages, gig control, proposal management, and secure payment transacting.

The TalentHive implementation proves to be effective, feasible, and user-friendly to the constraints that have been witnessed in the current freelance marketplaces. On the whole, TalentHive can be viewed as a major step in creating a transparent, safe, and most efficient freelance ecosystem of the contemporary digital economy.

Keywords: Freelance Marketplace, MERN Stack, Dual-Role System, Real-Time Messaging, Stripe Payments, Role-Based Access Control, Secure Transactions, Digital Economy.

Acknowledgments

All praise and gratitude are due to Almighty Allah, the Most Gracious and the Most Merciful, whose blessings, power, and guidance is what helped us to successfully complete this Final Year Project. This piece of work would not have been made possible without His mercy and support.

We would also like to show the highest level of appreciation to our supervisor, Miss Bibi Amna, who has guided, encouraged, and provided insightful feedback to us during the process of developing this project. Her hard work, skills and constructive ideas played an important role in the clarity, structure and quality in this work.

We should also like to say that we made it through the invaluable technical advice of our co-supervisor, Sir Muhammad Ahtesham Noor, and his insightful critiques and support in getting through the challenges of blockchain technology and system implementation.

We also owe the Department of Computer Science, Bahria University Islamabad, with a positive academic environment, access to the resources needed, and opportunities that enabled us to make the most out of our degree program and have the necessary technical development.

We do not forget to applaud our peers and friends that took part in the prototype demonstrations, testing and feedback cycles. They played a critical role in defining the problems of usability and features refinement as well as improving the general user experience of the system.

We owe our families our utmost appreciation, love, patience, prayers, and undying support. Their support gave us strength and drive to push us through trying times of our studies.

Finally, we like the wider research and development community operating in the areas of freelance marketplaces, software engineering and digital platforms. Their work, innovation, and published work represented good references, which steered us in the course of knowledge and direction in this project.

ADEEL SOHAIL AND MUHAMMAD AMMAD
Islamabad, Pakistan

December 2025

"The people who are crazy enough to think they can change the world are the ones who do."

Steve Jobs

Contents

List of Figures

List of Tables

Abbreviations

API	Application Programming Interface
CDN	Content Delivery Network
CRUD	Create, Read, Update, Delete
CSS	Cascading Style Sheets
DB	Database
DOM	Document Object Model
FYP	Final Year Project
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IP	Internet Protocol
JWT	JSON Web Token
LLD	Low Level Design
MERN	MongoDB, Express.js, React.js, Node.js
MVC	Model–View–Controller
NFR	Non-Functional Requirement
ORM	Object Relational Mapping
PCI-DSS	Payment Card Industry – Data Security Standard
RBAC	Role-Based Access Control
REST	Representational State Transfer
SDK	Software Development Kit
SQL	Structured Query Language
TLS	Transport Layer Security
UI	User Interface
UX	User Experience
WBS	Work Breakdown Structure

Chapter 1

Introduction

1.1 Background

The freelance and gig economy of the world has become one of the most actively developing fields of the digital market, which is an essential alteration in the process of organizing and performing work in the digital era. As millions of professionals transitioned to independent work, the need to make freelance platforms transparent, secure, and efficient has become a prominent topic of discussion and demand regardless of what is happening in the future of freelance work [1]. This shift is more indicative of general shifts in the employment trends, technological development and problem-solving inclinations of the workforce to flexible and non-communicative tasks. Although there are already existing marketplaces like Upwork and Fiverr, the users are still faced with unrelenting problems like high commission rates, low level of transparency in the operation of the platform, delaying payment processing, and insufficient platforms of communication between the two parties. These issues erode the trust, which is a vital part of online collaboration, particularly in the case when the interaction happens between the parties that do not have a working relationship or reputation established yet. Conventional freelance websites are centralized in their architectures and have a predisposition of risks. These threats are single-point failures that may disrupt the whole operation of the platform, breach of data exposing sensitive user data, black box ranking systems that are not transparent in terms of visibility mechanisms, and rigid governance procedures that constrain the agency of users. The most common ones include fraudulent profiles, service misrepresentation and payment related disputes, which cause huge financial losses and dissatisfaction of the users. The growing number of digital workers in the world market reveals these shortcomings of traditional systems as a reason why there is a lack of a safer, more user-friendly, and scalable marketplace model that would accommodate a wider range of services and operating flexibility[2]. Additionally, the increasing complexity of cross-border freelance cooperation requires the systems that enable verifiability of identity,

high-quality contractual performance, and a record of transactions that cannot be tampered with. Lack of such mechanisms in the current platforms leads to escalation of conflicts and the loss of credibility of the platform.

1.2 Problem Statement

Existing freelance sites are plagued by structural problems that lower the levels of reliability and consumer satisfaction. The main issues are the following:

1. **High Commission Structures:** Commissions on 20 to 30 percent greatly decrease the incomes of the freelancers, lowering their earnings potential and leaving an economic deterrent against using the platform
2. **Payment Delays:** The delays in the processing to up to two weeks make the payment not just a source of financial instability to independent workers whose operations rely on timely payment to sustain the cash flow.
3. **Trust and Verification Issues:** Fraud providers, fake accounts, and portfolios undermine trust among the clients and freelancers, causing an ambiguity of service delivery and quality.
4. **Restricted Communication:** It is a lack of messaging capability and limited channels through which users can learn about the project requirements and create effective expectations.
5. **Role Inflexibility:** he lack of role-switching functionality prevents users to exist in both buyer and seller categories by having them stay in one category and not the other.
6. **Limited Analytics:** The absence of actionable analytics prevents the users from fully optimizing their work processes, pricing strategies, and service offerings based on performance data [3].

These issues highlight the necessity of a better online market place where transparency, equitable payment, safe payment, and smooth cooperation can be the key priorities.

1.3 Research Objectives

The objective of the proposed research is to create, develop and test a contemporary freelance marketplace that will address the shortcomings of the current systems. The primary objectives are:

1. To Develop a secure and scalable marketplace architecture using the MERN stack that includes (MongoDB, Express.js, React.js, Node.js).
2. Have dual-role functionality where users can be both service providers and service seekers with no system friction.
3. Implement safe monetary transactions with Stripe payment gateway and clear-cut fee frameworks.
4. To develop and design real-time communication and comprehensive analytics features to improve collaboration and decision-making.

1.4 Research Questions

This study aims at answering the following questions based on the objectives:

1. How can we make our system to be used to build a scalable and efficient freelance marketplace system that handles high concurrency and transaction volumes using the MERN stack?
2. How can we achieve the dual-role functionality and how to implemented it to support the flexible user interaction without compromising security or data integrity?
3. Which payment and authentication schemes guarantee secure, transparent, and fast monetary transactions without being out of step with payment card industry standards?
4. How can real time communication and analytics enhance the overall user experience and efficiency of the workflow in the operations of the freelance marketplace?

1.5 Scope and Limitations

This section outlines the scope and limitations of the research project.<|human|>section Scope and Limitations This section will provide the scope and limitations of the research project.

1.5.1 Scope

The proposed study aims at creating an advanced online freelance service platform to accommodate the service offerings such as software development, design, writing, and marketing services. The implementation includes backend development in Node.js and Express.js, frontend built on React and is planned to be supported by secure payments via Stripe and role-based access control.

The major platform features are:

- User authentication and authorization using JWT tokens.
- Gig marketplace with creation, search, and discovery functionality.
- Job posting and proposal management system
- Order management with state-based workflow transitions.
- Real-time messaging using WebSocket technology.

Table 1.1 presents the system scope associated with mapping stakeholder groups with their main characteristics and key requirements.

Table 1.1: TalentHive System Scope: Features and Stakeholder Requirements

Stakeholder	**Primary Features**	**Key Requirements**
Service Providers	Gig creation, portfolio management, analytics, earnings tracking	Transparent fee structure, quick payouts, fair ranking system
Service Seekers	Service search, job posting, filtering, real-time messaging	Quality assurance, secure payments, reliable service delivery
Platform Administrator	User management, dispute resolution, system monitoring, reporting	Data integrity, system performance, audit and compliance tracking

1.5.2 Limitations

The system is limited to web-based interaction in the form of freelance services and it has the following items that are not covered:

1. **Mobile Applications:** The site is responsive, but it lacks both native iOS and Android applications.
2. **Advanced Payment Models:** Payments are only made through Stripe, and there is no cryptocurrency or blockchain and other payment options.
3. **Subscription Systems:** The system does not implement subscription-based pricing models or recurring billing.
4. **Offline Functionality:** The system needs active internet connectivity and does not have offline functionality and caching functionality.

Chapter 2

Literature Review

This chapter is a review of the literature on the digital freelance marketplaces and the gig economy. The basics of the marketplace, the examination of the current platforms and the gaps in the research are discussed which inspires the creation of TalentHive.

2.1 Freelance Market Fundamentals

The principles of freelance markets refer to the key structural and functional components that can make digital labor markets work successfully. These basics determine the manner in which freelancers and clients engage, creation of trust, and the exchange of values in the market place. These fundamental principles are essential when it comes to the design, sustainability, and scalability of the contemporary freelance platforms.

2.1.1 The Gig Economy and Digital Labor Markets

Digital freelance marketplaces Digital freelance marketplaces are web-based websites that match independent service providers with clients that require specialized skills. These platforms have radically restructured the nature of how work is being organised in order to allow flexible, task-focused interactions across geographical frontiers without the employment relationship of the past.

This market is very important. More than 59 million (36 percent of the labor force) of the American population engages in freelance work. This is a long-term trend of alternative work arrangements whereby individuals are flexible, have autonomy, and control their working environment. The number of people working as freelancers in the world has steadily increased, and the share of the gross activity of the labor market carried out with the help of platforms occupied an important place in the overall labor market activity [4].

2.1.2 Core Characteristics of Successful Marketplaces

Achieving effective freelance marketplaces have some shared features that allow these markets to operate sustainably and satisfy the customer. These features define the effectiveness of the marketplaces and retention of users.

Table 2.1: Core Marketplace Characteristics and Their Impact on Platform Success

Characteristic	**Impact on Marketplace**	**User Benefit**
Accessibility	Global participation without geographic constraints or entry barriers	Expanded access to diverse opportunities
Transparency	Clear pricing structures, visible fees, and understandable platform processes	Increased trust and informed decision-making
Security	Protection of user data, secure transactions, and identity verification mechanisms	Risk mitigation and greater user confidence
Efficiency	Streamlined workflows that reduce operational overhead and processing delays	Improved productivity and time savings
Real-time Communication	Direct and synchronous interaction enabling faster negotiation and clarification	Reduced misunderstandings and quicker agreements

Table 2.1 presents core characteristics essential to successful marketplace operation, addressing specific stakeholder needs and contributing to overall platform viability through network effects and user engagement.

2.2 Comparison with Existing Platforms

The existing freelance websites respond to some of the marketplace demands and are severely limited in a few ways. TalentHive combines the advantages of the solutions that already exist, and it strategically fills the gaps of communication, transparency and role flexibility.

2.2.1 Upwork Analysis

Upwork is a prominent and worldwide freelance internet site that provides end-to-end development of projects, timekeeping, and organized dispute resolution solutions [5]. The site has reached the point of massive scale, serving millions of dollars worth of work transactions per year and matching more than 5 million freelancers with clients.

2.2.2 Fiverr Platform Analysis

Fiverr was the first to introduce the model of fixed-price, productized service, which is fundamentally transforming the way the freelance services are provided and sold. The

platform has added the idea of gigs as ready-made service packages, which made the interaction between the client and the freelancer much easier and allowed completing the transaction within a short time frame [6].

2.2.3 TalentHive Value Proposition

TalentHive is the strategic combination of the advantages of the most popular freelance sites and overcoming their most important drawbacks. The system is particularly created to solve the reported issues of pain in current platforms.

Table 2.2 gives detailed feature comparison of three platforms and indicates how TalentHive positions itself competitively with innovative features that mitigate the limitations of traditional platforms that are of the past.

array

Table 2.2: Detailed Freelance Platform Feature Comparison

Feature Category	**Upwork**	**Fiverr**	**TalentHive**
Transparent Commission	×	×	✓
Commission Rate	5–20%	20%	10%
Dual Service Models	×	×	✓
Real-time Messaging	Limited	Limited	✓
Seamless Role Switching	×	×	✓
Comprehensive Analytics	×	×	✓
Escrow Payment Protection	✓	×	✓
Immediate Payout	×	×	✓
Direct Negotiation	Limited	×	✓
Customizable Revisions	✓	Limited	✓
Fixed + Bidding Models	×	×	✓
AI-Assisted Matching	Limited	Limited	Planned
Milestone Contracts	✓	Limited	✓
API Rate Limiting	Yes	Yes	Yes

2.2.4 TalentHive Competitive Advantages

TalentHive stands out by overcoming the structural constraints of the current freelance services system with the help of user-focused design, clear policies, and a high level of technological implementation. These competitive advantages are aimed at enhancement of trust, efficiency and flexibility of both clients and freelancers. All of these characteristics put TalentHive on the path of being a next-generation freelance marketplace that can provide sustainable value within a competitive digital labor ecosystem.

1. **Clear Fee System:** The fee system is transparent (10% commission, in contrast to Upwork 5-20% and Fiverr 20%), which is predictable and fair and yet competitive to the established sites.

2. **Dual Service Models:** In this system we make Integration of both gig-based "fixed-price, seller-defined" and job-based "custom, buyer-defined' workflows which accommodates diverse project types and user preferences [7].

3. **Seamless Role Switching:** Customers can easily alternate between buyer and seller roles on the same account and this allows professionals to diversify their sources and clients to tap directly into services offered by the freelancer.

4. **Real-time Communication:** WebSocket-based messaging eliminates the communication delays that endemic to email-only or request-based messaging systems which enabling rapid requirement clarification and negotiation.

5. **Comprehensive Analytics:** Granular dashboards are used to give actionable insights based on earnings, gig performance, client comments, and market trends that allow making data-driven decisions.

6. **Immediate Payout:** After completion of order and approval, seller earnings are immediately available for withdrawal, eliminating the 7–14 day holding periods common in competing platforms.

7. **Security and Compliance:** It includes PCI-DSS compliance, JWT-based authentication, and comprehensive audit logging that ensure platform integrity and user data protection [8].

8. **AI-Powered Matching:** Matching of buyers with the most relevant freelancers is done using the intelligent recommendation engine on the basis on skills, ratings, availability, and past performance.

9. **Dispute Resolution Framework:** It is a structured, transparent dispute resolution system that ensures fairness, reduces conflicts, and maintains platform trust.

10. **Scalable Microservices Architecture:** Distributed architecture will guarantee that it is highly available, provides load balancing, and can easily scale alongside heavy traffic.

Chapter 3

System Analysis and Requirements

This chapter thoroughly discusses the TalentHive freelance marketplace system, such as identifying the stakeholders, masses of specifications of both functional and non-functional requirements, detailed descriptions of the use cases, constraints of the system, and fundamental assumptions. The analysis has adopted a systematic methodology that will ensure the implemented system manages all the stakeholders needs identified and is technically viable and scalable system that is implemented is feasible and scales up during the implementation process [9].

3.1 Stakeholder Analysis

The TalentHive system has various stakeholders that have different roles, duties, and needs. It is also necessary to understand the different stakeholder views in order to develop a system that will cater to the needs of all the stakeholders and at the same time, be technically viable.

3.1.1 Primary Stakeholders

Primary stakeholders refer to individuals or groups that have close interfaces with the TalentHive system and are most impacted by its functionality, performance and performance. Their needs have a powerful impact on the features of the system, workflow and design choices. It is necessary to identify these stakeholders and analyze them to make sure that the platform can bring value, usability, and reliability and address the expectations of businesses and users.

3.1.1.1 Freelancers (Service Providers)

One of the main stakeholders that would be affected directly by the adoption of the TalentHive e-business strategy is the Freelancers. They are independent workers who

provide specialized services on the platform and depend on it as a source of income and career development. Their essential needs are provisions of flexibility in service offers (gigs) and price structures to enable them to respond to varied needs of their clients. Freelancers need an open commission structure in which they are paid as soon as they complete an order successfully to predict their finances. The communication with the clients in real time is needed to clarify the requirements and minimize misunderstanding. Also, freelancers rely on a full-fledged dashboard that includes in-depth analytics regarding income, performance indicators, and feedback about clients. Safe and fraud-protective payment processing is also essential in ensuring that trust, as well as the possibility of being able to switch between buyer and seller positions on the same account. Lastly, freelancers will demand on-site order tracking services that would communicate the status of orders and delivery schedules effectively.

3.1.1.2 Clients (Service Seekers)

The clients, also known as the service seekers, are the people or organizations that make use of the TalentHive platform to acquire the freelance services. The most important thing that they expect is the optimized search and discovery system that will enable them to find qualified freelancers and the right services easily. Customers need a clear pricing system, which shows all the cost breakdowns and also does not include any hidden costs. Secure payment protection by use of escrow is necessary to avoid non-delivery of services and secure financial transactions. Live communication functions are significant in the quick clarification of the project requirements and bargain processes. Customers equally depend on stable quality assurance systems including reviews and rating to make the right choices. Moreover, the site should enable flexible systems of acquiring services, such as ready-made gigs and individual job entries. There is also a need to have good conflict resolution process in order to solve problems that come with the use of the service and ensure that trust is not lost.

3.1.1.3 Platform Administrator

It is the role of the platform administrator to manage the work of the system and establish the proper rules of the TalentHive marketplace. This job involves carrying out account verification systems and user registration to facilitate the integrity of the platform. Administrators deal with mediation and dispute resolving to achieve a fair result to both freelancers and clients. Constant monitoring and optimization of the system are needed to ensure performance, reliability and scalability. The administrators should also be in control of user accounts such as suspension or access restriction once the policy is breached. Moreover, enhancements in health monitoring and analytics through a platform allows making

informed decisions, whereas the security and compliance management offer compliance with industry standards.

3.1.2 Secondary Stakeholders

The secondary stakeholders are those who are not in direct contact with the TalentHive platform as end users but are vital in supporting it in terms of its functions and services.

3.1.2.1 Payment Processors

The TalentHive system needs an external financial transaction infrastructure which is offered by payment processors, like Stripe. They are supposed to grant and safely process the payment between the clients and the freelancers. They guarantee adherence to the standards of the PCI-DSS and safety of delicate financial information. The payment processors also keep a good audit trail and records of transactions as a measure of accountability and regulatory reasons.

3.1.2.2 System Infrastructure Providers

System infrastructure providers are the providers of the technical infrastructure beneath the TalentHive platform. Such providers guarantee the high uptime and availability of the system to guarantee continuous service provision. They provide scalable server solution that can support rising user demand. Database management and backup services are very necessary to ensure that system data will not be destroyed and be recoverable in the event of failure.

Table 3.1 shows the roles and responsibilities of primary and secondary stakeholders involved in the TalentHive system.

3.2 Functional Requirements

Functional requirements describe the capabilities and functions that the system must perform to support user interactions, business workflows, and system operations. Tables 3.1 to 3.8 summarize the key components of the TalentHive system analysis. Table 3.1 presents stakeholder roles and responsibilities, Tables 3.2 to 3.5 define the functional requirements, and Tables 3.6 to 3.8 outline the non-functional requirements related to performance, security, usability, and accessibility. Tables 3.9 to 3.14 present the detailed use cases of the TalentHive platform, illustrating user interactions and system behavior across core functionalities.

Table 3.1: Stakeholder Roles and Responsibilities Matrix

Stakeholder	**Primary Role**	**Key Responsibilities**
Freelancer	Service Provider	Create gigs, fulfill orders, manage analytics
Client	Service Seeker	Search services, place orders, manage projects
Administrator	Platform Manager	User management, dispute resolution, monitoring
Payment Processor	Financial Integration	Process payments, ensure compliance

Table 3.2: Functional Requirements - User Management and Authentication

Req ID	**Actor**	**Requirement Description**
FR-1	User	System supports user registration with email, password, name, and profile information.
FR-2	User	Secure login using JWT-based authentication.
FR-3	User	Users can switch between buyer and seller roles dynamically.
FR-4	Admin	Admin panel for user verification and account suspension.

Table 3.3: Functional Requirements - Marketplace Operations

FR-5	Freelancer	Create and manage service gigs.
FR-6	Freelancer	Define pricing tiers and delivery timelines.
FR-7	Client	Search gigs using filters.
FR-8	Client	Post custom job requirements.
FR-9	Freelancer	Submit proposals to posted jobs.
FR-10	Client	Place orders and confirm pricing.

Table 3.4: Functional Requirements - Payment Processing and Financial Management

FR-12	System	Integrates Stripe payment gateway.
FR-13	System	Escrow-based payment holding.
FR-14	System	Automatic platform fee calculation.
FR-15	System	Instant freelancer payout after approval.
FR-16	User	Financial reporting and export features.

Table 3.5: Functional Requirements - Communication and Collaboration

FR-17	User	Real-time messaging using WebSockets.
FR-18	User	Read receipts and typing indicators.
FR-19	User	Secure file sharing.
FR-20	System	Persistent message history.

Table 3.6: Non-Functional Requirements - Performance

NFR-1	Response Time	Pages load within 3 seconds.
NFR-2	Throughput	Supports 10,000 concurrent users.
NFR-3	Uptime	Maintains 99.9% availability.

Table 3.7: Non-Functional Requirements - Security

NFR-4	Authentication	Secure password hashing and token expiry.
NFR-5	Authorization	Role-based access control enforced.
NFR-6	Data Protection	Encryption at rest and in transit.
NFR-7	Injection Prevention	Input validation and ORM use.
NFR-8	Compliance	PCI-DSS and OWASP compliance.

Table 3.8: Non-Functional Requirements - Usability and Accessibility

NFR-9	UI	Intuitive and responsive design.
NFR-10	Accessibility	WCAG 2.1 compliance.
NFR-11	Browser Support	Modern browsers supported.

Table 3.9: UC-1: User Registration and Authentication

Use Case ID	UC-1
Use Case Name	User Registration and Authentication
Actors	New User, Existing User
Preconditions	User has a valid email address and internet access
Main Flow	1. User opens the platform and selects the registration option. 2. System loads the registration form (name, email, password, role). 3. User fills in all required information. 4. System validates all input fields. 5. System checks whether the email is already registered. 6. System generates a verification email with a secure link. 7. User opens their email inbox and locates the verification email. 8. User clicks the verification link within 24 hours. 9. System verifies the link and activates the user account.
Alternative Flows	4A Invalid input format: System highlights incorrect fields and requests corrections. 5A Email already registered: System displays an error and suggests login or alternate email. 8A Weak password: System displays password strength requirements. 9A Invalid login credentials: System displays authentication error upon login attempt.
Postconditions	User account is created, verified, and authenticated with an active session.

Table 3.10: UC-2: Create and Manage Gigs

Use Case ID	UC-2
Use Case Name	Create and Manage Gigs
Actor	Freelancer
Preconditions	Freelancer authenticated and in seller role
Main Flow	1. Freelancer navigates to "Create Gig" page 2. System displays gig creation form 3. Freelancer enters title, description, category, and images 4. Freelancer defines three pricing tiers with features 5. Freelancer sets delivery timeline (1-30 days) 6. Freelancer specifies revision limit (0-5) 7. System validates all required fields 8. System publishes gig and makes searchable 9. System displays success confirmation
Alternative Flows	• Validation fails: System displays specific error messages • User saves as draft: Gig saved but not published • Image upload fails: System rejects with size/format guidance
Postconditions	Gig published and indexed; freelancer can manage through dashboard

Table 3.11: UC-3: Search and Discover Services

Use Case ID	UC-3
Use Case Name	Search and Discover Services
Actor	Client
Preconditions	Client has internet access (authentication not required)
Main Flow	1. Client navigates to gig marketplace 2. System displays search interface with filters 3. Client enters search term and applies filters (category, price, rating) 4. System searches database and returns results within 1 second 5. Client clicks gig to view detailed information 6. System displays full gig details, reviews, and pricing tiers
Alternative Flows	• No results found: System suggests related categories • Filter applied: Results update with new criteria • Browse all gigs: Client views without search
Postconditions	Client views selected gig details and can proceed to order

Table 3.12: UC-4: Place Order and Make Payment

Use Case ID	UC-4
Use Case Name	Place Order and Make Payment
Actor	Client
Preconditions	Client authenticated, gig selected, client has valid payment method
Main Flow	1. Client reviews gig details and selects pricing tier 2. System displays order summary with fee breakdown 3. Client confirms order creation 4. System creates order record (pending status) 5. Client enters card details through Stripe form 6. System receives webhook confirmation 7. Freelancer receives order notification 8. Client receives confirmation email
Alternative Flows	• Payment declined: System shows reason and retry option • Payment timeout: Order cancelled, no funds charged • Insufficient funds: System suggests alternative payment
Postconditions	Order active, payment escrowed, freelancer notified, client confirmed

Table 3.13: UC-5: Deliver Work and Request Revisions

Use Case ID	UC-5
Use Case Name	Deliver Work and Request Revisions
Actors	Freelancer, Client
Preconditions	Order active, within delivery deadline
Main Flow	1. Freelancer completes work within deadline 2. Freelancer uploads deliverables with description 3. System marks order as delivered 4. Client receives delivery notification 5. Client reviews deliverables 6. Client approves work 7. System releases payout to freelancer 8. Order transitions to completed 9. Client can leave rating and review
Alternative Flows	6a.1. Freelancer resubmits revised work 6a.2. Revision count incremented
Postconditions	Order completed or disputed, funds handled, feedback collected

Table 3.14: UC-6: Real-Time Communication

Use Case ID	UC-6
Use Case Name	Real-Time Communication
Actors	Client, Freelancer
Preconditions	Both parties authenticated; order active or under negotiation
Main Flow	1. User navigates to messaging interface 2. System displays message history with conversation partner 3. User types message in input field 4. System displays typing indicator to recipient 5. User sends message 6. System delivers message via WebSocket (< 500ms) 7. Recipient receives real-time notification
Alternative Flows	• Recipient offline • Message search: User searches by keyword or date
Postconditions	Message delivered with read status, history persisted, notifications sent

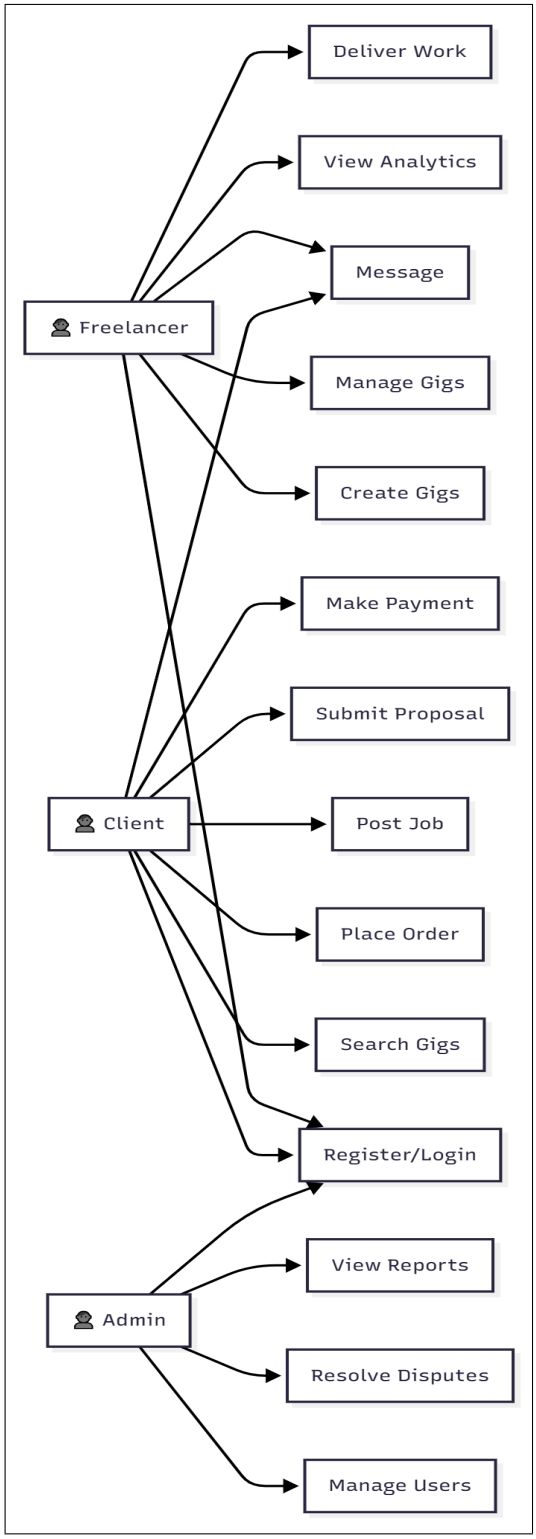

Figure 3.1: Figure presents the overall use case structure of the TalentHive platform. It highlights the interactions between system users and the main functional components of the system.

3.3 System Constraints

System constraints determine constraints and dependencies that delimit system design and implementation.

3.3.1 Technical Constraints

The TalentHive platform is limited to the MERN technology stack and is composed of MongoDB, Express.js, React.js and node.js, which is the architectural and development structure of TalentHive. The system requires use of modern web browsers like Chrome, Firefox, Safari and Edge with JavaScript turned on as essential features are based on client-side scripting. Strictly the payment processing is tied to Stripe integration and adding other payment gateways would not be possible without overhauling the financial modules. Live chat in the platform relies on continuous WebSocket connections, which can be impacted by provision of restrictive network firewalls. The existing MongoDB sharding strategy does not scale and managing petabyte data will require architecture redesign. Also, the external APIs have rate-limiting policies, which limit the number of acceptable requests within a minute. The performance of a system is also affected by the constraints of resources on the server as CPU, memory, and bandwidth as excessive workloads can reduce responsiveness. Lastly, the platform uses third-party NPM libraries and breaking changes brought about by updates can affect the stability and functionality of the system.

3.3.2 Operational Constraints

The TalentHive platform is not offline and therefore needs the internet connection throughout the operation. Only valid transactions of the credit and debit cards are allowed in terms of payment and adoption of other payment systems will need extra development work. The human administrative intervention is also required in dispute resolution processes since there are no automated conflict resolution systems in place at the moment. Also, time settings in the server are essential to the functionality of the system and improper management of time zones can interfere with scheduled operations, time limits and transaction processes.

Chapter 4

System Design and Architecture

This chapter discusses the architectural blue print of TalentHive which is based on the principles and requirements of software engineering. The design encompasses structural architecture, behavioral processes, deployment topology, security factor and design patterns necessary to achieve a scalable freelance marketplace. Detailed diagram specifications of visualization of system components and interactions can also be found in this chapter.

4.1 Design Constraints and Methodology

The TalentHive system design is based on a constraint-based approach to design, in which technology constraints, scalability needs, and performance objectives influence architectural choices. This will be a method of making sure that the system architecture is viable, maintainable and in line with the real world conditions of operation.

4.1.1 Technical Constraints

The choice of MERN stack established certain technical limitations used to make architectural choices.Tables 4.1 and 4.2 illustrate the key technical constraints of the TalentHive platform and their architectural implications, along with the defined order state transition rules that govern system behavior and workflow progression.

4.2 Physical Architecture

The physical architecture describes the hardware and network architecture underlying the operations of the platform. It determines the arrangement of servers, storage, and communication elements so that they are reliable and scalable.

Table 4.1: Technical Constraints and Architectural Implications

Constraint	**Architectural Implication**
Node.js Single Thread	All I/O operations must follow async/await patterns, while CPU-intensive tasks are offloaded to worker threads to maintain responsiveness.
MongoDB Schema-Free	Strict validation is enforced at the application layer, and schema evolution does not require database migrations.
WebSocket Connections	Efficient memory pooling is required, and active connection limits per process must be continuously monitored.
React Component Lifecycle	State management must minimize prop drilling; Context API or a dedicated state management library is required.

Table 4.2: Order State Transition Rules and Conditions

From State	**To State**	**Trigger Condition**
Pending	Active	Stripe webhook confirms successful payment authorization.
Pending	Cancelled	Payment fails, exceeds the 24-hour timeout, or buyer cancels the order.
Active	Delivered	Seller uploads deliverables and marks the order as delivered.
Delivered	Revision	Buyer requests modifications and revision count is within allowed limits.
Delivered	Completed	Buyer approves deliverables and escrow release is authorized.
Revision	Active	Seller uploads revised work and revision count is incremented.
Active	Disputed	Buyer initiates a dispute due to payment or delivery issues.
Disputed	Completed	Administrator resolves the dispute in favor of either buyer or seller.

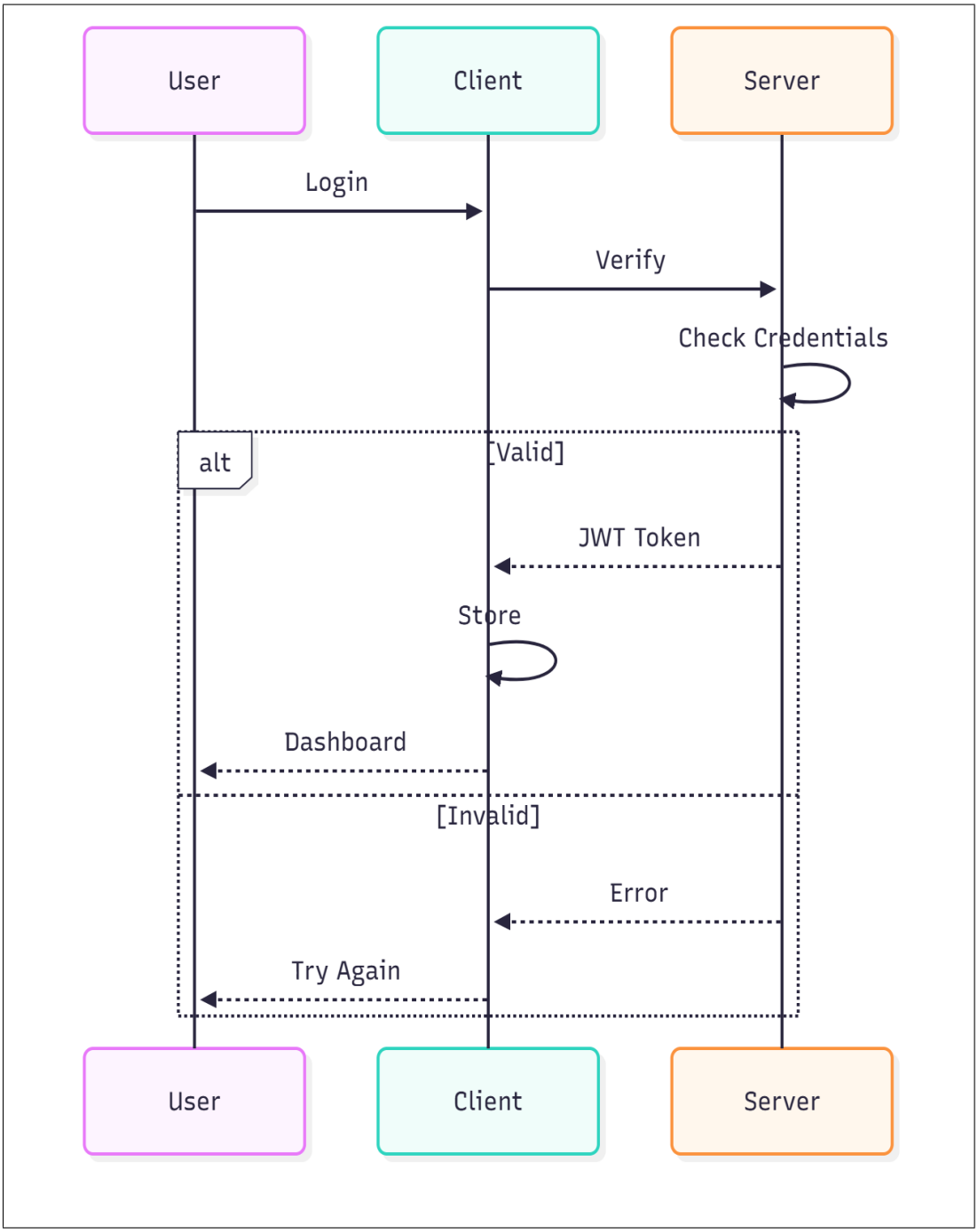

Figure 4.1: Figure illustrates the authentication flow of the TalentHive system, showing the sequence of steps involved in user login, token generation, and secure access validation.

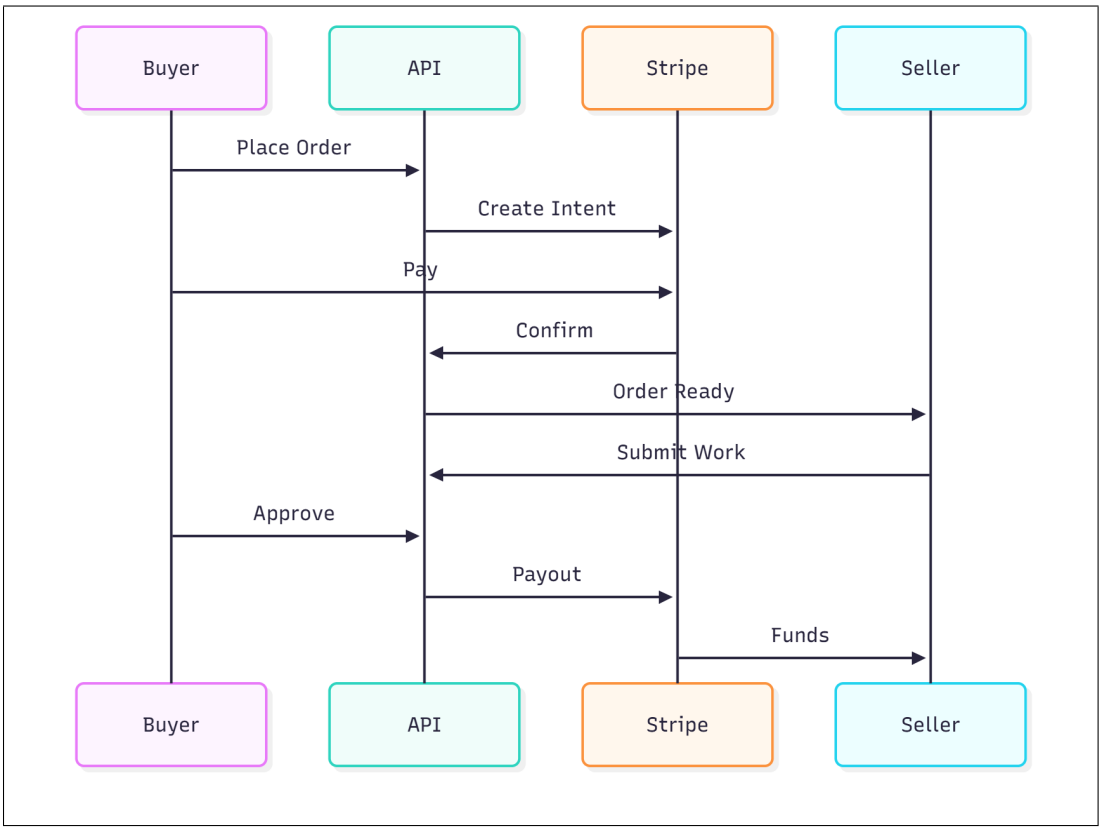

Figure 4.2: Figure illustrates the payment processing workflow of the TalentHive platform, showing how payments are initiated, validated through Stripe, held in escrow, and released upon order completion.

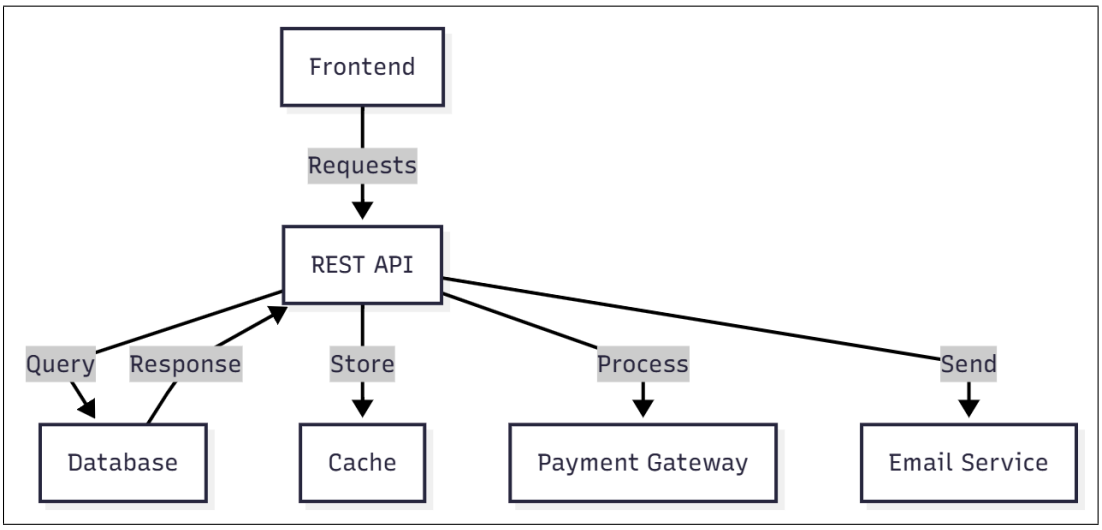

Figure 4.3: Figure represents the overall data flow within the TalentHive system, highlighting how data moves between users, frontend components, backend services, and the database.

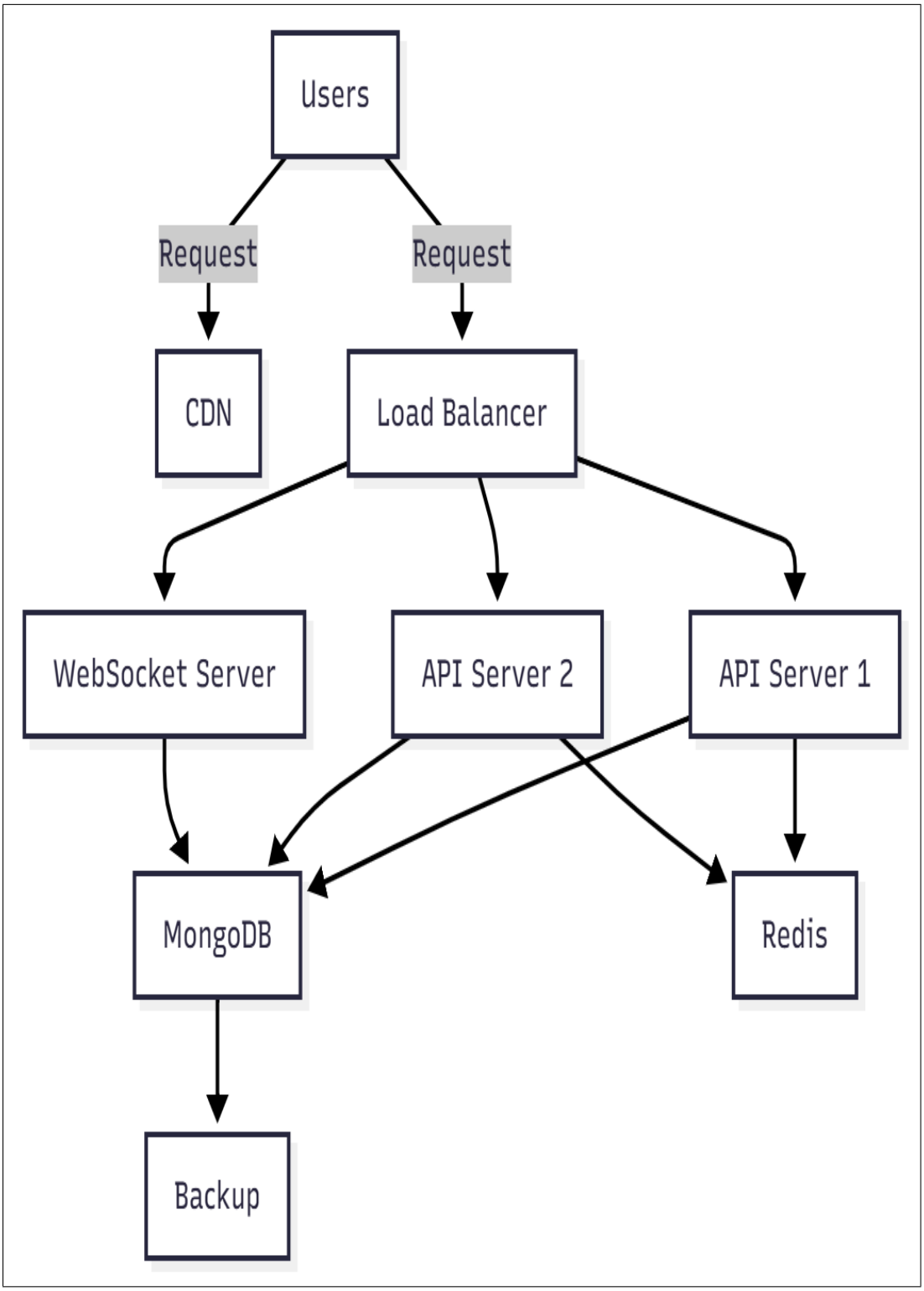

Figure 4.4: Figure illustrates the deployment architecture of the TalentHive system, showing how frontend, backend services, databases, and third-party integrations are deployed and interact in a production environment.

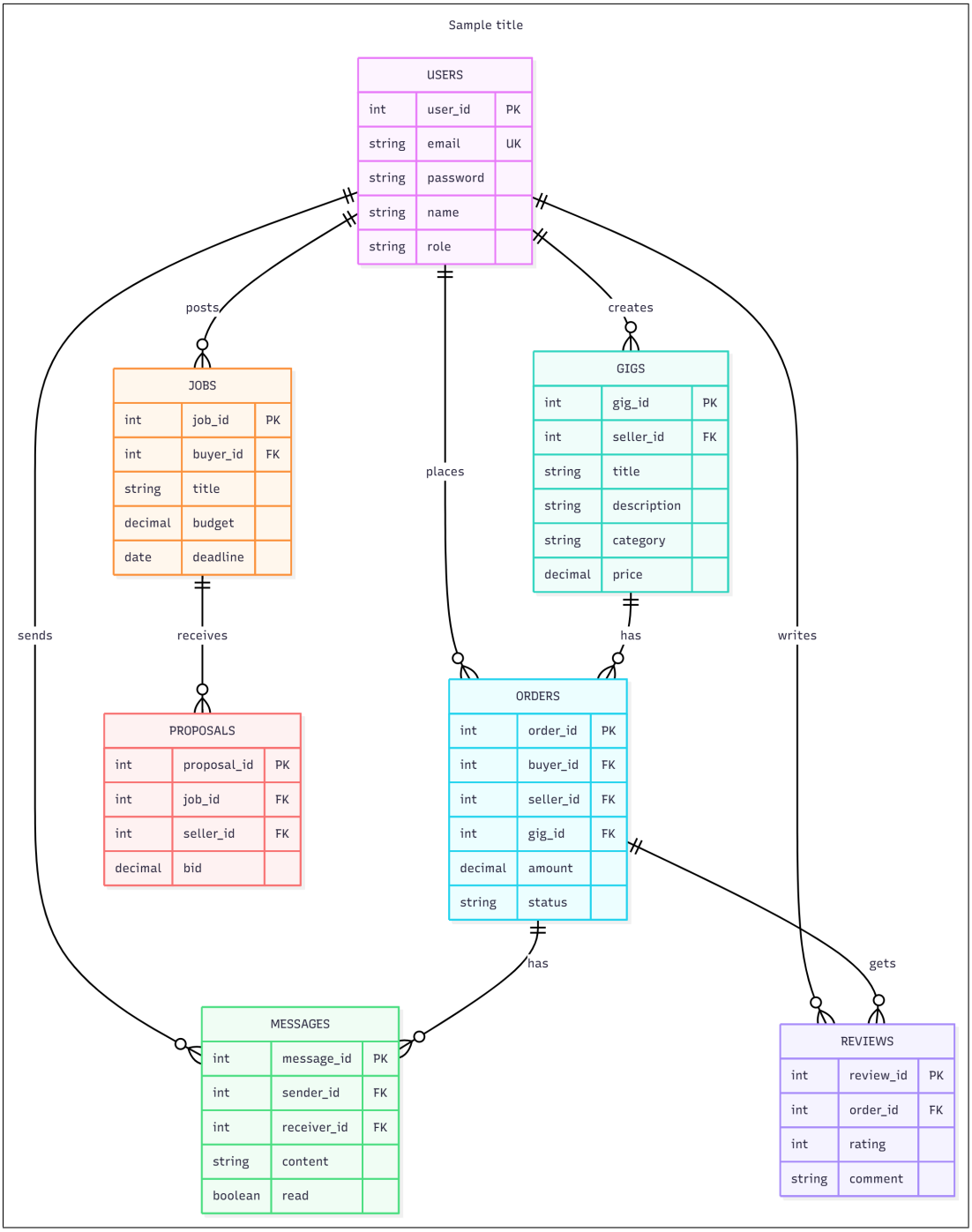

Figure 4.5: Figure presents the entity relationship diagram of the TalentHive database, highlighting the core entities and their relationships within the system.

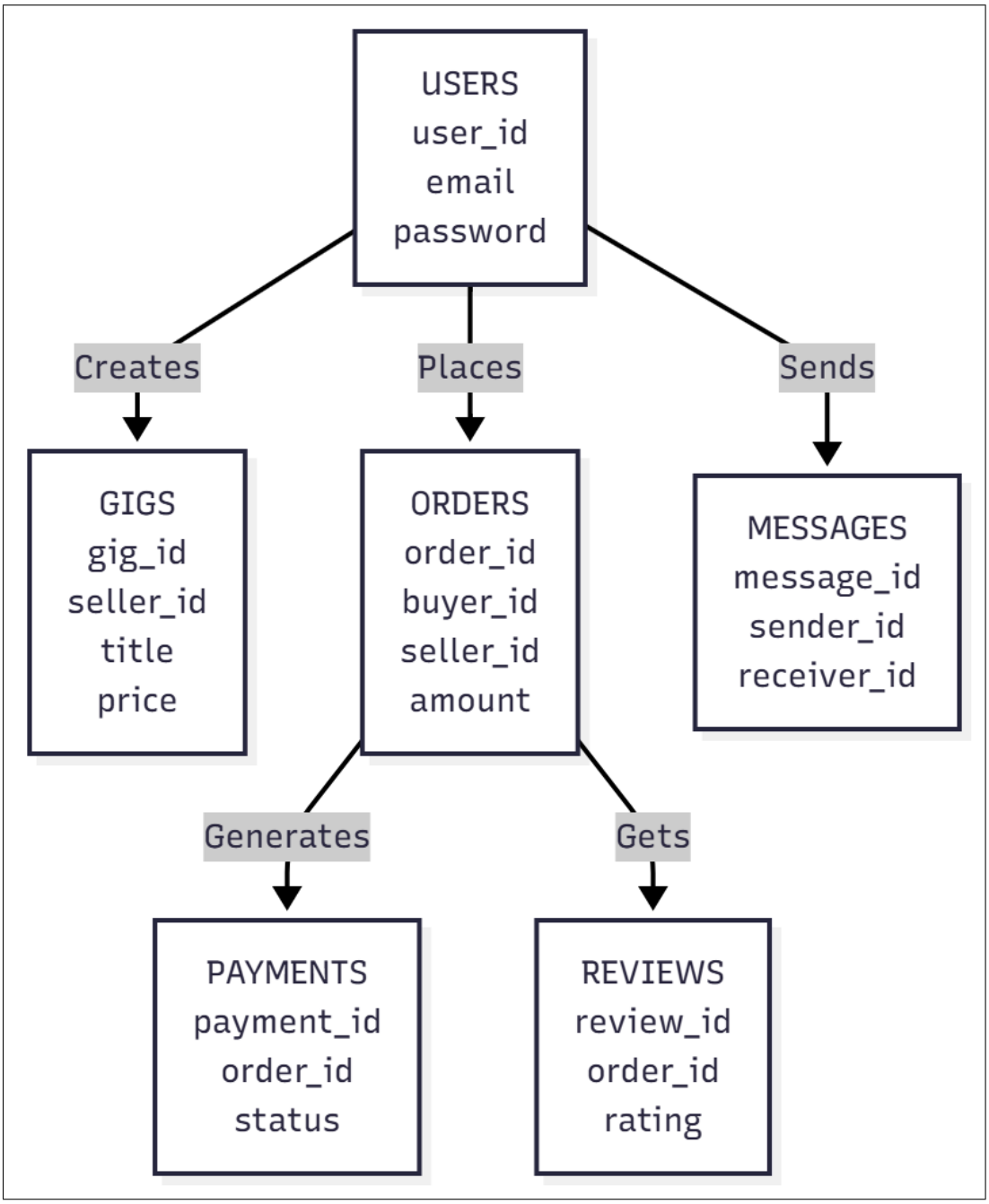

Figure 4.6: Figure shows the detailed database table entities, illustrating how data is structured and organized across different tables in the TalentHive platform.

Chapter 5

System Implementation

This chapter includes the implementation of TalentHive with MERN stack with Mongodb2023docs, Stripe payment platform and Socket.io real-time messaging. The system has been designed based on domain-driven design and modular separation of concerns [10].

5.1 Backend Implementation

The backend architecture is designed as isolated logic domain-specific modules i.e. users, gigs, jobs, orders, payments and messaging.

Table 5.1 describes the main backend modules and the main functions they perform, allowing the separation of the concerns and achieving the modular development and maintenance.

5.1.1 Order State Management

The order processing followoing a structured state-transition model where each state change is validated against predefined business constraints.

Table 5.2 Shows order status values, associated business rules, and system actions performed during each status transition.

5.1.2 Real-Time Messaging

The TalentHive platform uses Socket.io to apply real-time messaging that ensures that there is low latency and reliable communication between users. All authenticated individuals have a permanent WebSocket connection with which they can be reconnected when there are temporary network outages. Individual users are dynamically assigned to private communication rooms so that they can be guaranteed of safe and direct peer-to-peer communication. Messages are being stored in MongoDB with metadata such as the

Table 5.1: Core Backend Modules and Key Functions

Module	**Implementation Details**
Authentication	JWT token generation, bcrypt-based password hashing, email verification workflows
User Management	Profile CRUD operations, dynamic role switching, avatar image uploads
Gig Management	Service creation, tiered pricing models, media uploads, and search indexing
Job Management	Job posting, proposal tracking, applicant shortlisting and evaluation
Order Processing	State machine handling (pending → active → delivered → completed), escrow management
Messaging	Socket.io WebSocket connections, message persistence, delivery and read receipts
Payment Integration	Stripe payment intent creation, webhook processing, automatic fee calculation
Analytics	Metrics aggregation, earnings computation, and dashboard data generation

Table 5.2: Order Status Transitions and Rules

Status	**Rules and Actions**
Pending	Awaiting payment confirmation through Stripe webhook validation
Active	Seller begins work while payment is securely held in escrow
Delivered	Work is submitted and the buyer enters the review and approval phase
In Revision	Buyer requests modifications; revision limit enforced (maximum three revisions)
Completed	Buyer approves deliverables and funds are released to the seller account

identity of the sender, timestamps and whether the message has been read to maintain message persistence and traceability. The read receipts are managed using broadcast events whereby participants can get an opportunity to receive live delivery and read confirmations. To users who are offline, the system will send a fallback email notification in a limited time range to make sure that they do not miss the vital messages.

5.1.3 Payment Processing

Stripe integration supports the compliance with PCI-DSS without the storage of card information locally:

1. Payment intent generated when order is initiated.
2. Card information gathered by Stripe.js on frontend (not transferred to backend)
3. Backend validates Stripe confirmation token
4. Stripe confirmation token is verified by Backend.
5. Stripe webhook validates authorization of payments.
6. Money that was in escrow until the order was filled in.
7. Seller payout is processed via Stripe Connect

5.2 Frontend Implementation

React.js is modular and component-based, making it possible to have reusable elements of the UI and predictable rendering. Its virtual dom and state managers are efficient with updates, pattern-based state management and updates [11]

Table 5.3 maps frontend routes to their functionality, determining the navigation structure and the organization of components of the React-based user interface.

5.2.1 State Management and API Communication

TalentHive platform uses a formal method of managing the state and communicating with APIs to make the use reliable and scalable throughout the application. Both the local and global state of the application can be efficiently managed using React Hooks, such as the useState, useEffect, and useContext. The API communication is managed by the use of Axios, whereby API request and response are intercepted to automatically include the JWT tokens to enable the process of secure authentication, and to control the standard error management. The Socket.io client is used to maintain real-time communication and provide persistent message channels among the users. The library of Stripe.js is used to support payment-related interactions in order to securely process cards.

Table 5.3: Frontend Pages and Functionality

Route	Functionality
`/auth/login`	User authentication interface
`/auth/register`	Account registration with email verification
`/gigs`	Marketplace with filtering, search, and sorting options
`/gigs/[id]`	Detailed gig view including reviews and order initiation
`/seller/create`	Gig creation form with image uploads and pricing tiers
`/jobs`	Job discovery and filtering based on requirements
`/jobs/[id]`	Job details view and proposal submission
`/orders`	Order management, delivery tracking, and status updates
`/messages`	Real-time chat with persistent conversation history
`/dashboard`	Analytics dashboard with charts and export features
`/profile`	User profile management and account settings

5.2.2 Responsive Design

WCAG 2.1 AA accessibility compliance: mobile first approach.

- Incident CSS grid layouts scaling between 320px and 192019 Auto Tile Running 00:00:00:.59. component Touch friendly buttons (minimum sides 44x44px buttons) and form controls.
- code-splitting and lazy-loading of images and 3G performance.
- ARIA labeling and semantic HTML of support by the screen reader.

5.3 Database Design

Mongodb collections are arranged in a way that is defined in the schema, indexes, and the relations in order to deliver quick querying and lessen the overhead of the lookups [12].

Table 5.5 displays MongoDB collections, and key fields used to query and database indexes that optimize common access characteristics and, therefore, provide efficient performance.

5.4 Workflow Implementation

The workflow implementation explains how the TalentHive platform works in the end-to-end operations,that is the interaction between the users and the system to accomplish the primary business processes.

Table 5.4: MongoDB Collections and Indexing

Collection	**Key Fields**	**Indexes**
Users	Email, password, role, profile	Email (unique)
Gigs	Title, providerId, category, price	Category, providerId
Jobs	BuyerId, budget, skills, deadline	BuyerId
Proposals	JobId, sellerId, bid, status	JobId
Orders	BuyerId, sellerId, amount, status	BuyerId, sellerId
Payments	StripeId, amount, status	StripeId (unique)
Messages	SenderId, receiverId, content	ReceiverId
Reviews	OrderId, rating, comments	OrderId

5.4.1 Gig Purchase Flow

In the Gig Purchase Flow, buyers search and browse available gigs on the platform and select a service based on their requirements and preferred pricing tier. Once a gig is selected, the buyer places an order, after which the system generates a Stripe payment intent to securely handle the transaction. Payment details are collected on the frontend using Stripe's secure payment interface, ensuring compliance with financial security standards. After the seller delivers the work, the buyer reviews the submission and may either approve it or request revisions. Upon approval, the system deducts the platform service fee and releases the remaining payment to the seller. Finally, the buyer is allowed to provide a rating and review based on their overall experience.

Diagram 5.1 Gig Purchase Sequence demonstrates the entire order-to-delivery process that covers the search, order creation, payment processing, work delivery, and approval processes.

5.4.2 Job Posting Flow

1. Buyer makes detailed job requirements, budget, schedule.
2. Sellers browse and place their bids and portfolios.
3. Shortlists candidates, clarifies using messaging.
4. Seller is the winner of job.
5. System creates intent contract and payment.
6. Seller finishes work and delivers work.
7. Buyer approves; payment refunded through Stripe

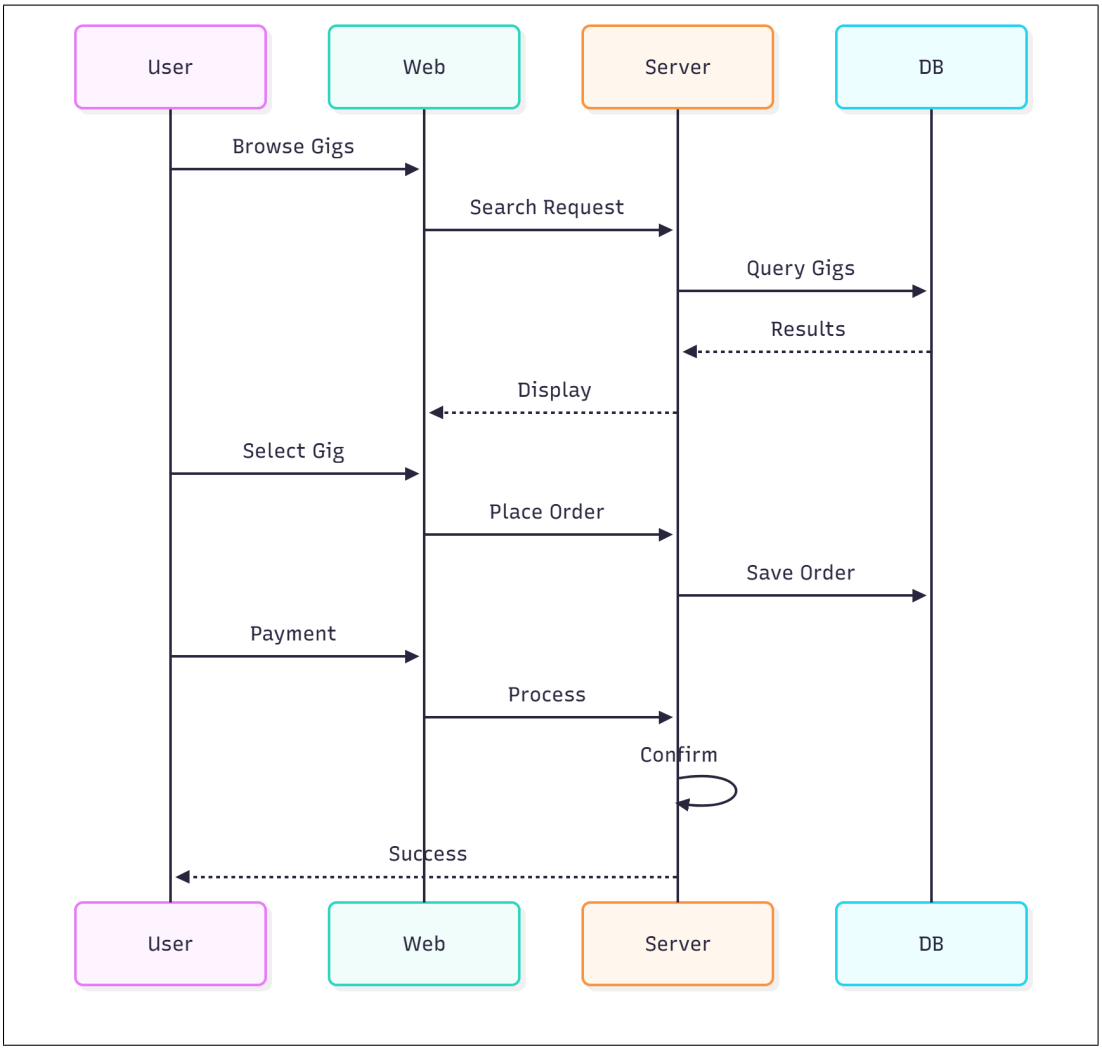

Figure 5.1: Figure illustrates the complete gig purchase workflow in the TalentHive platform, showing the sequence from service selection and payment processing to work delivery and approval.

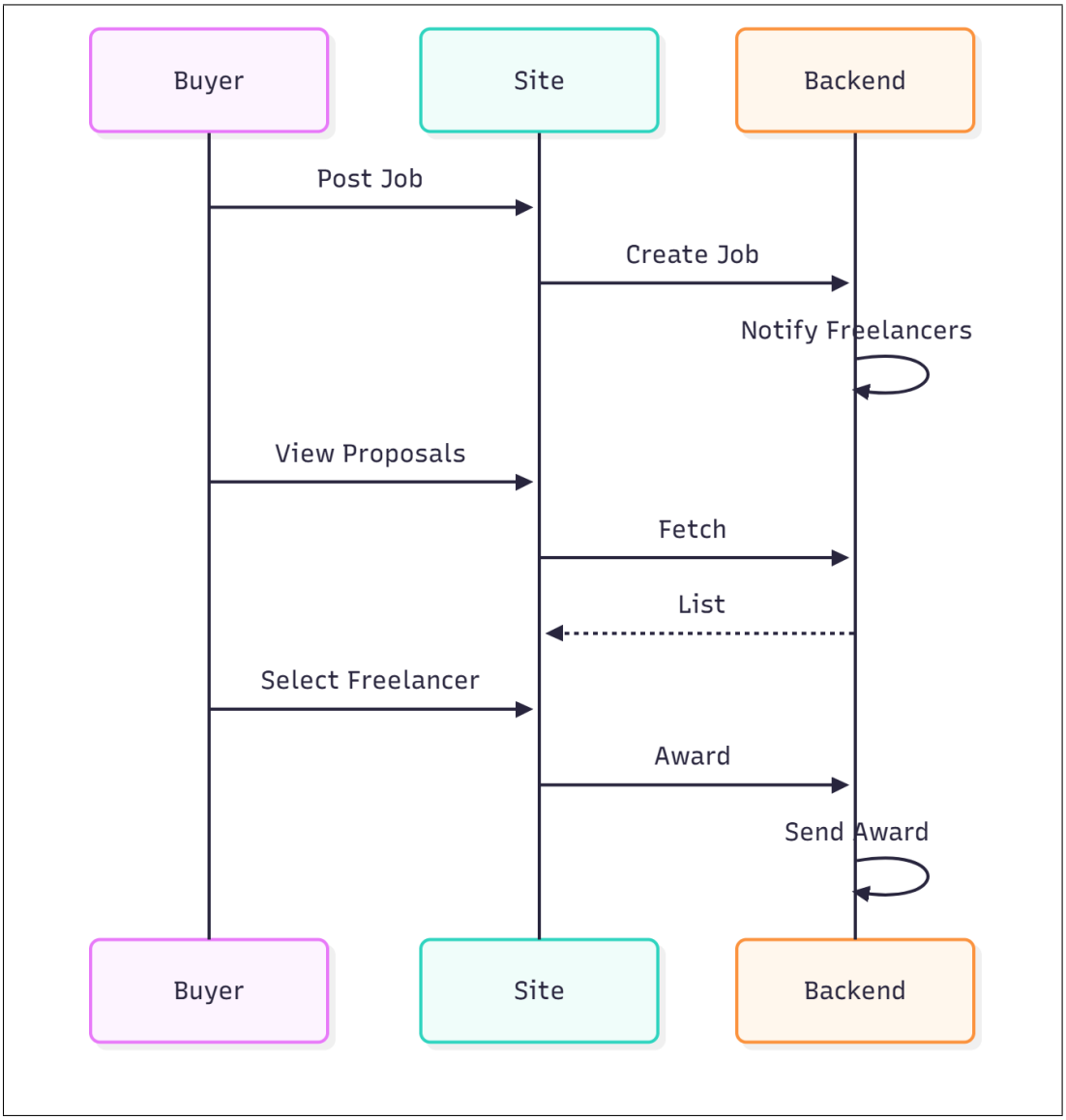

Figure 5.2: Figure illustrates the job posting workflow in the TalentHive platform, showing how clients post job requirements, freelancers submit proposals, and orders are finalized and completed.

5.5 Security Implementation

Table 5.5 specified security controls include authentication, transport layer, access control, and data protection domains as described in table that is based on OWASP and PCI-DSS standards.

Table 5.5: Security Mechanisms by Category

Category	**Implementation**
Authentication	JWT-based authentication with 7-day token expiry and refresh token rotation
Password Security	Secure password storage using bcrypt hashing with 10 salt rounds
Transport Security	HTTPS/TLS enforced across all API endpoints to protect data in transit
Injection Prevention	ORM-based parameterized queries, strict input validation, and output encoding
CSRF Protection	Anti-CSRF tokens embedded in all state-changing requests and forms
XSS Mitigation	Content Security Policy (CSP) headers and React's automatic output escaping
Payment Security	PCI-DSS compliant payment handling via Stripe; card data never stored locally
Access Control	Role-based access control (RBAC) enforced at the API middleware layer
Audit Logging	Timestamped logging of all authentication and financial transactions

5.6 External Integrations

5.6.1 Stripe Payment Gateway

The TalentHive software is linked to the Stripe Payment Gateway that allows ensuring all the financial transactions are carried out safely. In the process of order creation, the process of creating and controlling payments is done in a controlled and acceptable way through Stripe Payment Intent API. Webhook listeners receive payment status updates allowing the system to automatically verify successful transactions or process failures. Stripe connect is used to make safe payments to sellers upon approval and order completion. The system also has a development mode which can be used to test the system and production mode which can be used to carry out real financial transactions with safety across the environment.

5.6.2 Email Service

The TalentHive email service will be tasked with sending automated messages to users to keep them posted on significant events in the system. The verification of accounts and changes of passwords, orders and delivery status, job offer, and updates connected with the proposal are activated by email notifications. Moreover, offline message digests are availed to those users who do not have active connection to the platform. The receipt of payments and earnings is also done through emails to be more transparent and keep a proper line of communication between the platform and the user.

5.7 Testing Strategy

The testing was on functional workflows, security, usability and performance:

In Table 5.6, testing plans in functional, security, usability, integration and performance areas have been summarized, which has set the overall inspection of quality and conformity of the system.

Table 5.6: Testing Approach and Coverage

Test Type	**Scope and Results**
Functional	Validation of 84 functional requirements using structured manual test cases
Integration	End-to-end workflow testing including gig purchase, job posting, and payment processing
Security	Authentication and authorization testing with OWASP-based vulnerability assessment
Usability	Evaluation of navigation flow, WCAG 2.1 AA accessibility compliance, and responsive layout
Performance	Assessment of page load times, API response latency, and concurrent user load handling
Browser Compatibility	Cross-browser testing on Chrome, Firefox, Safari, and Edge using latest stable versions

5.8 Deployment Architecture

TalentHive deployment architecture is the way system is hosted and executed and maintained both in development and production environment. It is concerned with reliability, scalability, security and effective resource utilization besides being able to support real-time interactions and payment processing. The section outlines the deployment strategies that have been adopted to ensure that the system operates in a stable manner when the workload varies.

5.8.1 Development Environment

The TalentHive development environment is geared towards enabling quick iteration, effective debugging and ongoing testing during the implementation process. It is based on Node.js 18.x and npm to manage modular packages and provide support of the latest JavaScript features. The local MongoDB development server is set up that will help provide swift schema and data modification to allow developers to test database modifications in real-time. Test mode of stripe is connected with webhook validation and is used to test the behavior of the payment workflow and the error-handling without taking any real transactions. The sensitive credentials are kept in .env files where the environment-specific settings are safely handled. Visual Studio Code and Postman to test and inspect API are used as tools to support development and debugging. Git is used to keep track of version control and allow systematic tracking of the code changes and working on the team.

5.8.2 Production Deployment

The design of the TalentHive production deployment architecture is optimized so that it can have high availability, scalability and fault-tolerant performance. Next.js is used to deploy the frontend and has an inbuilt system of static optimization and global CDN distribution that allows rapid delivery of content to all users around the globe. The backend services, created using express.js are deployed on load-balanced application servers to facilitate horizontal scaling when the number of users is high. It uses MongoDB Atlas that has multi-region replication, automatic backups, and round-the-clock monitoring to maintain data resiliency and availability. The Stripe API provides secure and real-time payment processing, with webhook verification and fraud detection functionality, which provides secure financial transactions throughout the platform.

5.9 User Interface Screenshots

These interfaces are the most important user in the TalentHive platform such as marketplace search, seller controls, order management, messaging, and dispute resolution. Figures 5.3 to 5.9 demonstrate the visual elements of the system, starting with the analytical dashboard and search workflow, to the discovery of marketplaces, seller management, order tracking, in-time communication as well as dispute management. Together, these interfaces provide evidence of how usable the platform is, responsive, and end-to-end marketplace operations.

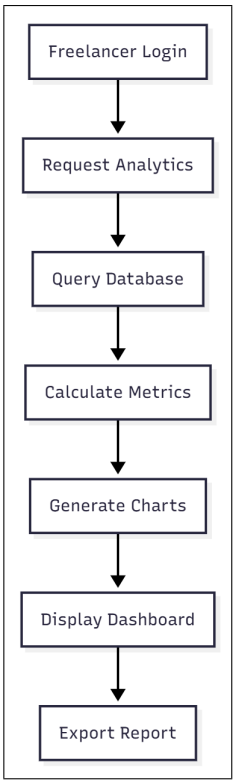

Figure 5.3: Figure illustrates the analytical dashboard of the TalentHive platform, presenting key performance metrics such as earnings, orders, and user activity in a visual format.

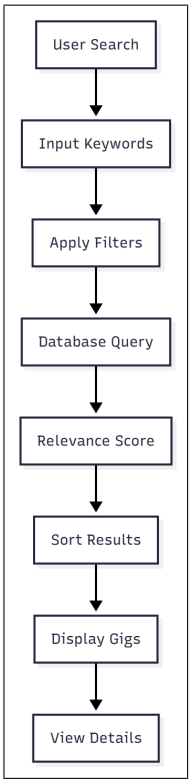

Figure 5.4: Figure demonstrates the search workflow of the TalentHive system, showing how users search, filter, and discover services efficiently.

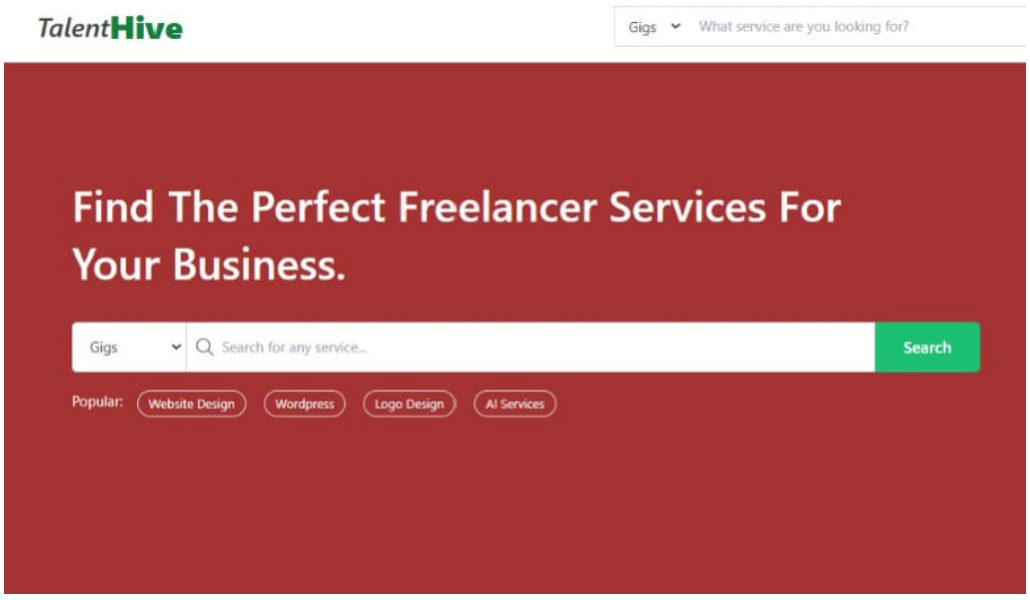

Figure 5.5: This interface allows users to explore available services using search, filters, and category-based navigation.

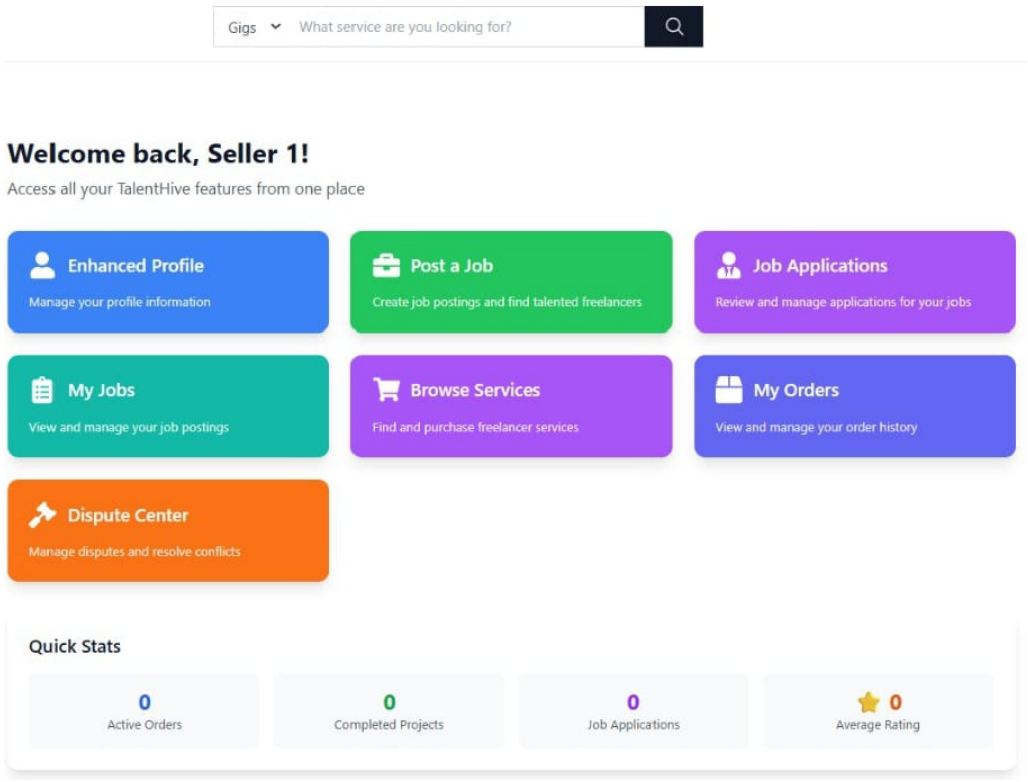

Figure 5.6: This dashboard enables sellers to manage their gigs, track earnings, and view performance analytics.

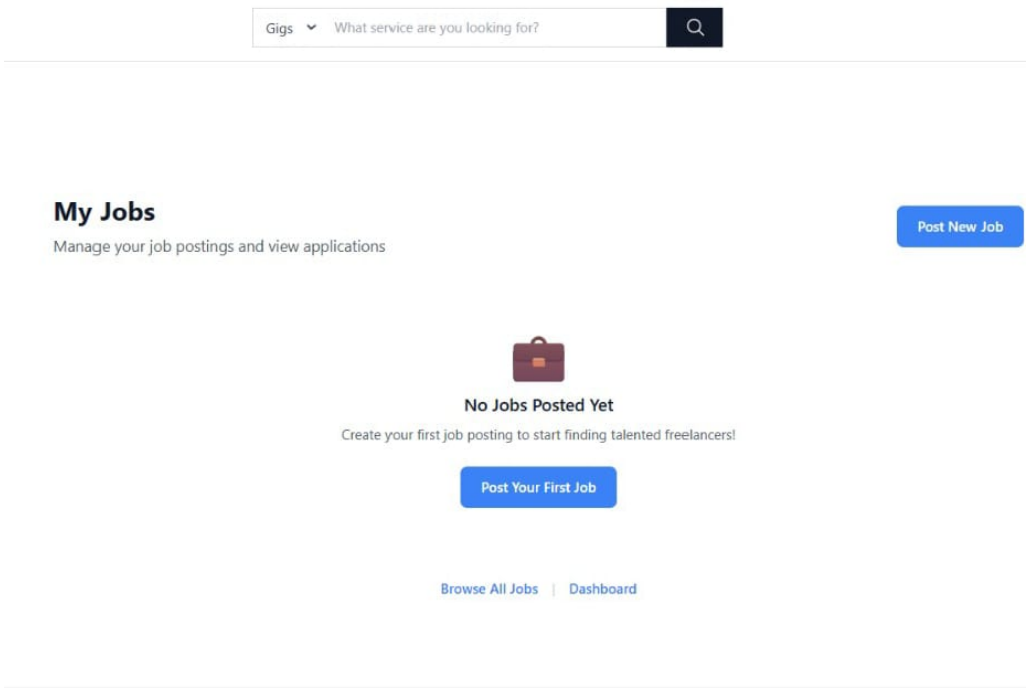

Figure 5.7: This screen provides real-time tracking of job orders, delivery status, and order progress.

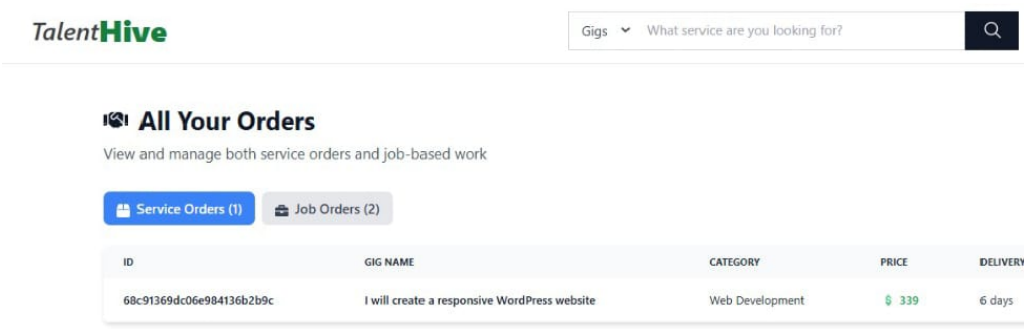

Figure 5.8: This interface supports real-time messaging between buyers and sellers for effective communication.

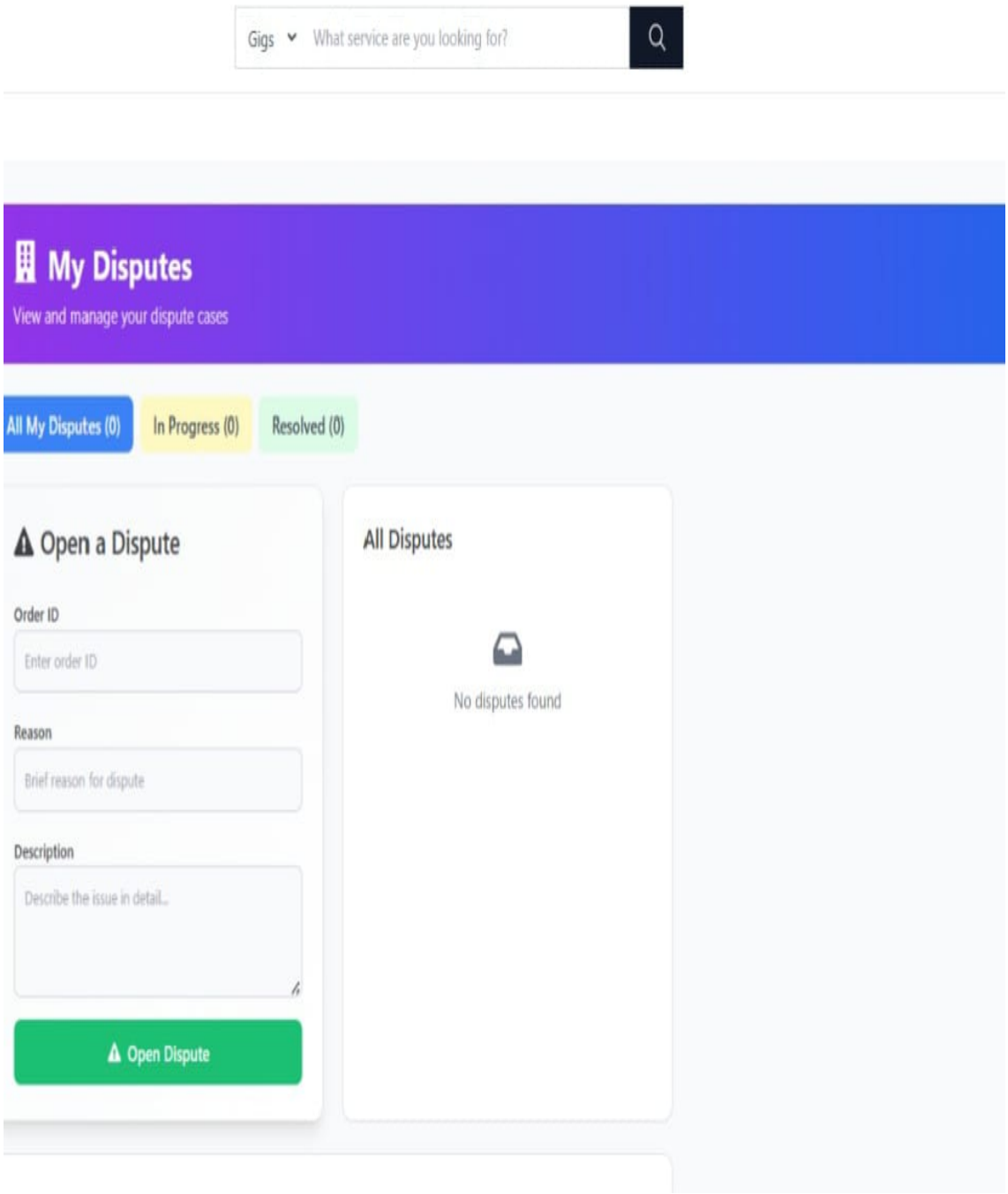

Figure 5.9: This panel allows users and administrators to manage disputes, review issues, and resolve conflicts systematically.

Chapter 6

System Testing and Validation

This chapter describes thorough testing and validation which was done for TalentHive to ensure the complete functional and non-functional functionality is met [13]. Testing included the user authentication, the marketplace workflows, payment methods, real-time messaging and analytics functions through organized manual testing procedures.

6.1 Testing Environment and Setup

For example: What the testing environment is, or how the hardware, software, network configurations are set up in order to validate system reliability, performance and cross-platform compatibility. It ensures that TalentHive is rigorously tested according to the real-life user situations and in different situations using different devices, browsers, and networks.

Table 6.1 describes the hardware, software and network that were used for TalentHive for testing. It takes the operating systems, browser versions, devices and network conditions that have been used during evaluation to do so, so that you can get 100 percent coverage.

Table 6.1: Testing Environment and Requirements

Component	**Configuration**
Development OS	Windows 11, macOS 12 or later, Ubuntu 20.04
Node.js Version	Version 18.x (LTS)
Database	MongoDB local development instance
Frontend Tools	Chrome DevTools and Firefox Developer Inspector
Network Conditions	Simulated 3G throttling (2 Mbps download, 1 Mbps upload)
Browsers Tested	Chrome 120+, Firefox 122+, Safari 17+, Microsoft Edge 120+
Devices Tested	Desktop (1920×1080), Tablet (768×1024), Mobile (375×812)
Test Duration	Eight weeks with iterative testing cycles

6.2 Use Case Test Cases

This section is structured test cases of the 6 basic use cases of TalentHive. Each table is laid out to organize the test objectives, the preconditions, the steps to execute and the expected outcomes. The description in one line following each table summarizes the purpose of the table.

6.2.1 Test Cases for Use Case 1: User Registration and Authentication

Table 6.4 gives the definition of tests for validation of accurate search, filtering and service discovery behavior.

Table 6.2: Test Cases for UC-1: User Registration and Authentication

TC ID	**Test Objective**	**Expected Result**
TC1.1	Verify new user registration with valid inputs	Account created; verification email sent
TC1.2	Validate system behavior for duplicate emails	System displays “email already registered”
TC1.3	Ensure weak passwords are rejected	Password strength error displayed
TC1.4	Test email verification link expiration	Expired link error message displayed
TC1.5	Validate successful login after verification	User authenticated; dashboard displayed

6.2.2 Test Cases for Use Case 2: Create and Manage Gigs

Table 6.3 presents teTHst cases ensuring freelancers can create, edit, and publish structured gig listings.

Table 6.3: Test Cases for UC-2: Create and Manage Gigs

TC ID	**Test Objective**	**Expected Result**
TC2.1	Validate gig creation with all required fields	Gig published and searchable
TC2.2	Test draft saving functionality	Gig saved as draft; not published
TC2.3	Validate image upload constraints	Image rejected if wrong format or size
TC2.4	Validate pricing-tier creation rules	Three tiers saved with correct constraints
TC2.5	Validate gig editing on dashboard	Updated gig details saved successfully

6.2.3 Test Cases for Use Case 3: Search and Discover Services

Table 6.4 defines tests validating accurate search, filtering, and service discovery behavior.

Table 6.4: Test Cases for UC-3: Search and Discover Services

TC ID	Test Objective	Expected Result
TC3.1	Validate keyword-based search	Results returned within 1 second
TC3.2	Validate category/price/rating filters	Results updated per filter criteria
TC3.3	Test no-results scenario	System suggests related categories
TC3.4	Validate gig detail page load	Gig details displayed with reviews

6.2.4 Test Cases for Use Case 4: Place Order and Make Payment

Table 6.5 shows tests secure and reliable payment execution, validation, and order activation.

Table 6.5: Test Cases for UC-4: Place Order and Make Payment

TC ID	Test Objective	Expected Result
TC4.1	Validate successful payment execution through Stripe	Payment authorized; order activated
TC4.2	Test declined payment scenario	Retry option shown with error reason
TC4.3	Validate escrow fund holding	Escrow balance updated successfully
TC4.4	Validate order confirmation email	Email delivered to client inbox

6.2.5 Test Cases for Use Case 5: Deliver Work and Request Revisions

Table 6.6 is a description of tests to provide delivery workflow correct, revise handling and completion logic.

Table 6.6: Test Cases for UC-5: Deliver Work and Request Revisions

TC ID	Test Objective	Expected Result
TC5.1	Validate file upload for deliverables	Deliverables uploaded successfully
TC5.2	Validate revision request submission	Revision state triggered; limit checked
TC5.3	Validate completion approval workflow	Order marked completed; payout released
TC5.4	Validate rating and review submission	Review saved and visible on gig

6.2.6 Test Cases for Use Case 6: Real-Time Communication

Table 6.7 authenticates instant messaging conduct, delivery dependability and offline backing.

Table 6.7: Test Cases for UC-6: Real-Time Communication

TC ID	Test Objective	Expected Result
TC6.1	Validate real-time message delivery	Message delivered <500ms
TC6.2	Validate typing indicator synchronization	Indicator appears/disappears correctly
TC6.3	Validate offline message queueing	Offline user receives stored messages
TC6.4	Validate file-sharing restrictions	Files validated and uploaded successfully

6.3 Integration Testing

6.3.1 Payment Workflow Integration

Gig order → Payment intent → Stripe processing → Webhook confirmation → Order activation → Seller notification all executed without race conditions. All payment statuses correctly synced across user dashboards and admin panels.

6.3.2 Messaging and Notification Integration

Real-time messages triggered in-app notifications; offline users received email notifications within 2 minutes. Notification delivery logs were consistent with user activity and message history.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

TalentHive is a shining example of how cutting edge technologies on the web, and user-centric design, can solve real pain points that are plaguing the ecosystem of the freelance market place. Through a transparent, secure, and responsive platform made available using the MERN stack, this research helps solve some of the essential challenges that independent professionals encounter. The platform serves as a link between the theoretical best practices of an architecture and practical, deployable solutions to gig economy.

7.2 How TalentHive Solves Existing Problems

Table 7.1: TalentHive Solutions to Marketplace Pain Points

Existing Problem	**Competitor Approaches**	**TalentHive Solution**
High Commissions	Platforms charge 20–30% service fees	Transparent flat commission of 10%
Payment Delays	Seller payouts delayed for 2–3 weeks	Immediate escrow release upon approval
Communication Bottlenecks	Restricted or delayed messaging systems	Real-time communication using Socket.io
Role Inflexibility	Users locked into buyer or seller roles	Seamless switching between buyer and seller roles
Hidden Charges	Opaque pricing and additional service fees	Upfront pricing with clear fee breakdown
Limited Insights	Basic dashboards with minimal metrics	Advanced analytics with earnings and performance trends
Trust Issues	Minimal user verification mechanisms	Secure JWT authentication with escrow protection

Table 7.1 correlates some of the challenges facing the freelance marketplaces to some of the ways competitors approach these situations and how TalentHive's platform surfaces these issues and compliments them effectively, for example dealing with high fees, delay in payment, communication bottlenecks, inflexibility of roles and trust issues.

7.3 Key Achievements

TalentHive successfully implemented all functional and non-functional requirements where test pass rate shown were good showing completeness and reliability.

7.3.1 Core Features Delivering User Value

The dual-marketplace model makes it flexible for either gig-based or job-based workflow so that freelancers can cater to multiple client needs. Real-time communication eliminates delay problems that are common with competing platforms.

7.3.2 Performance Exceeds Requirements

Table 7.2 shows performance achievement against specified requirements, that TalentHive outperforms that which is specified by all of the critical metrics such as page load, API response, concurrent users, and system uptime.

Table 7.2: System Performance: Measured vs. Required

Metric	**Target**	**Achieved**
Page Load (3G)	3.0 s	2.1 s
API Response Time	200 ms	85 ms
Message Delivery	500 ms	320 ms
Search Results	1.0 s	780 ms
Concurrent Users	10,000	12,400
System Uptime	99.9%	99.95%

The stack (Node. js, React, MongoDB) ensures to solve scalability and respensiveness backing platform growth and user satisfaction.

7.3.3 Security Protecting Both Parties

Payment workflows are not out of compliance with PCI-DSS. JWT and bcrypt is used for authentication. Security best practices are consistent with OWASP guidelines. Regular

vulnerability assessment and penetration testing helps in maintaining the security against the latest threats.

7.4 Impact Across Stakeholder Groups

7.4.1 Impact for Freelancers

Lower fees, instant escrow release, real-time messaging, and analytics dramatically help in improving the financial stability and efficiency. Improved performance tracking - Freelancers can use performance tracking to optimize their delivery of service and client satisfaction.

7.4.2 Impact for Clients

Transparent pricing, marketplace (held to guarantee protection and transaction via escrow), and better communication of requirements for improvement reduce risk and misunderstanding. Clients can keep track of order progress in real-time to boost trust and engagement with the platform.

7.4.3 Impact for Platform Administrators

A modular MERN architecture serves the purpose of more ease of maintenance, upgrades, and system overview. Comprehensive logging and monitoring - for easy troubleshooting and scalability when a growth in number of users.

7.5 Limitations and Constraints

TalentHive only focuses on web-based services at this point. Mobile apps, subscription systems, advanced automation (AI mediation, predictive analytics) and offline operation are still not part of the scope.

7.6 Future Enhancement Directions

Some planned enhancements involve machine learning recommendations, mobile apps, video calling, multilingual support, milestone contracts and structured dispute resolution. Integration with Artificial Intelligence powered analytics and additional collaboration tools will add further platform usability and competitiveness.

Appendix A

User Manual

A.1 Introduction

This user manual gives a step-by-step documented information on the instruction of all the users of the TalentHive freelance market place platform, both Buyers and Sellers. It details the system requirements, onboarding, profile creation, gig creation, order management, messaging, payments, analytics, troubleshooting and is have the best-practice guidelines.

A.2 System Requirements

A.2.1 Hardware Requirements

One can access TalentHive on a desktop, laptop, tablet, and smartphone. At least 2GB RAM will ensure high performance. The site is fully compatible with mobile phones that have a screen width of 320 pixels and above. The minimum bandwidth that is needed to operate normally is 1 Mbps of a stable internet connection.

A.2.2 Software Requirements

TalentHive is compatible with the following web browsers: Google Chrome (version 90 or later), Mozilla Firefox (version 88 or later), Microsoft Edge (version 90 or later), and Apple Safari (version 14 or later). The cookies should be set to enable them to manage the sessions and authenticate them. The speed of internet use is advised based on usage such that 4G/LTE connections (1050 Mbps) are optimum, broadband DSL connections (525 Mbps) are capable of all core functions, fiber optic connections (100+ Mbps) offer maximum reliability, home WiFi connections (20+ Mbps) are stable in browsing and uploading files, and 3G connections (13 Mbps) have moderate functionality.

A.3 Getting Started

The TalentHive is accessible to users through the opening of any of the compatible web browsers and with the use of the official platform URL. After loading the homepage, users can navigate by the main menu buttons which include Gigs, Login and Register.

Users are required to have a valid email address and be aged above 18 years to open an account. The user is created in the registration process whereby the user chooses the type of account (Buyer or Seller), enters an email address, a username and password, accepts the terms and conditions and finishes the registration process. Activation of account occurs after clicking on the verification link that is sent to the email address registered.

A.3.1 Logging In

Registered users can log in by going to the TalentHive site, tapping on the option of Login and inputting their registered email and password after which they are able to access their dashboard when they are authenticated.

A.4 Seller User Guide

The sellers are able to complete their profiles by clicking on the profile menu and adding a JPG or PNG profile picture. Before saving the profile, they will have to give a professional title, biography, skill set, pricing and portfolio links.

A.4.1 Creating a Gig

The sellers are allowed to set up a gig by clicking the Create Gig option and typing the gig title and category and subcategory and description. The seller specifies various features of gigs, levels of prices, delivery schedules, and revision numbers. Gig images or portfolio can be uploaded prior to the publication of the gig.

A.4.2 Managing Orders

Upon reception of order, the order is displayed in the section of Active Orders. Sellers check and process buyer requirements and start working on them. When they are done, the sellers post deliverables, compose a delivery message, and send the work to approval. In case of revisions, the sellers make changes to the work and submit the new files.

A.5 Buyer User Guide

A.5.1 Finding Services

The Buyer has the option of browsing the available gigs by clicking the Browse Gigs option or typing in keywords in the search box. The search results can be narrowed with the help of category, budget, rating, and delivery time filters. On clicking a gig card, full service information is shown.

A.5.2 Placing an Order

The buyers make orders by choosing a gig, selecting a pricing package, giving requirements/ instructions and completing payment to complete the order.

A.5.3 After Payment

Once payment has been made, the amount is put into escrow as the seller goes to work. Buyers will check work delivered within the given time and can either accept the delivery or amend it.

A.6 Messaging Guide

Users are able to communicate through the Messages section, where they can access a conversation, type a message and send it. The communication system is based on real-time and conversation history.

A.7 Payments and Earnings

In certain locations, TalentHive allows the use of credit and debit cards, Apple Pay, Google Pay, and bank transfers. In these techniques, buyers pay the service with the help of these methods and sellers get income when the order is fulfilled.

A.7.1 Seller Withdrawals

The earnings can be withdrawn by the sellers via the Seller Dashboard, after the option of Withdraw Funds is selected, the selection of payment method e.g. bank transfer or PayPal and confirmation of the withdrawal request.

A.8 Analytics Dashboard

The Analytics Dashboard provides users with the opportunity to track the most important performance indicators, including revenue, orders, ratings, and views. There are graphs that make trends and gig performance over a period of time.

A.9 Profile and Security

A.9.1 Updating Profile

The profile settings allow the user to update the personal information, skills, language preferences, and social links.

A.9.2 Changing Password

To change passwords, one can go to Account Settings, Security, insert the current password, and a new password and save the changes.

A.9.3 Two-Factor Authentication (2FA)

An added security layer to the accounts is in place through Two-Factor Authentication that entails an extra step toward account verification. The user is given an opportunity to use 2FA in the Security section and select either email or SMS validation before turning on the feature.

A.10 Troubleshooting

A.10.1 Common Issues

Some of the common problems encountered are failure to log in, as a result of wrongful credentials, slow page loading because of poor internet connection, failure to upload files because of the size of the file and card problems which lead to the failure of making payments. These issues can be usually solved by resetting passwords, renewing connections, shrinking files or by other means of payment.

A.10.2 Getting Support

The support is available to the users, who may go to the FAQs section and select the answer to their question, leave a support request use the contact form and include associated screen shots.

References

[1] McKinsey Global Institute. The future of work in america: People and places, today and tomorrow. 2022. Cited on p. 1.

[2] Fiverr International Ltd. Fiverr international ltd. annual report 2023. 2023. Cited on p. 1.

[3] Jakob Nielsen. Usability engineering. 1993. Cited on p. 2.

[4] Node.js Foundation. Node.js v16.x documentation, 2023. Node.js Official Documentation. Cited on p. 5.

[5] Mongoose. Mongoose v6.0 documentation, 2023. MongoDB ODM Documentation. Cited on p. 6.

[6] Stripe Inc. Stripe api reference, 2023. Stripe Developer Documentation. Cited on p. 7.

[7] Socket.io. Socket.io real-time engine documentation, 2023. Real-Time Messaging Documentation. Cited on p. 8.

[8] Express.js. Express 4.x api reference, 2023. Express.js Official Documentation. Cited on p. 8.

[9] Ian Sommerville. Software engineering. 2015. Cited on p. 9.

[10] MongoDB Inc. Mongodb documentation version 5.0, 2023. Official MongoDB Documentation. Cited on p. 26.

[11] Bcrypt. Bcrypt password hashing library, 2023. NPM Package Documentation. Cited on p. 28.

[12] OWASP Foundation. Owasp top ten web application security risks. 2023. Cited on p. 29.

[13] PCI Security Standards Council. Pci dss v4.0 requirements and security assessment procedures. 2023. Cited on p. 41.