Decentralized Job Portal – Complete Project Report & Implementation Guide

1. Introduction

The Decentralized Job Portal is a blockchain-powered recruitment platform designed to bring transparency, security, and authenticity to the hiring process. Unlike traditional platforms where data is controlled by centralized authorities, this system empowers users through decentralized identities and smart contracts.

2. Objectives

- Eliminate fake resumes and fraudulent credentials using blockchain verification.
- Build a trustless environment where job seekers, recruiters, and validators interact directly.
- Automate job offers, payments, and agreements through smart contracts.
- Ensure user control over personal data and privacy.

3. Target Users

The application serves three primary user groups:

• Job Seekers – Create blockchain-based profiles with verifiable credentials and apply for jobs.
• Employers/Recruiters – Post jobs, verify candidate data, and manage hiring via smart contracts.
• Validators – Provide unbiased assessment and verification services recorded on-chain.

4. Critical Features

1. Decentralized Authentication – Onboarding through blockchain-based digital identities (DIDs).
2. Credential Verification – Immutable storage and validation of resumes and certificates via IPFS.
3. Job Posting and Search – Employers post listings; candidates search through decentralized queries.
4. Smart Contract Job Offers – Automated contract generation for job offers and payment terms.
5. Application Tracking – Transparent updates on hiring status.
6. Reviews and Feedback – Immutable performance reviews stored on-chain to build reputation history.

5. Technology Stack

Frontend: Next.js (React framework)
Backend: Node.js (Next.js API routes)
Blockchain: Solidity (Ethereum / Polygon)

Storage: IPFS for documents and resumes
Database: MongoDB for user metadata
Wallet: MetaMask integration via Ethers.js

6. System Architecture

The system architecture is composed of three main layers:

1. **Frontend Layer** – Built with Next.js, providing an interactive UI and Web3 wallet integration.
2. **Smart Contract Layer** – Developed in Solidity, handles job postings, applications, verification, and automated agreements.
3. **Off-chain Storage Layer** – Uses IPFS for file storage (resumes, certificates) and MongoDB for basic metadata.
4. **Blockchain Network** – Ethereum or Polygon for on-chain logic execution.

7. Implementation Steps

Step 1: Initialize Next.js project and configure environment.
Step 2: Set up Web3 wallet connection using Ethers.js.
Step 3: Develop Solidity smart contracts for job postings, applications, and payments.
Step 4: Deploy contracts using Hardhat or Truffle.
Step 5: Integrate IPFS for file uploads and storage.
Step 6: Build Next.js pages for user registration, job listing, and application management.
Step 7: Connect frontend with blockchain through ABI and contract addresses.
Step 8: Implement feedback and review features on-chain.
Step 9: Test entire system with mock data.
Step 10: Deploy frontend on Vercel and backend contracts on testnet (e.g., Polygon Mumbai).

8. User Stories

• As a job seeker, I want to create a verified blockchain profile so my credentials cannot be falsified.
• As an employer, I want to post job openings and have applications automatically managed by smart contracts.
• As a validator, I want to provide unbiased assessment scores that are securely stored on blockchain.
• As a recruiter, I want to verify candidate credentials instantly and send automated offers.
• As a job seeker, I want to control who views my personal information.

9. Academic Rubric Alignment

This project satisfies the academic rubric under the following areas:

• Technical Implementation – Demonstrates integration of Web3 technologies (Next.js, Solidity, IPFS).
• Innovation – Decentralized identity and trustless job ecosystem.

• Documentation – Clear explanation of architecture, functionality, and implementation.
• Problem Solving – Addresses real-world hiring fraud and data transparency issues.
• Teamwork and Management – Can be developed collaboratively with defined frontend, backend, and smart contract roles.

10. Future Enhancements

1. Integration of AI for candidate-job matching.
2. Implementation of DAO governance for platform moderation.
3. Mobile DApp version using React Native.
4. NFT-based verifiable employment records.
5. Layer-2 migration for lower gas fees and faster transactions.

11. Conclusion

The Decentralized Job Portal redefines recruitment through transparency and blockchain-based trust. By leveraging Next.js, Solidity, and IPFS, the system ensures data authenticity, automation, and user empowerment. This project provides a strong foundation for decentralized HR ecosystems and aligns with the future of Web3 employment.