Technical Design Document (TDD) – *Helping Hand*

*INTRODUCTION*

The Helping hand is a community-driven web application aimed at simplifying the process of finding and connecting with trusted local service providers. It addresses the common challenge of locating reliable professionals—such as electricians, plumbers, tutors, and more—by allowing users to search based on location, view verified profiles, read community reviews, and make service bookings. The platform promotes transparency, trust, and convenience through features like user ratings, service recommendations, and real-time notifications, while also providing service providers with tools to manage their offerings and engage with potential customers.

*Technology Stack*

|  |  |
| --- | --- |
| **Layer** | **Technology Used** |
| Frontend | Angular, Bootstrap |
| Backend | Java (Spring Boot) |
| Database | MySQL |
| API Format | RESTful (JSON) |
| Version Control | GitHub |

*3. System Architecture*

3.1 High-Level Architecture Overview

The Helping Hand platform is a web-based community service provider system designed to connect local service providers with customers. It features multiple user roles — Customers, Service Providers (Helpers), and Administrators — each with tailored functionality. The architecture ensures smooth interaction, data management, and security throughout the platform.

3.2 Components

3.2.1 Client Side (Browser / User Interface)

* Technology: Responsive web UI (Angular, Bootstrap React templates with Spring Boot).
* Responsibilities:
  + Display landing page with search, top performers, and customer testimonials.
  + Provide forms for Service Provider and Customer sign-ups, and Login/Registration functionality.
  + Render dashboards for logged-in users with personalized information (profile, booking history, etc.).
  + Facilitate navigation to services listing, detailed service profiles, and booking workflows.
  + Display dynamic content such as top services filtered by city and profession.
  + Show ratings, and contact information clearly.
  + Maintain branding consistency via logos, images, and UI themes.

3.2.2 Web Server (Spring Boot Application)

* Technology: Spring Boot framework (Java) for backend logic.
* Responsibilities:
  + User Management: Handle registration, authentication, password recovery.
  + Role-Based Access: Differentiate behaviour for Customers, Service Providers, and Admins.
  + Business Logic: Manage service listings, provider profiles, booking requests and ratings.
  + Search & Filtering: Process user queries based on location, profession, and other filters.
  + Data Flow Control: Validate inputs from the UI, transform data for persistence, and retrieve data for display.
  + Admin Dashboard: Oversee user activity, manage reports, and monitor content credibility.
  + File Handling: Support image uploads for profile pictures and service banners

3.2.3 Database (MySQL)

* Technology: Relational Database Management System (MySQL).
* Responsibilities:
  + Store user profiles including service providers and customers with fields such as name, contact info, location, and photographs.
  + Maintain service listings with service details, categories, and provider associations.
  + Record user ratings linked to service providers.
  + Log booking history and status updates for customers and helpers.
  + Manage authentication data including usernames, hashed passwords, and roles.
  + Store admin data and activity logs.

3.3 System Workflow Summary

1. Landing Page Interaction: Users arrive, browse top performers, read reviews, or search services.
2. Registration: Users choose to sign up as Customers or Service Providers via separate sign-up forms.
3. Authentication: Login via username/password with appropriate authentication checks.
4. User Dashboard: Post-login, users access a personalized dashboard displaying services, search options, profile info, and booking history.
5. Service Discovery: Customers search for services by location and profession; filtered results display service provider listings.
6. Service Provider Profile: Detailed view of selected service providers with ratings, completed work, and contact info.
7. Admin Monitoring: Admins monitor platform activity, user reports, and ensure data integrity.

*Architecture Overview*

// this part is just for explanation , I am just telling how it can be done , all this text will be removed in the final tdd

Detail explanation of architechutre overview , description of each component   
for example

Controller

UI

and moving forward with other   
we need to explain every component

after this a new section can be added for the explanation of ui