PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
Ministry of Higher Education and Scientific Research
Numidia Institute of Technology
Faculty of Computer Science

Software Engineering Project Documentation

Speciality : **Artificial Intelligence**

presented by : **Fares Abdi,Al Asmar Anas,Zobir Raouf,Benmati Ziad,Hechiche Nouha**

Hanini

CONTENTS

CHAPTER 1

INTRODUCTION

HANINI is a cutting-edge mobile application designed to simplify the process of discovering and offering a wide range of essential services. Whether looking for plumbing, housekeeping, electrical work, or any other home service, HANINI acts as a comprehensive bridge between customers and service providers. The app serves as a dynamic and easy-to-navigate platform that connects people with trusted service professionals, allowing users to access help quickly, securely, and with just a few taps on their mobile devices.

In today's fast-paced world, the need for on-demand services has grown exponentially. Customers seek convenience, reliability, and quality in the service providers they choose. On the other hand, service providers often struggle to reach a wide audience and manage their service requests effectively. HANINI addresses these challenges by offering a simple, intuitive platform that not only helps users find the right services but also gives service providers the tools they need to showcase their skills and manage their operations efficiently.

The key feature of HANINI lies in its user-friendly interface, which makes it accessible to people with varying levels of technical expertise. Whether a customer is looking for specific help or a service provider wants to expand their reach, HANINI ensures that both sides can easily navigate the app and connect with each other seamlessly. HANINI's goal is to make finding and offering home services as effortless as possible. By leveraging modern technologies and a user-centered design, the app facilitates the connection between service providers and customers, ensuring a seamless and trustworthy service experience. Whether a plumber, cleaner, or handyman is needed, HANINI brings users closer to professional help in a matter of seconds.

What sets HANINI apart from other e-service platforms is the combination of a simple, intuitive user interface with powerful backend systems that ensure smooth communication, secure transactions, and trust-building mechanisms. The app is designed to accommodate both seasoned professionals and users with no technical background, making it accessible to a broad audience. Moreover, the advanced AI verification and rating systems promote

safety and reliability, making HANINI a trustworthy platform for both service providers and customers.

The primary vision behind HANINI is to reduce the complexity involved in finding quality service providers and managing service requests. By offering a streamlined solution to both customers and service providers, HANINI aims to create a thriving community of professionals and users who can rely on the app for all their home service needs. The mission is to ensure that every customer gets the help they need, at the right time, and from a trusted professional, while empowering service providers to grow their businesses and maintain high standards of service quality. In the future, HANINI plans to expand the range of services offered, integrate more advanced AI and machine learning features for service matching, and further enhance the user experience. The platform is committed to continuous improvement based on user feedback and emerging technologies, ensuring it remains at the forefront of the e-services industry.

CHAPTER 2

OVERVIEW

2.1 Project Overview

The HANINI app is an innovative, dynamic E-Services platform that simplifies the process of connecting customers with service providers. It offers a wide range of services including plumbing, housekeeping, electrical work, repairs, and more, catering to a broad user base with diverse needs. The application is designed to address the challenges of finding reliable and timely assistance, all while making it easier for service providers to grow their businesses and reach potential clients.

At its core, HANINI provides a user-friendly, intuitive interface built with Flutter for both Android and iOS platforms, ensuring a seamless experience for users across different devices. The app's primary goal is to simplify the service request process, making it easier for customers to post requests and connect with the appropriate service provider in a matter of clicks.

2.1.1 Key Technologies Used

Frontend (Flutter)

Flutter is used to build the cross-platform user interface for HANINI, enabling smooth navigation, a visually appealing design, and fast performance. Flutter's rich widget library ensures that the app delivers a native-like experience on both Android and iOS.

Backend (Node.js))

The backend is powered by Node.js, which handles the application's business logic, authentication, and serves as the communication layer between the frontend and the database. Node.js ensures scalability and efficiency, allowing the app to handle large volumes of user requests and manage real-time data interactions

Database (Firebase)

Firebase is used for user authentication and database storage. Firebase Firestore stores user profiles, service requests, provider information, reviews, and transaction histories. It allows for real-time data synchronization and provides a scalable solution for storing and retrieving user data. Firebase Authentication handles secure sign-in and sign-up processes using email/password or third-party logins like Google .

2.1.2 Integration of AI, Machine Learning (ML), and Deep Learning

AI-Powered Identity Verification

HANINI uses identity card scanning technology to enhance security. Users are required to scan their identity card (e.g., national ID, passport, driver's license) during the registration or login process. The AI model then verifies the authenticity of the ID by extracting key features and comparing them with official records. This added layer of verification ensures that users and service providers are properly authenticated, offering a higher level of security for both parties. As the system is exposed to more identity documents, the deep learning models improve over time, making identity verification faster and more accurate.

Recommendation Systems (Collaborative Filtering + Machine Learning)

HANINI utilizes collaborative filtering in its recommendation system to suggest services to users based on the behavior and preferences of similar users. This means the system learns what other users with similar preferences have liked and recommends those services. Machine Learning (ML) models are used to predict the best service providers for a given customer request. By analyzing previous service interactions, preferences, location data, and other user attributes, the system can suggest service providers who are most likely to fulfill the request effectively.

CHAPTER 3

FEATURES AND FUNCTIONALITY

3.1 Key Features of HANINI

3.1.1 User Registration and Profiles

Users can create accounts, manage their profiles, and keep track of their service history.

Sign Up / Sign In

Users can sign up using their email/password or sign in using their Google account

AI-Based Identity Verification

AI models are used to verify the user's identity using scanned identity documents.

Service Listings

Service providers can list their services, and customers can search and browse the services easily.

Request Posting

Customers can post requests for specific services, which are matched with relevant service providers.

Favorites

Users can save favorite services and service providers for easy future access.

Settings

Users can modify their settings, including password changes, notification preferences, and privacy settings.

Logout

Users can log out securely from their accounts.

3.2 Page Logic Functionality

This section describes the functionality and logic behind each page of the HANINI app.

2.3.1 Login / Sign-Up Page

2.3.1.1 Login Flow

Sign-Up Flow

=>When the user first opens the app, they are presented with the option to sign up using their email or Google account. Upon selecting email, the user is asked to input their details (email, password, confirm password).
=>After validating the email format and password strength, the app calls Firebase Authentication to create a new account.

AI-powered identity verification

Users are prompted to scan their identity card (e.g., national ID or driver's license) using the phone's camera. The app uses TensorFlow AI models to verify the identity card by matching the document with the user's details entered during sign-up.

Sign-In Flow

=>Users can also choose to sign in using their registered email/password or use Google Sign-In. After a successful login, Firebase Authentication sends a token back to the app to keep the user session active.
=>If the sign-in is successful, the app fetches the user's profile data from Firebase Firestore (e.g., profile name, saved services, request history).

2.3.2 Home Page

Once logged in, the user is directed to the Home Page. This is the main screen where users browse various service categories such as Plumbing, Cleaning, Electrical, etc.

2.3.2.1 Features

Service Categories

The home page showcases various categories in a GridView layout. Each category contains a list of services offered by professionals.

AI-based Service Recommendations

The app uses machine learning algorithms to suggest personalized services based on the user's previous actions, behavior, and preferences.

2.3.2.2 How it work

1.User Action(1): The user navigates to the Home Page to browse recommended services.
2.Backend Interaction:
.The app sends a GET request to the backend to fetch personalized service recommendations.
.This request includes data such as the user's past interactions, ratings, and location (for location-based recommendations)
.The backend processes this data using Collaborative Filtering + Machine Learning algorithms to recommend services based on the user's preferences, ratings, and behaviors of similar users.
3.User Action(2): The user taps on a service card to view more details.
.A GET request is sent to the backend to retrieve the full details of the selected service.
.The app retrieves data such as description, ratings, reviews, availability, and provider information.

2.3.3 Search Page

1.User Action: The user types a search query (e.g., "plumber", "cleaning service") into the search bar.
2.Backend Interaction:
.The app sends a GET request to the backend to search for services that match the query.
.The backend searches through the Firestore database or Realtime Database for relevant services that match the query.
3.Database Interaction
.The backend searches the services collection in the database to retrieve relevant service records based on keywords or tags matching the search query.
.The results are sent back to the frontend, where they are displayed as Service Cards.

2.3.4 Favorites Page

1.User Action: The user navigates to the Favorites Page to view saved services.
2.Backend Interaction
.The app sends a GET request to the backend to retrieve the user's favorite services.
.The backend queries Firestore or Realtime Database to fetch the list of services the user has marked as favorites.
3.Database Interaction
.The app retrieves the list of favorite service IDs from the user's profile document stored in the Firestore database.
.The database returns the full details of each service, such as name, price, and provider info, to display on the Favorites Page.

2.3.5 Profile Page

1.User Action(1): The user navigates to the Favorites Page to view saved services.
2.Backend Interaction
.The app sends a GET request to retrieve the user's profile data from Firestore or Realtime Database.
.The app loads the name, profile picture, and account details into the profile section.
3.Database Interaction
.The app queries the user profile document in the database to fetch the user's details (name, email, and profile picture).
4.User Action(2): The user taps the Edit button to update their profile.
.The app prompts the user to edit their name and profile picture.
.A PUT request is sent to the backend to update the user profile data in Firestore.
.The database is updated with the new name and profile picture.
5.User Action(3): The user taps Become a Provider to offer services.
.The app prompts the user to fill out a service form.
.The user's service data (e.g., type, pricing, description) is submitted via a POST request to the backend.
.The backend stores this data in the services collection in Firestore.

2.3.6 Settings Logout

.The user navigates to the Settings Page to change their password or update notification preferences.
.Changes are submitted as PUT requests to the backend to update the user's data.
.Upon logout, a signOut() request is sent to Firebase Authentication to log the user out.

3.3 User Interface User Experience (UI/UX)

3.3.1 Sign-Up / Sign-In Flow

UI

1.Sign-Up Options

.Display two prominent buttons LogIn and sign-in with Google.

.Use icons to indicate the method (e.g., Google icon for Google sign-in).

.The background clean and minimal with a welcoming color scheme(Soft blue)

2.Input Fields

.Use clear input boxes for email/Username, password.

Ux

1.Sign-Up Process

.Provide feedback immediately (e.g., "Account created successfully", or "Error creating account").

.Upon successful sign-up, automatically redirect the user to the Home Page without extra steps.

2.Google Sign-In

.For the Google sign-in, users can simply click the Google button and are logged in without needing to fill out additional details.

3.Progressive Disclosure

.Initially, ask for only essential details (email, password) and progressively ask for more details later (like profile picture, service preferences, etc.).

3.3.2 Home Page

UI

1.Main Content

. vertical scroll of recommended services based on the user's preferences.

.Each service should be shown in service cards that contain(A service name,Rating stars,Location), and heart icon to mark favorit services.

2.Navigation Bar

.Place a bottom navigation bar with easy-to-recognize icons for Home, Search, Favorites,

and Profile. These should be centered and well-spaced.

3.Feedback

.Use subtle animations or loading indicators when content is loading, to reassure users that the app is working.

Ux

1.Personalization

.Provide AI-powered recommendations on the home page based on user's past behavior, ratings, and service requests.

2.Smooth Transitions

.When a user clicks on a service, smoothly transition to the Service Details Page with an animation, ensuring the user doesn't feel lost or disoriented.

3.3.3 Search Page

UI

1.Search Bar

.The search bar should be prominent at the top of the screen with a magnifying glass icon.
.Use placeholder text like "Search for services" to guide the user.

2.Search Results

.After the user searches, display results as cards with service name, price, ratings, and brief description.
.Each result should be tappable to navigate to the Service Details Page.

Ux

1.Instant Search Results

.Implement live search where results update as the user types, reducing time spent waiting for results.

2.Location-Based Search

.Automatically detect the user's location (with permission) to show relevant services nearby.

3.3.4 Favorites Page

UI

1.Favorites List

.Display all favorited services as cards with the service name, price, and a heart icon to indicate it's in the favorites.

.Use heart icons that turn color(red) when tapped to mark/unmark favorites.

Ux

1.Quick Access

.The user can tap on any favorited service to access its details immediately.

3.3.5 Profile Page

UI

1.Profile Header

.Display the user's profile picture at the top of the page, with an option to change it by clicking the Edit icon.

.Show the user's name, email, and user type (customer or service provider).

2.Edit Profile

.Use editable fields for the user's name and profile info. Place an edit icon next to the name and photo.

3.Become a Provider

.Place a call-to-action button saying "Become a Provider" that directs users to the page where they can list their services.

Ux

1.Editable Fields

.Provide easy-to-edit options for name, contact details, and profile picture.

.Use camera access for users to take a new profile picture or upload an image from their gallery.

2.Become a Service Provider

.When the user clicks "Become a Provider", show a simple, guided process to input service details (type, description, price).

3.3.6 Notifications Page

UI

1.Background Color

.The notification page should have a soft purple and white background to give it a calm and professional aesthetic. This subtle color scheme ensures the content stands out without being overwhelming.

2.Top Section

At the top of the page, display two main sections

1.Listings

.This section shows notifications related to service listings (e.g., a service has been posted, updated, or removed).

2.Reviews

.This section contains notifications for reviews left by customers on services the user has provided or booked.

. star Icon related to reviews.

3.Icons Under Listings and Reviews

1.Sent Icon (Outgoing Notifications)

.Below the Listings and Reviews sections, add a sent icon

.This icon represents notifications that the user has sent or is involved with

2.(Received Icon (Incoming Notifications))

.Underneath the sent icon, place a received icon

.This icon represents notifications the user has received, such as responses from service providers or new reviews from customers.

4.Notification Items

.Each notification should be displayed as a list item.

Ux

1.Real-Time Updates

.Ensure real-time notifications are received immediately and dynamically loaded in the list as new updates are available.

2.Organized Listing

.Separate notifications into Categories (Listings and Reviews) for easy navigation. You can use collapsible headers to minimize space when a section has no notifications.

3.Clear Feedback

.If the user has no notifications, display a message like "You have no new notifications" with a soft purple icon, such as a bell with a line through it to indicate no new alerts.

3.3.7 Language Preferences Settings

UI

1.Language Selector

.Add a language toggle icon in the top navigation bar to allow users to switch between languages.

.When the user taps it, a dropdown menu with a list of available languages should appear.

Ux

1.Seamless Transition

.Upon selecting a new language, automatically update all UI text across the app to the selected language.

.Ensure all translated content is contextually accurate and fits within the UI design (e.g., button text, service descriptions).

2.User Preferences

.Allow users to adjust other settings such as notifications, password, and privacy preferences in the Settings Page.

CHAPTER 4

CONCLUSION

The HANINI app is designed to be a dynamic E-Services Platform that seamlessly connects service providers with customers for various physical services such as plumbing, housekeeping, and more. With a strong emphasis on user experience (UX), interface design (UI), and efficient backend architecture, the HANINI app aims to provide users with a trustworthy, easy-to-navigate, and feature-rich platform. The primary objective is to create an efficient, user-friendly platform that bridges the gap between service providers and customers, enhancing both parties' experience. HANINI is focused on providing a safe, personalized, and intuitive experience for all users, with features such as AI-based identity verification and personalized service recommendations using machine learning. The core features of the app include user registration and profiles, where users can sign up via email or Google account and become verified service providers. Service providers can upload detailed service listings, while customers can browse and submit service requests. The app also includes an Admin Dashboard for administrators to manage accounts and monitor service requests. AI-based identity verification is implemented to ensure secure user logins, and a recommendation system using collaborative filtering helps to offer personalized service suggestions based on browsing history, ratings, and preferences. The UI/UX design principles are centered around simplicity, cleanliness, and intuitiveness. The color schemes, such as soft purple and white, create a professional and welcoming aesthetic. Navigation is optimized for easy access to key pages like Home, Search, Favorites, and Profile. The notification system is designed with interactive icons, real-time updates, and customizable settings, making it easy for users to stay informed. The recommendation engine personalizes each user's experience by suggesting relevant services based on their history and behavior. The app's page logic and functionality include an easy sign-up/sign-in process, with options for email or Google sign-in, and AI-based identity verification for security. The home page greets users with personalized service recommendations, and the bottom navigation bar provides access to Search, Favorites, and Profile pages. The search page features a search bar, filters, and real-time results, while

the favorites page allows users to organize saved services. The profile page enables users to manage personal details, and those interested in becoming service providers can click on the "Become a Provider" button. The admin dashboard enables administrators to efficiently manage the platform. The app's backend is built with Node.js for scalability, while Firebase is used for data storage and real-time synchronization, ensuring secure and reliable management of user profiles, service listings, and other data. Real-time updates ensure that notifications, service requests, and other dynamic data are synchronized across devices instantly. The HANINI app represents a robust and forward-thinking solution for users seeking a hassle-free experience in the service industry. Its clean UI, personalized UX, and scalable backend ensure that both customers and service providers can enjoy a seamless, efficient, and secure experience. With AI-powered identity verification, collaborative filtering recommendations, and real-time updates, HANINI is positioned to enhance convenience and user satisfaction. As the app continues to grow, it will expand its features to meet the needs of an even broader user base, always prioritizing simplicity, personalization, and user security. This concludes the HANINI app documentation, providing a comprehensive overview of its features, technical specifications, design considerations, and functionality. The app has been meticulously designed to offer a streamlined and user-friendly experience for all users, ensuring that both customers and service providers can connect efficiently and securely.