International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Kaamsetu An Integrated Worker Booking and Field Labour Management System

Guide - Prof. Vaishnavi Patharwat[1]

Niyati Rajesh Battul[2]

Rajeshwari Mangesh Mane[3]

Iqra Feroz Tamboli[4]

Ayush Sunil Gaikwad[5]

Sumit Prakash Rathod[6]

Department of Artificial Intelligence and Machine Learning,

Zeal Polytechnic, Pune, Maharashtra, India

ABSTRACT

Finding suitable workers, arranging bookings, and coordinating field labour can be difficult when information is maintained through phone calls, informal contacts, or separate records. This paper presents KaamSetu, an integrated worker booking and field labour management system intended to organize interaction between customers, workers, and administrative users in one digital platform. The system's functional scope includes worker information management, service requests, booking records, booking-status updates, administrative coordination, and easy to login, subject to the features implemented in the submitted application. The system accepts a service request, validates the submitted information, records the booking, and makes relevant status information available to authorized users. The application was developed using Python for programming language streamlit for frontend framework, and SQLite for Database. Scikit-learn and pandas for text processing and worker matching. The system was tested using manual functional testing with predefined test cases to check user registration, job posting, worker matching, booking management and suspicious activity review. Verified test counts and measured results should be added from the completed application before final submission. It describes the problem, related work, system architecture, workflow, functional modules, implementation, and evaluation approach .

Index Terms— Worker booking, field labour management, service platform, job coordination, digital records, KaamSetu.

I. INTRODUCTION

Workers provide essential services in construction, repair and maintenance, agriculture, household work, and other labour-intensive activities. Customers and organizations need to find workers with suitable skills and availability, and also workers need a clear way to receive work requests and manage assignments. When these activities rely on phone calls, personal contacts, paper records, or separate messaging channels, booking details can become scattered and updates may not reach every relevant person.

KaamSetu is an integrated worker booking and field labour management system designed to organize these activities through a centralized application. Its purpose is to support the process of easy to use for different types of users, users can entered their job using text typing, Audio based input (voice typing) recording worker details, receiving service requests, managing booking status, and coordinating field work. The system brings customers, workers, and administrative users into a defined workflow rather than treating each activity as a separate record-keeping task.

A. Problem Statement

The main problem addressed by KaamSetu is the difficulty of coordinating worker discovery, service booking, and field labour records through disconnected or manual processes. Such processes may create duplicate information, unclear booking status, scheduling conflicts, and extra effort for administrators. The project therefore aims to provide a centralized system for organizing worker profiles , service bookings and for making the status of a request easier to follow. May also be supported to make job entry more convenient especially for users who are less comfortable with typing. However,the using audio input is optional.

B. Objectives

The objectives are: (1) to organize worker and service information; (2) to provide a structured booking-request workflow; (3) to record and display booking status; (4) to support field-work coordination and administrative monitoring according to the implemented features; (5) to reduce reliance on scattered manual records; and (6) to

evaluate the application's functions through documented test cases.

C. Scope

The project focuses on the digital coordination of customers, workers, and labour-management employee. The current paper describes the core booking and management workflow. Features such as GPS-based matching, online payment, attendance tracking, automatic worker recommendations, and push notifications should be included as implemented features only if they exist and have been tested in the submitted application.

II. LITERATURE REVIEW

A. Digital Service Platform

Digital service platforms provide structured ways for users to discover services, submit requests, and communicate with service providers. Research on platform-mediated work describes how digital platforms coordinate demand and labour, while also noting that platform design can affect work allocation, information visibility, and the relationship between workers and customers [1], [2]. These findings provide context for considering how a worker-booking system should organize user roles and service requests.

B. Workforce and Field-Work Coordination

Workforce-management systems commonly support employee records, scheduling, task assignment, attendance, and operational reporting. The exact capabilities vary by application. For field labour, the coordination process may require a clear record of the requested task, the worker assigned, the relevant location or schedule, and the current status. A booking application can help organize these details, but its actual usefulness depends on implementation quality, user adoption, and reliable data entry.

C. Research Gap and Project Motivation

The motivation for KaamSetu is to bring worker booking and field labour information into one practical workflow. This manuscript does not claim that worker booking or labour management is a new research area. Instead, it presents a project-level integration of the features implemented by the student team. A stronger research-gap claim should be added only after comparing KaamSetu with specific existing applications and peer-reviewed studies.

III. PROBLEM STATEMENT AND OBJECTIVES

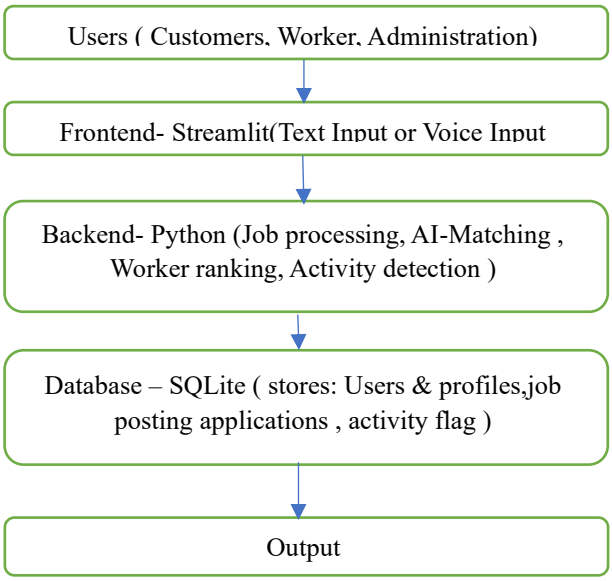
Customers may find it difficult to identify available workers and keep track of a request when information is exchanged through multiple channels. Workers may need a clearer record of incoming jobs, while administrators may need to review worker information and booking progress. The system must also validate submitted information and handle incomplete or invalid requests appropriately.

The project objectives are to provide a common interface for the relevant user roles; maintain organized worker and booking records; support the booking lifecycle defined by the application; make status information available to authorized users; and test the main workflows under normal and exceptional input conditions. The precise user roles, booking states, and access permissions must match the final implementation.

IV. PROPOSED SYSTEM

A. System Architecture

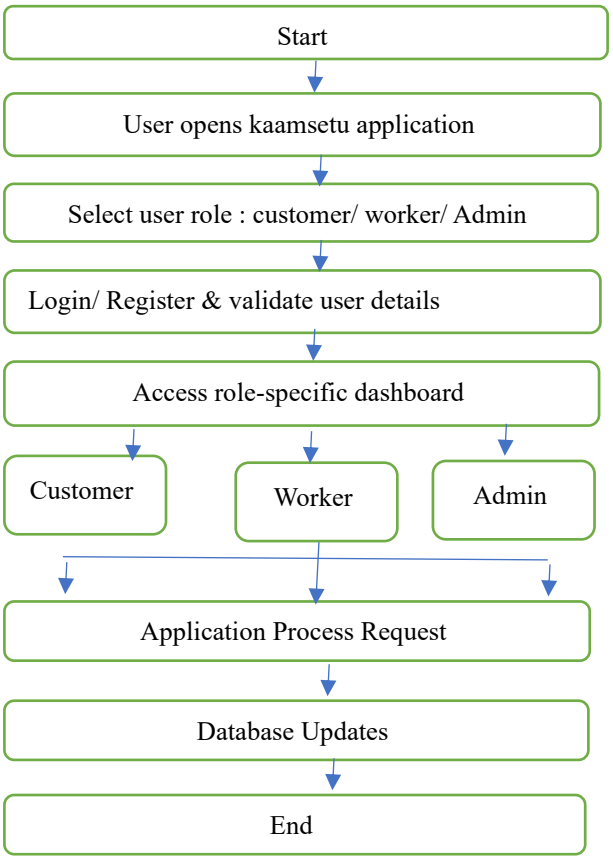
The system can be described using four logical components: (1) a user interface for customers, workers, and administrators; (2) application logic for validating requests and processing booking operations; (3) a data-storage layer for user, worker, and booking records; and (4) a management interface for reviewing relevant records. The technologies used for each component should be identified from the actual source code and configuration files.



B. Workflow

The general workflow begins when a user accesses the application. Ass customer selects or describes a service and submits the required booking details. The application validates the submitted information and stores the request if it is valid. The request is then handled according to the system's implemented assignment process. The booking status is

updated as permitted by the workflow, and relevant users can view the recorded status. Administrators review worker and booking information according to their permissions.



C. Functional Modules

User management handles user details and, where implemented, registration, login, and role-based access. Worker management stores relevant worker information such as skills, service category, and availability if those fields are supported. Booking management records service requests and their current status. Field labour management organizes job assignments and work updates to the extent implemented. Administrative management provides authorized users with access to records needed to oversee the booking process.

D. Data Management

The data model may include user, worker, booking, and job-assignment records. A booking record commonly needs a unique identifier, customer reference, requested service, date or time, location if applicable, and status. Worker records may contain profile information and service-related details. The final paper must describe the actual database tables, fields, relationships, and validation rules used by KaamSetu rather than this generic model.

E. Technologies

The implementation uses [Python], [Streamlit], [Python], and [SQLite]. The application was developed and tested in [development environment].

V. PROSPECTIVE BENEFITS

KaamSetu is intended to provide a single place to organize worker information and booking requests. A centralized record may reduce the need to search through separate messages or paper notes. A defined booking status can help users understand whether a request is awaiting action or has progressed to another state. Administrative access to organized records may also make it easier to review requests and coordinate field work.

These are intended benefits, not measured findings. Any statement that KaamSetu reduces booking time, improves worker utilization, increases customer satisfaction, or lowers errors should be supported by an appropriate comparison, user study, or recorded test result.

VI. STRATEGY FOR IMPLEMENTATION AND ASSESSMENT

A. Implementation

The development process consists of requirement identification, interface design, data-model preparation, implementation of the booking workflow, integration of the user and management functions, and functional testing. The actual implementation details has include the application structure, key modules, validation logic, database operations, and any authentication or authorization controls used.

B. Functional Testing

Testing should verify the main user journeys, including valid booking submission, worker-record handling, booking-status changes, incomplete form submissions, and unauthorized or invalid actions where applicable. Each test case is record its input, expected result, actual result, and pass/fail outcome.

C. Evaluation Measures

For functional testing, the test-pass rate is calculated as: (passed text cases / total test cases executed) × 100. If response time is measured, report the hardware, software version, test conditions, number of repetitions, and the method used to calculate the average. User feedback may be reported only if it was collected using a documented procedure.

VII. CONSTRAINTS, PRIVACY, AND ETHICAL IMPLICATIONS

The system's usefulness depends on the accuracy of worker profiles, service descriptions, availability information, and booking updates. Incorrect or outdated information is received , inappropriate tasks may be assigned, or errors in communication may occur. Internet availability, device compatibility, and user familiarity with digital systems may also affect adoption.

Worker and customer information should be collected only when needed for the service, protected from unauthorized access, and retained according to a defined policy. Contact

details, addresses, and job records are visible only to authorized users. If location data, identity documents, or payment details are collected, the final implementation should document how they are protected and when they are deleted. The paper should describe existing safeguards accurately and must not claim security testing or legal compliance unless those checks have been performed.

The system should communicate booking status clearly and provide a way to correct inaccurate records. Any automated worker ranking or recommendation, if present, should be explainable to users and evaluated for unfair outcomes. Human review should remain available for disputes or exceptional cases

VIII. CONCLUSION AND FUTURE SCOPE

This paper presents KaamSetu, an Integrated Worker Booking and Field Labour Management System intended to organize the interaction between customers, workers, and administrative users. The system's core purpose is to centralize worker information, service requests, booking records, and relevant status updates in a structured workflow. The completed application should be evaluated through documented functional tests and actual observations.

Future development may include location-aware worker discovery, attendance records, notifications, online payment integration, job-history reports, or data-driven worker recommendations. These are future possibilities unless already implemented.

ACKNOWLEDGEMENT

The authors would like to thank [Vaishnavi Paharwat], the faculty members of the Department of Artificial Intelligence and Machine Learning at [Zeal Institute], and all individuals who provided guidance and feedback during the development of KaamSetu. Replace this acknowledgement with the names and contributions approved by the project team.

REFERENCES

[1] A. Vallas and J. B. Schor, “What Do Platforms Do? Understanding the Gig Economy,” Annual Review of Sociology, vol. 46, pp. 273–294, 2020. doi: 10.1146/annurev-soc-121919-054857.

[2] V. De Stefano, “The Rise of the ‘Just-in-Time Workforce’: On-Demand Work, Crowdwork, and Labor Protection in the ‘Gig-Economy’,” Comparative Labor Law & Policy Journal, vol. 37, no. 3, pp. 471–504, 2016.

[3] [Author(s)], “[Title of a peer-reviewed study on online service booking or labour-management systems],” [Journal/Conference], [volume/issue/pages], [year], [DOI/URL].

[4] [Organization], “[Name of the actual framework/database documentation used in KaamSetu],” [URL], accessed [date].

[5] [Author(s)], “[Title of a relevant software testing or usability study],” [Journal/Conference], [publication details], [year], [DOI/URL].