**DevConnect: A Developer Collaboration and Career Platform**

Kou Kimheng, Nop Relexsun, Path Nimol, Heng Keanghor,

Hak Kimheng, Hoeurn Somany, Yean Devit, Sek Pisal

Korea Software HRD Center

#12, St. 323, Sangkat Boeung Kak II, Khan Toul Kork, Phnom Penh, Cambodia

[info.kshrd@gmail.com](mailto:info.kshrd@gmail.com)

{kimhengkou1604 , relexsunnop, pathnimol0425, enghourheng26, Hakkimhengg, zuukie333, suerporlong123, pisalsek88}@gmail.com

Mentors: Ms. Vey Seavlang, Mr. Penh Seyha and Mr. Srey Sitharo

**Abstract**. **DevConnect** is a platform designed to bridge the gap between developers and recruiters by solving the problem of finding the right talent and showcasing skills in the technology sector. Our platform is built using **Spring Boot**, **Next.js**, **PostgreSQL**, and **GitHub APIs**. **DevConnect** supports developers in showcasing their skills and growing their careers through features like project collaboration, coding challenges, and resume building. For recruiters, the platform offers job postings and access to talent demonstrated through real activity, making hiring more efficient and targeted. **DevConnect** aims to easier and encourage teamwork while also building a strong technology community. The platform enhances career growth for developers while streamlining and improving the recruitment process.

**Keywords:** Developer Platform, Spring Boot, Next.js, Job Recruitment, Web Application

1. **Introduction**

The technology industry evolves rapidly, requiring developers to continuously improve their skills with innovative ideas. Developers struggle to showcase their true abilities, find meaningful collaboration opportunities, and access career-building challenges such as hackathons or project work in one place. Meanwhile, recruiters face difficulties identifying candidates with the right skills. Despite numerous platforms for job hunting, coding practice, and networking, these tools remain fragmented. Developers manage separate accounts for portfolios, resumes, and projects, while recruiters sift through incomplete or outdated information. Such lack of integration wastes time, creates friction, and hinders progress in a field that thrives on efficiency and innovation. To address that problem, we created **DevConnect**, a single platform that brings developers and recruiters together in one place.

**DevConnect** allows developers to participate in coding challenges, join hackathons, collaborate on projects, discuss the latest technologies, and apply for job opportunities that match their skills. At the same time, it gives recruiters a better way to find and evaluate talent based on real contributions rather than relying solely on resumes. Through **DevConnect**, developers can grow their skills and build a career path, while recruiters gain easier access to engaged and capable candidates. The platform’s main goal is to support both technical growth and career development. **DevConnect** brings together essential tools like resumes, project showcases, job posts, and discussion forums in one unified space. While some features such as chat messaging, live collaboration, and a resume builder with multiple template options are planned for the future, our paper focuses on the current version of the system. The following sections cover how **DevConnect** was designed, what features it includes, how it was implemented, and the potential impact it can have on the experience between developers and recruiters.

1. **Background**

The traditional method of finding a good repair service typically involves calling around, asking friends for referrals, or thumbing through directories. While they work well enough, they are very time-consuming and generally yield less than desirable results. This only becomes more of a problem in emergency circumstances when people need an immediate fix for fixing their devices or appliances. As technology continues to progress, customers more and more want to find services easily and quickly online.

Existing service listing sites such as **Yelp** or **Angie’s List** provide listings of trader but rarely interactive features to make experience smooth. Additions of the requirement for assistance cannot be rapidly posted by the users. Their experience cannot be rated, or communication between traders and clients cannot be facilitated on these sites. Traders also lack significant tools for advertising their services or communicating directly with potential clients in real time.

**KbaeNak** was developed to address the issues through offering a platform where users were able to post service requests and find proximal repair suppliers in real-time. The system enhances the user experience through offering real-time chat functionality, which allows users to communicate directly with traders. Additionally, the system’s rating and review feature allows users to make informed decisions on which service suppliers to trust. **KbaeNak** seeks to bridge the gap between users and traders, making the repair service process more efficient and transparent.

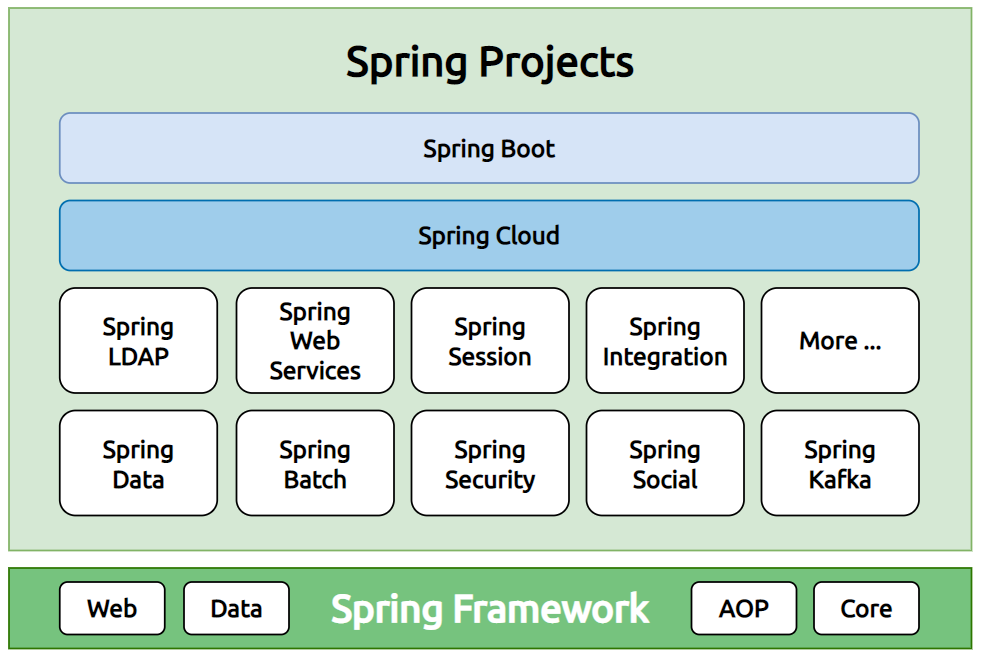
The applicability of this research is the ability to improve the access to repair services and improve the ability of small traders to scale their businesses. As digital platforms continue to grow, there are increasingly more businesses that can access customers online today, and **KbaeNak** takes advantages of this trend to create a win-win situation for both traders and users.

1. **System Architecture**

The system architecture is a conceptual model that defines the structure, behavior, and other views of a system. The purpose of system architecture activities is to define a comprehensive solution based on principles, concepts, and properties that are logically related and consistent with each other. An architecture description is a formal description and representation of a system organized in a way that supports reasoning about the structures and behaviors of the system.

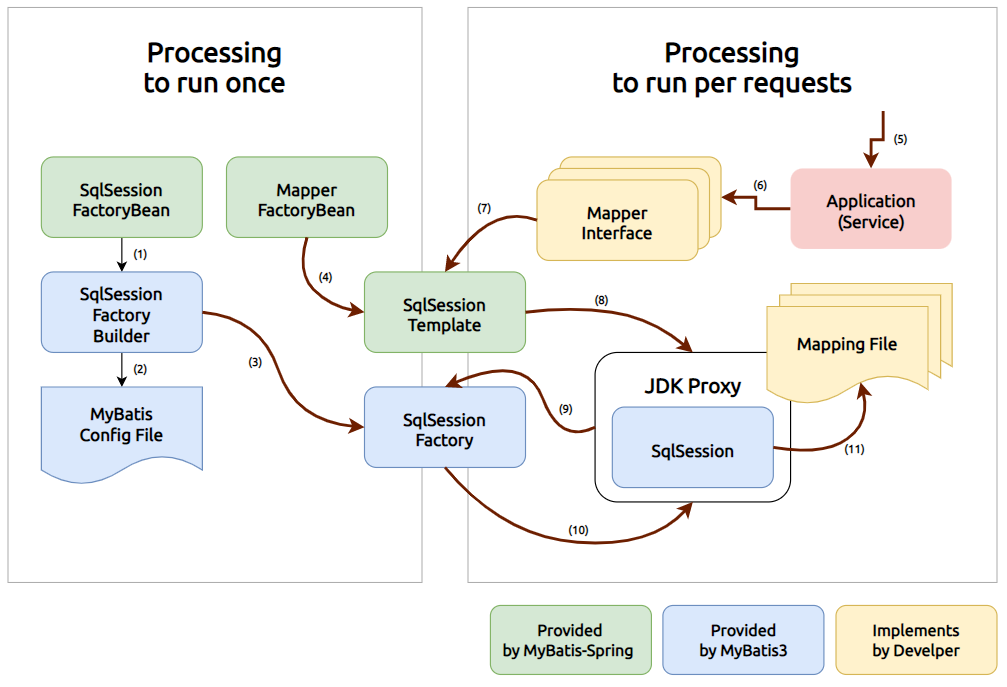
* 1. **Selected Technology**
* Software Use:
  + **IDE**: IntelliJ IDEA, Visual Studio Code
  + **Design**: draw.io, Figma
  + **Communication**: Google Drive, Telegram, Google meet, Morningmate
  + **Documentation**: Microsoft Office, Google Docs, Google Slides
* **Client Side**: HTML, CSS, Tailwind CSS, JavaScript, Next.js
* **Database**: PostgreSQL
* **Spring Boot**: Spring is a framework that provides a comprehensive programming and configuration model for modern Java-based enterprise applications on any kind of deployment platform.

**Figure 1.** Spring Boot



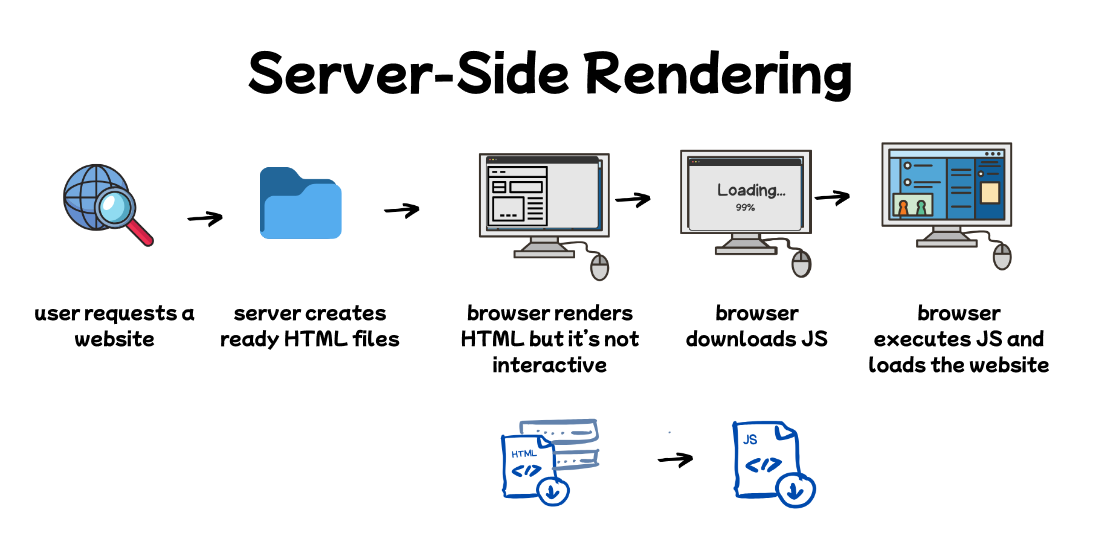
* **Spring MyBatis**: MyBatis is a SQL Mapping framework with support for custom SQL, stored procedures, and advanced mappings.

**Figure 2.** Spring MyBatis

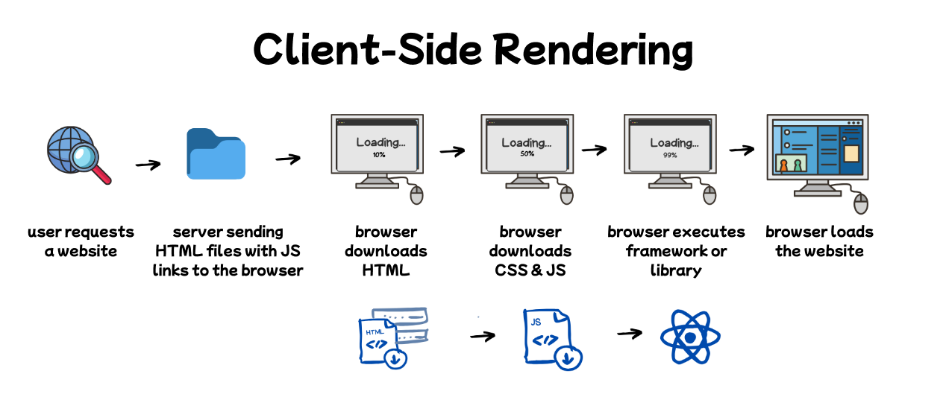


* **HTML**: stands for Hypertext Markup Language. It allows the user to create and structure sections, paragraphs, headings, links, and blockquotes for web pages and applications.
* **CSS**: Cascading Style Sheets. It is a simple design language intended to simplify the process of making web pages presentable.
* **Tailwind CSS**: is a utility-first CSS framework for rapidly building custom user interfaces. It is highly customizable and gives you all of the building blocks you need for designing a web application.
* **JavaScript**: is a scripting language used to create and control dynamic website content.
* **Next.js**: is a React framework for building full-stack web applications by using React Components to build user interfaces, and Next.js for additional features and optimizations.

**Figure 4.** Server-Side Rendering

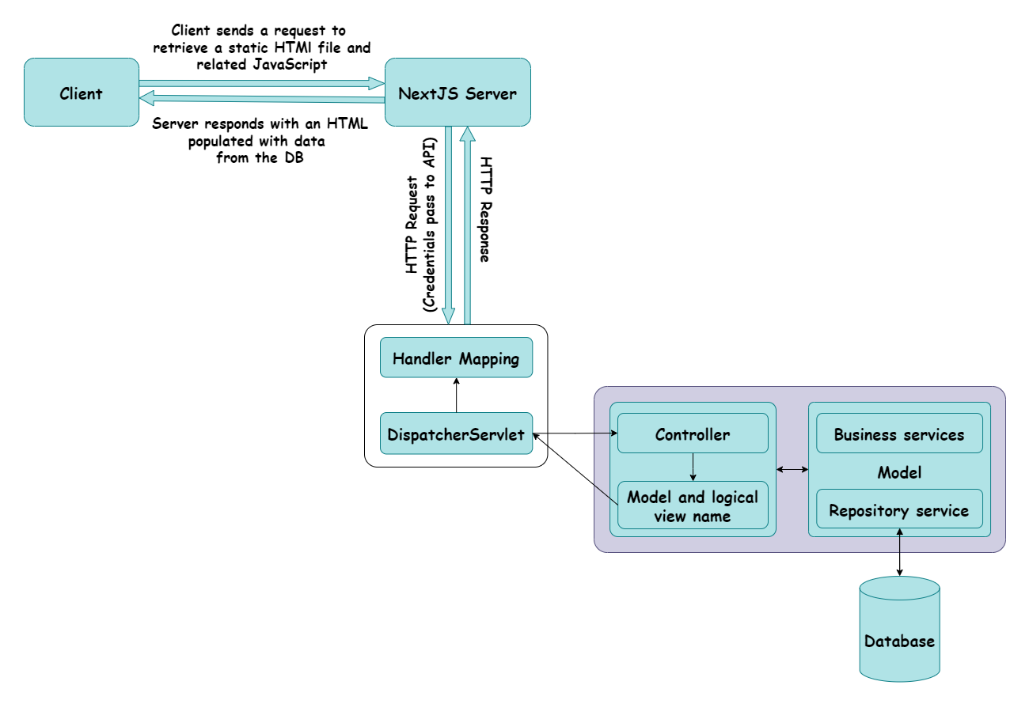


**Figure 3.** Client-Side Rendering

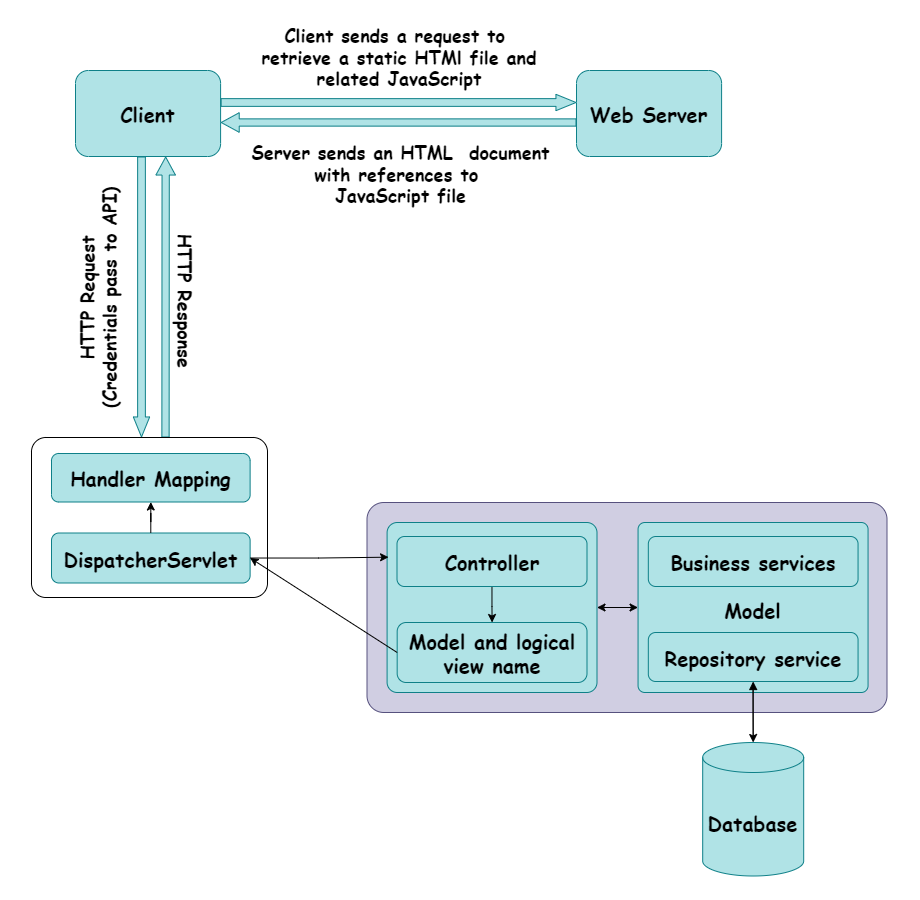


* 1. **Selected Pattern**
* In Next.js, there are two rendering environments where web applications can be rendered: the client and the server.
  + **Client-Side Rendering**: the client refers to the browser on a user’s device that sends a request to a server for your application code. It then turns the response from the server into a user interface.
  + **Server-Side Rendering**: the server refers to the computer in a data center that stores your application code, receives requests from a client, and sends back an appropriate response.
* To access data in the API, we need to authenticate with JWT when we want to request something.
* Spring Boot has Repository, Service, and RestController to work with data in PostgreSQL and return data back to view.
  + **Repository**: is responsible for communicating with the used data storage.
  + **Service**: is responsible for encapsulating the business logic implementation, centralizing data access, and defining where the transactions begin and end.
  + **RestController**: is responsible for processing the user’s input and returning the correct response back to the user.

**Figure 5**. Next.js server-side rendering with Spring Boot server



**Figure 6**. Next.js client-side rendering with Spring Boot server

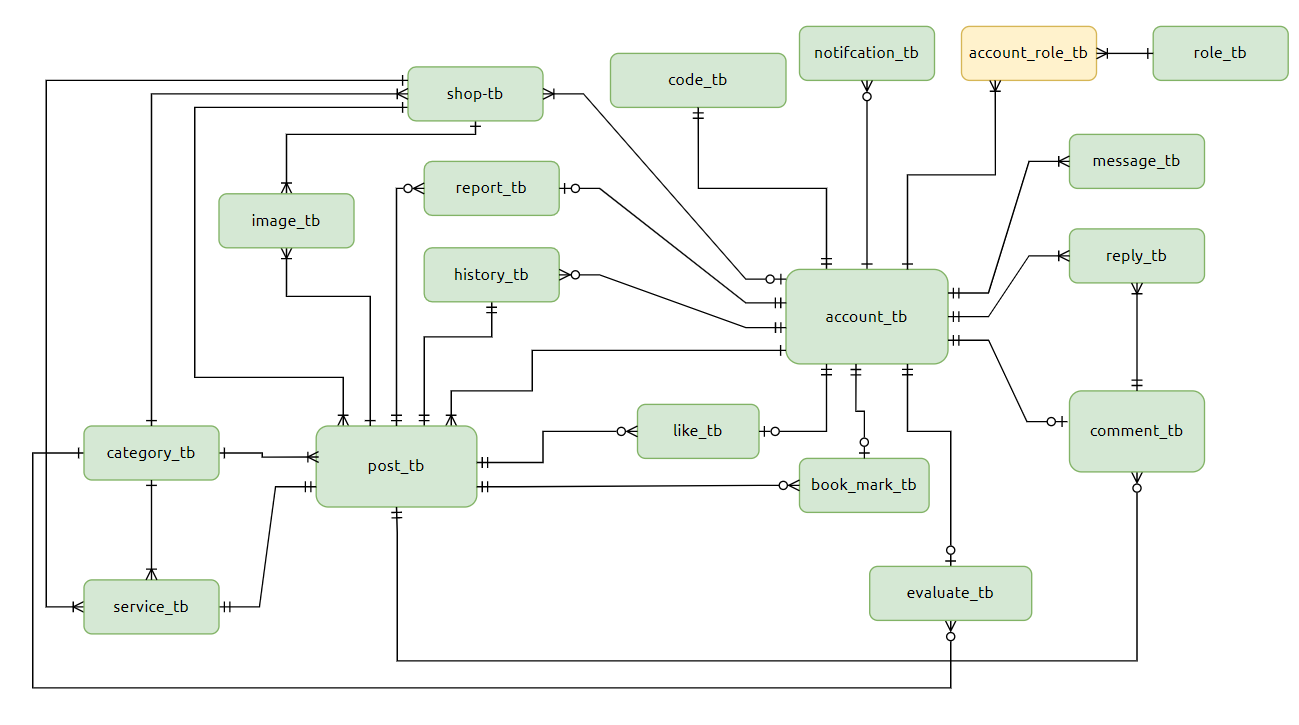


* 1. **Entity Relational Diagram**

An entity-relationship diagram (ERD) shows the relationship among entity sets. An entity set is a group of similar entities, and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in a database, so by showing relationships among tables and their attributes, ERD shows the complete logical structure of a database.

1. **Evaluation**

**Figure 7.** ERD Diagram



The **KbaeNak** website was developed to optimize the repair services seeking experience for users effortlessly. Along with a simple user interface so that users can utilize it without any difficulty, it is implemented with several additional functions to simplify the processing operation. However, it merely functions as a platform and offers no payment options.

* 1. **Project Strengths**

**KbaeNak** offers several strengths that enhance the user experience and functionality for both users and traders, including:

* Users can create posts to ask for help: Users can easily create posts requesting assistance with specific issues they face, facilitating a direct way to get support.
* Users can report shops that provide bad services: This feedback feature helps maintain service quality by allowing users to report bad experiences, which can alert others.
* Traders can create posts about their services and ask for help:Traders can post their offerings and even request assistance, making the platform dynamic and useful for both users and traders.
* Users and traders can chat directly: The integrated chat feature fosters real-time communication between users and traders, making it easier to resolve issues and provide quick solutions.
* Users and traders can comment on, react to, and bookmark the posts**:** This functionality enhances the social aspect of the platform, allowing users to engage with posts and keep track of useful information.
* Admin can ban users or traders: This administrative feature ensures that the platform maintains its integrity by preventing misuse of inappropriate behavior.
  1. **Project Weaknesses**

Despite its many strengths, **KbaeNak** also encounters some issues that need to be addressed in future updates:

* Users and traders could not share live locations: The lack of live location sharing makes it more difficult for users to find nearby services quickly, especially in urgent situations.
* Users and traders could not send images via chat: Currently, there’s no option to share images, which could be useful for users to show the issue they’re facing or for traders to send service-related information.
* Traders cannot involve with online payment: The absence of an online payment system means transactions must be handled offline, which could be less convenient for users and traders alike. Adding payment features would improve this aspect significantly.

1. **Conclusion**

**KbaeNak** is a unique and innovative solution to the problem of finding quick and reliable repair services. The platform successfully combines the power of digital technology with the needs of users and traders, providing a seamless experience for both parties. By allowing users to post service request, browse nearby traders, and interact directly with traders, **KbaeNak** streamlines the process of finding repair services more efficiently.

Despite its success, there are still areas for improvement. Adding features such as live location sharing and online payment options would enhance the platform’s functionality and make it even more convenient for users. Additionally, improving scalability to handle a larger number of users and transactions is an important consideration for future development.

Overall, **KbaeNak** has the potential to revolutionize the way users find and engage with repair services. It not only improves the user experience but also provides small traders with a valuable platform to grow their businesses and connect with customers. Future enhancements will continue to improve the platform’s effectiveness and usability.

**References**

1. Feature: [www.facebook.com](http://www.facebook.com)

www.booking.com

1. User Interface: www.instagram.com
2. System Architecture: www.geeksforgeeks.org