CROWDSOURCING PLATFORM

Titled as: PROCONNECT

Submitted by:-

Anjani shaw

Subhranjali Panda

INTRODUCTION

- **ProConnect is a modern freelancing platform** built to connect skilled professionals with clients from around the world. We're here to redefine how freelancers and businesses collaborate—making it faster, simpler, and more secure.

- **Our platform offers a professional environment** where freelancers can showcase their expertise, find relevant projects, and build lasting client relationships, while businesses can easily discover and hire top-tier talent tailored to their needs.

- **ProConnect caters to a wide range of industries**, including technology, design, writing, marketing, business consulting, and more—ensuring diverse opportunities for freelancers and a broad talent pool for clients.

- **We use smart matching algorithms** that help clients connect with freelancers who best fit their project requirements, based on skills, experience, ratings, and availability.

- **Our mission is simple**: to bridge the gap between brilliant minds and forward-thinking businesses. With a user-friendly interface, secure payment systems, and a wide range of categories, ProConnect makes freelancing efficient, reliable, and rewarding.

PROBLEM STATEMENT

- **Freelancers struggle to find consistent and quality work**, especially when starting out, due to high competition and lack of visibility on existing platforms.
- **Clients face difficulty identifying reliable and skilled freelancers**, often leading to mismatched hires, delays, or subpar project outcomes.
- **Lack of trust and transparency** due to fake profiles, unverifiable portfolios, and unreliable reviews makes it hard to make informed hiring or project decisions.
- **Payment delays and disputes are common**, with limited protection for either party, leading to frustration and financial risk.
- **There is no all-in-one platform** that seamlessly combines smart talent matching, project management, secure payments, and a supportive community for growth.

ABSTRACT

ProConnect is a next-generation freelancing platform designed to bridge the gap between talented professionals and businesses seeking reliable, high-quality services. In an era where remote work is rapidly becoming the norm, ProConnect addresses the growing challenges faced by both freelancers and clients—ranging from poor job visibility, high service fees, and unreliable communication to difficulty in finding trusted, skilled talent.

By leveraging smart matching algorithms, integrated project management tools, secure milestone-based payments, and transparent rating systems, ProConnect offers a seamless, end-to-end solution for freelance collaboration. The platform supports a wide range of industries including tech, design, writing, marketing, consulting, and more—providing opportunities for freelancers to grow and enabling businesses to scale with confidence.

With a user-centric approach, ProConnect not only facilitates efficient hiring but also empowers freelancers to build their brand, develop long-term client relationships, and gain access to resources that support professional development. ProConnect aims to become a global hub for freelancing—where work is not just transactional, but meaningful, reliable, and future-ready.

Why We Need ProConnect?

In today's rapidly evolving digital economy, freelancing is no longer a side hustle—it's a full-fledged career path. Yet, many freelancers struggle to find consistent, high-quality work, while businesses are frustrated by inefficient platforms, poor talent matches, and high costs. Existing freelancing websites often fail to address key pain points such as trust, transparency, fair compensation, and long-term collaboration. **ProConnect** is designed to solve these problems by offering a streamlined, intelligent, and user-focused freelancing platform that benefits both freelancers and clients.

The purpose of ProConnect is to:

- **Bridge the gap** between skilled freelancers and businesses seeking high-quality, dependable remote talent.
- **Simplify the freelancing process** with a user-friendly platform that integrates job discovery, communication, task management, and payment.
- **Promote fairness and transparency** through verified profiles, secure milestone-based payments, and honest reviews.
- **Empower freelancers** to grow their careers through skill-based job matching, personal branding, and professional networking.
- **Help businesses save time and resources** by providing quick access to pre-vetted, well-matched talent across various industries.

Project Scope

The scope of the ProConnect project includes:

✅ **Platform Features:**

- **User Registration & Profile Creation** for both freelancers and clients
- **Smart Matching Algorithm** to recommend freelancers or jobs based on skills, experience, and ratings
- **Job Posting and Proposal System** with filters for categories, budget, timelines, etc.
- **Built-in Communication Tools** including messaging, notifications, and file sharing
- **Project Management Dashboard** for tracking progress, deadlines, and deliverables
- **Secure Payment System** using escrow, milestone tracking, and transaction history
- **Ratings and Review System** to ensure transparency and accountability
- **Admin Panel** for managing users, content, payments, disputes, and support

TECH STACK

FRONT-END

HTML,CSS,JS

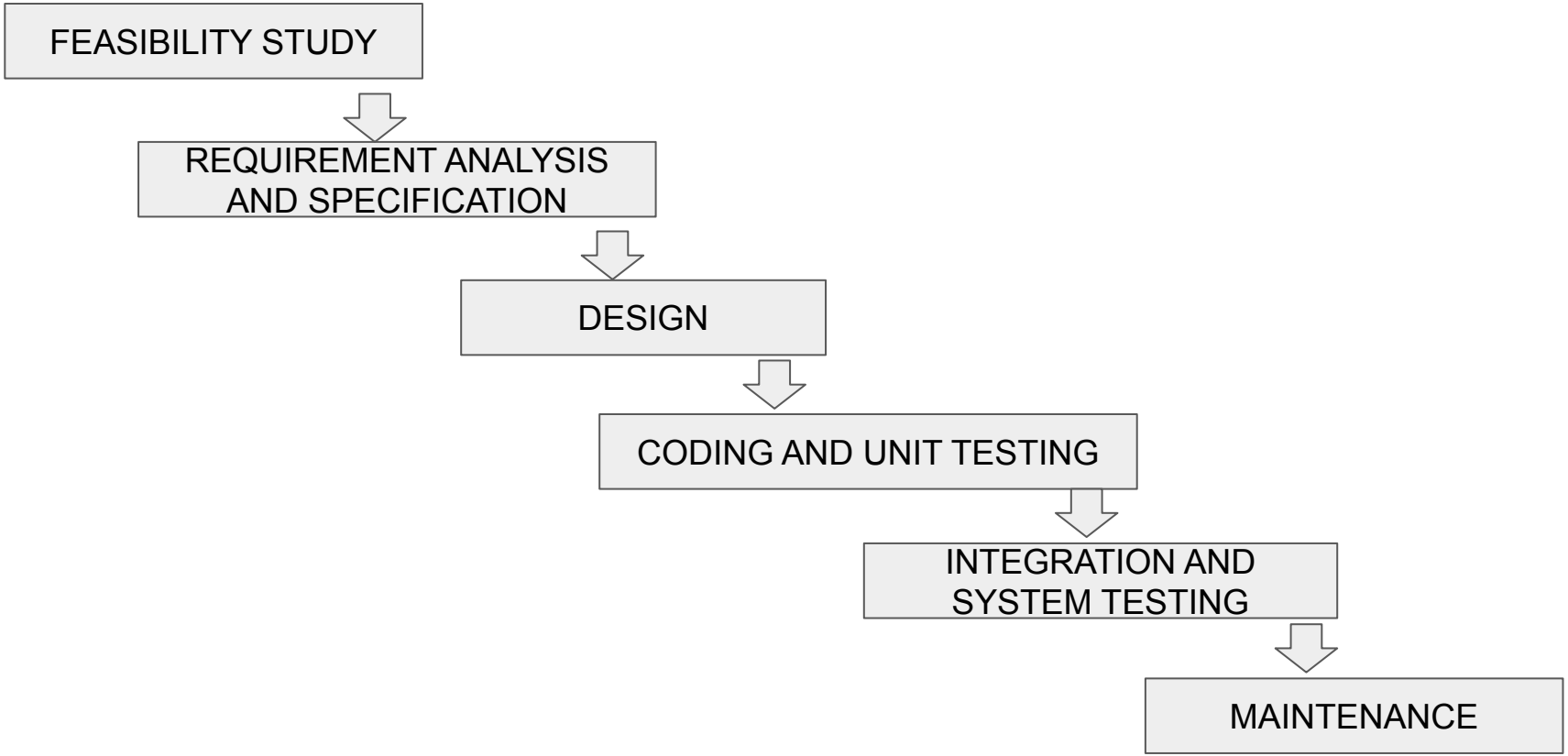
BACK-END

JAVA, SPRINGBOOT

DATABASE

MySql

SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)



FEASIBILITY STUDY

Economic Feasibility

- This project is economically feasible as it doesn’t require much. The only requirement is of a web browser along with any operating system.
- This is free for viewing on World Wide Web as a registered name.
- The people who use this can connect to the browser and this is the only thing that acts as a cost that is incurred in this application.

Technical Feasibility

- For the successful development of this application all we need is an internet connection, a database server and a supporting web server. This can be successfully implemented in Microsoft Edge environment and Chrome .

SOFTWARE REQUIREMENT SPECIFICATION

• The purpose of this document is to define the software requirements for **ProConnect**, a web-based freelancing platform that connects freelancers with businesses seeking remote talent. It outlines the system's functionality, performance, design constraints, and overall architecture.

• Operating Environment

➔ VS CODE
➔ JAVA
➔ SPRING BOOT
➔ MYSQL
➔ GIT
➔ GITHUB

FUNCTIONAL REQUIREMENT

User Registration and Login

- Users can register as Freelancer or Client
- Email verification is required
- Login via email/password or third-party (Google, LinkedIn)

Profile Management

- Freelancers: skills, experience, portfolio
- Clients: profile, job history, payment verification
- Editable dashboard with updates

Payment

- Easy and direct payment
- Admin-controlled dispute resolution system

Job Posting

- Clients can post job listings with detailed scope, budget, and timeline
- Freelancers can browse, filter, and apply to jobs
- Proposal management interface for both parties

Project Management

- Status tracking: Open, In Progress, Submitted, Completed

Admin Panel

- User management (suspend/verify)
- Job and content moderation
- Financial report generation

NON-FUNCTIONAL REQUIREMENTS

Performance

- The platform should handle up to 10,000 concurrent users with minimal latency.

Scalability

- Microservices-based architecture for easy scaling as the user base grows.

Usability

- Clean, modern, and responsive UI with accessibility standards..

Security

- Encrypted user data and transactions (SSL, HTTPS, 2FA)
- Role-based access control
- Regular security audits and vulnerability scanning

Maintainability

- Modular codebase with proper documentation
- Continuous integration and deployment (CI/CD) pipeline

WHAT IS SPRING BOOT?

Spring Boot is an extension of the **Spring Framework** that simplifies the process of building and deploying Spring-based applications. It provides an easy and fast way to set up, develop, and run production-grade Spring applications with minimal configuration.

Spring Boot aims to reduce the complexity of using Spring by providing:

- **Auto-configuration:** Automatically configures your application based on the dependencies in the classpath.
- **Standalone:** Runs independently without needing an external server (uses embedded servers like Tomcat or Jetty).
- **Production-ready:** Includes features like health checks, metrics, and externalized configuration via the Spring Boot Actuator.
- **Spring Initializr:** A web-based tool to generate Spring Boot project templates quickly.

WHY SPRING BOOT?

Traditional Spring applications required a lot of boilerplate code and manual configuration (like XML files). Spring Boot solves this by:

- Automatically configuring beans and dependencies.
- Embedding servers, so no external deployment is needed.
- Providing default project setups.
- Making applications easier to test and run.

When to Use Spring Boot

- Building **RESTful APIs**
- Developing **microservices**
- Creating **web applications**
- Prototyping **quick applications**
- Replacing legacy applications with modern Spring-based solutions

KEY FEATURES OF SPRING BOOT

✅Auto Configuration

Spring Boot automatically configures your application based on the libraries and settings it detects. For example, if Spring Boot finds Spring MVC in your classpath, it sets up a basic web application.

✅ Embedded Servers

Spring Boot supports **embedded servers** such as **Tomcat**, **Jetty**, and **Undertow**, allowing applications to run independently as Java applications without needing deployment to a traditional web server.

✅ Spring Boot Starter Dependencies

Starters are pre-defined dependency descriptors that simplify Maven or Gradle configuration. For example:

- `spring-boot-starter-web` – For building web, including RESTful, applications.
- `spring-boot-starter-data-jpa` – For Spring Data JPA with Hibernate.

Continue…

✅ Spring Boot Actuator

Provides production-ready features like health checks, metrics, environment information, and monitoring endpoints which can be easily integrated with tools like Prometheus and Grafana.

✅ Externalized Configuration

Spring Boot allows developers to configure applications through `application.properties` or `application.yml`, supporting different environments without modifying the codebase.

✅ Spring Initializr

A web-based tool (https://start.spring.io) that helps developers generate Spring Boot project scaffolding quickly with selected dependencies and settings.

Benefits of Using Spring Boot

- **Reduces Boilerplate Code**: Eliminates repetitive configurations.
- **Microservice-Friendly**: Ideal for building microservices due to its modular structure and lightweight setup.
- **Flexible Configuration**: Supports properties, YAML, and environment variables.
- **Integrated Testing Support**: Includes testing libraries and annotations like `@SpringBootTest` for unit and integration testing.
- **Large Ecosystem**: Integrates seamlessly with Spring projects like Spring Security, Spring Data, and Spring Cloud.

Common Use Cases

- Developing **REST APIs**
- Building **Microservices** architecture
- Creating **web applications**
- Developing **data-driven applications** using JPA or NoSQL databases
- Applications that require **quick deployment and scaling**

WORKFLOW OF SPRING BOOT

1. Go to Spring Initializr
2. Choose:
    - Project type (Maven/Gradle)
    - Language (Java/Kotlin)
    - Dependencies (e.g., Spring Web, JPA, H2)
3. Download and import the project into an IDE.
4. Create a main class with `@SpringBootApplication`.
5. Create REST controllers, services, repositories.
6. Run with `mvn spring-boot:run` or `java -jar`.

Controller in Spring Boot

In a Spring Boot web application, a **Controller** is a special Java class that handles **HTTP requests** from users and sends appropriate responses. It is a **core component in the MVC (Model-View-Controller)** architecture, where it acts as the **middle layer** that connects the front-end (user interface) and the back-end (business logic and database).

What Does a Controller Do?

1. **Receives Web Requests:**
   When a user visits a URL (like `http://localhost:8080/hello`), Spring Boot maps that request to a method in the controller.

2. **Processes Input (if any):**
   It can take input from the user through query parameters, path variables, or request bodies.

3. **Calls Business Logic:**
   The controller doesn't usually contain complex logic itself. Instead, it calls methods from the **service layer** to process the request.

4. **Returns a Response:**
   Based on the result, the controller returns a **response**, which can be:

   - JSON/XML (in APIs)
   - An HTML page (in traditional web apps)

CONTINUE…

Types of Controllers in Spring Boot

1. `@RestController` – For REST APIs

- Sends data directly (usually JSON) as the response.
- Useful for backend services or APIs consumed by front-end apps like React, Angular, etc.

This controller will respond with plain text or JSON when a user accesses `/api/hello`.

2. `@Controller` – For Web Pages

- Works with view technologies like **Thymeleaf**, **JSP**, or **FreeMarker**.
- Returns the name of a view (HTML page) instead of raw data

This controller will render an HTML page (`welcome.html`) using a template engine.

CONTINUE…

Common Annotations Used in Controllers

Annotation	Purpose
@Controller	Defines a controller that returns a view (HTML)
@RestController	Returns data (like JSON) directly in the response
@RequestMapping	Maps the base URL path for a controller
@GetMapping	Handles HTTP GET requests
@PostMapping	Handles HTTP POST requests
@PathVariable	Binds URL parameters to method arguments
@RequestParam	Binds query parameters (?key=value) to method arguments
@RequestBody	Binds request JSON to Java objects

CONTINUE…

Entity Class in Spring Boot

In Spring Boot, an **Entity class** is a **Java class that maps to a table in a database**. It is a core concept in **JPA (Java Persistence API)** and is used to define the structure of the data you want to store and retrieve from the database.

✅ Purpose of an Entity Class

- Represents a **table** in the database.
- Each instance of the class represents a **row** in the table.
- The **fields** (variables) in the class map to the **columns** of the table.
- Entity classes are managed by **Spring Data JPA** to perform CRUD operations easily

How to Define an Entity Class

You create an entity by:

- Annotating the class with `@Entity`
- Annotating the primary key field with `@Id`
- Optionally using `@GeneratedValue` for auto-incrementing IDs
- Using `@Column` to customize column names (optional)

CONTINUE…

Entity Relationships in Spring Boot (JPA)

In Spring Boot, when you're working with **JPA (Java Persistence API)** and **Hibernate**, you often need to define **relationships** between different entity classes — just like **foreign keys** in relational databases.

These relationships help you **model real-world data more accurately**, like:

- A user has many orders
- A product belongs to a category
- A course has many students

CONTINUE…

Types of Relationships in JPA

Relationship Type	Example
@OneToOne	A user has one profile
@OneToMany	A customer has many orders
@ManyToOne	Many orders belong to one customer
@ManyToMany	A student can enroll in many courses, and each course has many students

CONTINUE…

Repository in Spring Boot

In Spring Boot, a **Repository** is a special interface that helps you perform **database operations** (like save, delete, find, update) on your **Entity classes** without writing SQL code manually.

It is a key part of the **Spring Data JPA** module, which allows you to interact with databases using standard Java objects.

✅ What is a Repository?

A **Repository** is used to:

- Access and manipulate data stored in the database
- Perform **CRUD** operations (Create, Read, Update, Delete)
- Query data using method names or custom queries

Instead of writing SQL queries, Spring Boot allows you to **extend repository interfaces** like `JpaRepository` or `CrudRepository`, and it generates the implementation for you at runtime.

A **Repository** in Spring Boot is a powerful and easy way to interact with the database. It takes care of all the heavy lifting related to data access so that you can focus on building business logic. With just a few lines of code, you can create full-featured data layers for any application.

CONTINUE…

Spring Security

Spring Security is a powerful and customizable **authentication and authorization framework** for Java applications, especially those built with **Spring Boot**. It helps developers secure their applications from common security threats like **unauthorized access**, **CSRF**, **session fixation**, and more.

What Does Spring Security Do?

Spring Security helps you:

1. **Authenticate users** – Verify who the user is (login).
2. **Authorize users** – Control what actions a user can perform (roles/permissions).
3. **Protect resources** – Secure APIs, web pages, or data based on user roles.
4. **Prevent security attacks** – Like CSRF, XSS, and clickjacking.

Basic Setup in Spring Boot

- All endpoints are **secured by default**.
- Spring shows a **default login page**.
- A default user with a generated password is created

CONTINUE…

Security Features Spring Handles

Feature	Spring Security Handles
Authentication	✅
Authorization	✅
CSRF Protection	✅
Password Encryption	✅
Role-Based Access	✅
Session Management	✅
Custom Login/Logout	✅

Key Functional Modules of the Admin Dashboard

1. 📂 Category Management

Purpose:
Ensure the platform maintains a clear structure by managing the available categories under which clients post projects and freelancers offer services.

Functionality:

- Add new categories (e.g., Web Development, Digital Marketing, UI/UX Design)
- Edit existing category names or descriptions
- Delete or deactivate unused or spammy categories
- Organize categories into parent–child (subcategories)

Benefits:

- Keeps the marketplace organized
- Allows for scalability by adding new industry sectors

CONTINUE..

2. 👥 User & Freelancer Management

🔸 **User (Client) Management**

- View list of all registered clients
- Search by name, email, registration date, status
- Activate, deactivate, or ban accounts
- View client profile, project activity, and reports
- Handle reported users (flagged by freelancers)

🔹 **Freelancer Management**

- View list of all registered freelancers
- Review profiles, skills, ratings, and job history
- Approve or reject profile updates or verifications
- Ban accounts involved in misconduct
- Review service request patterns or spam reports

Benefits:

- Ensures safety and trust in the community
- Helps moderate inappropriate behavior or spam

Key Functional Modules of the Client Dashboard

1. 🗂️ Add Projects by Category

Functionality:

- Clients can create and post new projects under specific predefined or custom **categories** (e.g., Web Development, Graphic Design, Content Writing).

Project Details Include:

- Project Title
- Category (dropdown selection)
- Budget range
- Description & objectives
- Required skills
- Deadline or time duration
- Attachment (optional)

Benefits:

- Helps freelancers filter and find relevant jobs
- Keeps the platform organized by domain expertise

CONTINUE…

2. 📩 View Service Requests from Freelancers

Functionality:

- Freelancers can send **requests to work on projects** posted by the client.
- The dashboard displays a **list of incoming service requests** with:
    - Freelancer name & profile preview
    - Proposed rate / quote
    - Message or proposal content
    - Skills and previous reviews (if available)

Client Actions:

- Accept / Reject request
- View freelancer profile in detail
- Message the freelancer directly for discussion

CONTINUE…

3. ✅ Oversee Project Completion Requests

Functionality:

- After completing the assigned project, freelancers can mark a **Project Completion Request**.
- Clients are notified and can:
    - Review the submitted work/files
    - Approve or request changes
    - Release payment (if integrated)
    - Leave feedback/review

Status Tracking:

- Track ongoing → under review → completed → archived status

Benefits:

- Smooth workflow with accountability for both parties
- Ensures proper delivery before marking a job as complete

CONTINUE…

4. 🧑‍💼 Client Profile Management

Functionality:

- Clients can view and edit their personal or business profile information:
    - Name / Company Name
    - Email, Contact Info
    - Profile picture or logo
    - Bio / Description
    - Preferred categories / skills
    - Location

Additional Features:

- Upload documents (e.g., business verification, brand guidelines)
- Link social profiles (LinkedIn, website)

Key Functional Modules of the Freelancer Dashboard

1. 📥 Apply for Projects

Functionality:

- Freelancers can **browse and search** for projects posted by clients.
- Apply for relevant projects by submitting a **service request** which includes:
    - A proposal or message
    - Expected budget
    - Estimated timeline
    - Previous work or sample attachments (optional)

Benefits:

- Easy way to pitch to clients
- Helps freelancers find work suited to their skills and preferences

CONTINUE…

2. 📌 View Applied Project Requests

Functionality:

- List of all projects the freelancer has **applied for**
- Each request displays:
    - Project title, category, client info
    - Date of application
    - Status (Pending / Accepted / Rejected)

Benefits:

- Track the status of each application
- Take follow-up action if required

CONTINUE…

3. ✅ Approved Projects Management

Functionality:

- Freelancers can view a separate list of **approved/accepted projects**
- Project details include:
    - Client name & contact
    - Full project brief
    - Deadline and agreed budget
- Option to **start work** and track progress

CONTINUE…

4. 🧾 Submit Completed Work

Functionality:

- Once the freelancer completes a project, they can:
    - Upload final deliverables (files, links, etc.)
    - Send a **Project Completion Request** to the client
- Await client review and approval

Benefits:

- Structured handoff system
- Ensures freelancers get acknowledgment for completed work

CONTINUE…

5. 📋 View Completed Projects

Functionality:

- Freelancers can access a list of all **projects marked as completed**
- Each project displays:
    - Client feedback
    - Ratings
    - Payment status (if applicable)
    - Downloadable copy of final submission (optional)

Benefits:

- Acts as a **portfolio** of past work
- Helps in profile credibility for future applications

CONTINUE…

6. ✏️ Edit Freelancer Profile

Functionality:

- Freelancers can manage and update their profiles including:
    - Name, email, contact info
    - Bio / About section
    - Skills & services offered
    - Profile picture
    - Resume or portfolio links

Benefits:

- Keeps their profile attractive to clients
- Increases chances of being hired

CONCLUSION

ProConnect is more than just a platform — it's a bridge that seamlessly connects talented freelancers with clients seeking quality services. Designed with a user-centric approach, ProConnect empowers **clients** to post projects, manage workflows, and hire the right professionals, while offering **freelancers** a dynamic space to showcase their skills, apply for projects, and grow their careers.

With dedicated dashboards for clients, freelancers, and administrators, ProConnect ensures smooth communication, transparent transactions, and an efficient project lifecycle. From category-wise project posting and service request handling to analytics and profile management — every feature is built to foster trust, productivity, and professional growth.

ProConnect aims to revolutionize the freelance marketplace by creating a smart, secure, and scalable environment where work meets opportunity.

REFERENCES

Open-Source & Online Projects for Reference

- GitHub Open-Source Freelancing Projects – Used to study the architecture of existing freelancing platforms.

https://github.com

- Online Freelancing Platform Case Studies – Research papers and real-world examples.

Upwork

. **AI-Based Assistance & Research**

- ChatGPT (OpenAI) – Used for structuring project documentation, refining explanations, and debugging.

https://chat.openai.com

- Stack Overflow – Used for resolving coding issues and best practices in development.

https://stackoverflow.com

CONTINUE…

YouTube Tutorials & Online Learning

- Java Techie (YouTube Channel) – Used for Spring Boot, Hibernate, REST APIs, and authentication tutorials.
- https://www.youtube.com/@JavaTechie
- Code With Durgesh (YouTube Channel) – Spring Boot and Hibernate tutorials.
- https://www.youtube.com/@CodeWithDurgesh
- Programming with Mosh – Java, frontend, and backend development tutorials.
- https://www.youtube.com/c/programmingwithmosh

UI/UX & Frontend Development

- W3Schools – Quick references for HTML, CSS, JavaScript, and responsive design.

https://www.w3schools.com

- Tailwind CSS – Utility-first CSS framework used for styling.

https://tailwindcss.com

CONTINUE…

Search Engines & Documentation

- Google – Used for general research, troubleshooting, and finding relevant documentation.

https://www.google.com

- Spring Boot Official Documentation – Comprehensive reference for Spring Boot, MVC, security, and REST API development.
- https://spring.io/projects/spring-boot
- MDN Web Docs – Used for HTML, CSS, JavaScript, and frontend-related concepts.

https://developer.mozilla.org

Security & Authentication References

- Spring Security Official Guide – Used for authentication, authorization, and role-based access control.
- https://spring.io/guides/gs/securing-web

THANK YOU