APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 6 No. 2 (2025) 1119-1137
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Traserve: Tradesman Service Application

Muhammad Ashraff Azhan[1], Ruhaya Ab Aziz[1]*

[1] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: ruhaya@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2025.06.02.060

Article Info

Received: 15 June 2025
Accepted: 20 November 2025
Available online: 30 November 2025

Keywords

Tradesman management, Order service, Mobile Application, Electrical and Plumbing

Abstract

Traserve: Tradesman Service is a mobile app developed with Amirul Tech Service to streamline the booking of electrical and plumbing services in Johor Bahru. Replacing the current manual WhatsApp-based system, Traserve addresses challenges such as inefficient order handling, overbooking, and lack of payment and tracking systems. Built using the Flutter framework with Firebase backend, the app features secure user authentication, real-time messaging, and integrated payments. Developed using the Prototyping Model, the process includes requirement gathering, design, prototyping, user evaluation, refinement, and deployment. Traserve enhances service efficiency and customer satisfaction by offering a digital platform for booking, tracking, and managing maintenance services. It supports Amirul Tech Service's operations by improving order management, reducing scheduling conflicts, manage order history report and boosting operational growth.

1. Introduction

The tradesman service sector has experienced significant growth due to increasing demand for professional home maintenance services. As consumer lifestyles become busier, there is a growing need for efficient, technology-driven solutions that can streamline service delivery while maintaining quality. Traditional service providers, which often rely on manual processes for bookings and payments, struggle to meet modern customer expectations for convenience and digital accessibility. Amirul Tech Service, established on 19th November 2022 in Johor Bahru, is a freelance technician and maintenance service specializing in electrical repairs for appliances like air-conditioners, fridges, and fans, with occasional plumbing work. The team currently takes service requests via WhatsApp or calls.

The current manual booking system of Amirul Tech Service, which relies on WhatsApp and calls, leads to inefficiencies such as overbooking, scheduling conflicts, and a time-consuming cancellation process. It lacks a centralized system and finds it difficult to handle several orders at once, which has a detrimental effect on customer satisfaction and service effectiveness [1]. Customers also struggle to locate trustworthy technicians, make appointments, and securely pay often involving multiple platforms, which complicates and inconveniently increases the process. Another key issue is the difficulty in tracking past orders, as technician must scroll down a

chat histories and record bookings manually, making reporting and order management difficult. This research aims to accomplish three primary objectives in the development of the Traserve. The first objective is to design a tradesman service application based on an object-oriented approach. The second objective involves the actual development of the application, implementing all planned features and functionalities to develop a mobile based tradesman service management in Johor Bahru. The third objective is to test the system, encompasses thorough testing and evaluation of the developed system, ensuring all components work as intended and meet the needs of both crew and customers. This article consists of six sections, which are introduction of the project, literature review, methodology, system development analysis and design, implementation and results and discussion of the project, conclusions of the system that has been developed.

2. Literature Review

This section discusses the domain background and as-is model of Amirul Tech Service Booking System, case study, and system's comparison with three existing system. The purpose of this literacy study is to provide a clear picture of some of the questions to be studied through various methods to find out the requirements that need to be met. In achieving analysis and meeting the requirement needed, reading methods, evaluation, and analysis are mainly related to this service booking system topic needed in the development process of this project.

2.1 Domain Background

Amirul Tech Service, a reputable service provider of tradesman services based in Johor Bahru. With a dedicated team of skilled tradesman, the company specializes in the repair of electrical appliances, while also offering plumbing services as needed. Over the past two years, Amirul Tech Service has built a solid reputation for reliability, quality workmanship, and customer satisfaction in the local service industry. Despite its active operations, Amirul Tech Service faces inefficiencies due to the absence of a systematic way to manage service orders and financial records.

Currently, the operation of tradesman crews at Amirul Tech Service is managed using traditional methods. The company uses the WhatsApp application to handle service orders, interact with customers, and store payment receipts. WhatsApp Business is used to create a service list, including price ranges and descriptions. Customers must browse the available services and message the admin to check whether a service is available. To place an order, customers are required to provide details such as the address, date, and time. All of this information is stored in the WhatsApp chat history. The current system is no longer compatible with the growing scale of Amirul Tech Service's operations, especially around the Johor Bahru area. The admin is responsible for both communicating with customers and assigning crews via WhatsApp, which slows down the process. Tracking chats and scheduling can become problematic, and scheduling conflicts may occur such as assigning the same crew to multiple jobs on the same day. Additionally, payment records are kept manually in a physical book, which poses several disadvantages. Financial issues can arise if the book is lost or misplaced.

The current service process involves the customer initiating a chat on WhatsApp, browsing the service catalog, and submitting an order form with the required details. The customer must then wait for the admin to respond and confirm whether the order is accepted or rejected. After that, the customer pays a deposit and waits for the assigned crew to arrive at the location. Once the service is completed, the remaining payment can be made via QR code (sent by the admin through WhatsApp) or in cash (to the crew). The admin records the completed payments manually in a book.

2.2 Booking Service Management System

A House Service Management System manages its service catalog through a centralized module where administrators or service providers can add, update, categorize, and remove household services such as plumbing, electrical work, and cleaning, ensuring that only accurate and available services are visible to users [13]. Each service entry typically includes essential details like name, description, duration, and location coverage, enabling users to make informed choices quickly [7]. The system also allows service status control such as active or inactive, technician assignment, and performance tracking, all managed through an admin or provider dashboard [6]. This organized structure improves operational efficiency, keeps the platform up to date, and enhances user satisfaction by offering a clear, searchable, and reliable list of services [14]. Thus, a systematics service management that has been review for managing the service list for customer play crucial part as references for building the Traserve system that can be utilized by users and adapts to Fourth Industrial Revolution.

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 1119-1137

2.3 Method or Technology

The development of the Traserve: Tradesman Service Application utilizes modern development frameworks and technologies to ensure reliability and scalability. The application employs Flutter framework for cross-platform mobile development, enabling the creation of high-quality native applications for both iOS and Android platforms from a single codebase. Firebase serves as the backend infrastructure, providing real- time data synchronization and secure user authentication. The application employs Flutter framework [2] for cross-platform mobile development, enabling the creation of high-quality native applications for both iOS and Android platforms from a single codebase. Using dart language make the system more effective as Dart is the single codebase language. Firebase [3] serves as the backend infrastructure, providing real-time data synchronization and secure user authentication. The system also integrates cloud platforms for scalable hosting capabilities, ensuring reliable performance even during peak usage periods.

For payment processing, the application incorporates secure payment gateways that enable safe and efficient transaction handling. The implementation of chatbot makes the system more complex as both crew and customer have real-time communication through the application. The system's database has been designed and implemented using Firebase to ensure real-time data synchronization and reliability [4]. These technological choices work together to create a robust and user-friendly application that meets modern standards for mobile service applications [5]. Thus, Flutter framework with Firebase as server-side for storing data has been chosen to build a mobile-based system named Traserve: Tradesman Service Application.

2.4 System Comparison

The proposed system is designed to include helpful features from existing tradesman or house service management systems while also addressing specific needs of the Traserve. Each of the systems has its own strengths, like different ways to log in, how they handle cataloguing, and easy search functions. By adding features like checking book availability and cataloguing from these systems, the proposed system aims to make the library experience better, improve efficiency, and meet the library's needs. The existing system that has been choose is Zonar, TukangGo, and Recommend.my which are the platform for anyone who can register, while Traserve is the internal platform for registered crew only in Amirul Tech Service that surely will be more focus the career growth of crews and organization. Table 1 shows the comparison between the existing systems and the Traserve system.

Table 1 *Comparison of existing systems*

Features/System	Zonar [8]	TukangGo [9]	Recommend. my [10]	Traserve
User Authentication	Yes	Yes	Yes	Yes
Manage Account	Yes	Yes	Yes	Yes
Manage Service	Yes	Yes	Yes	Yes
Manage Order	Yes	Yes	Yes	Yes
Manage Payment	No	Yes	Yes	Yes
Manage Chat	Yes	No	No	Yes

Even though these three systems have different intended users and features, taken as a whole, they provide insightful information about the state of house service booking systems today. Considering their advantages and disadvantages offers a chance to create more inventive systems that can adapt to changing consumer needs and preferences.

3. Methodology

Traserve: Tradesman Service Application use a prototyping software process model to develop the application system. The prototyping model is a systems development method in which a prototype is built, tested, and then reworked as necessary until an acceptable outcome is achieved from which the system or product can be developed [9]. This approach enables close collaboration between the development team and stakeholders, ensuring that the final product aligns with user needs and objectives. The prototyping model in this project will include six phases, which are requirement gathering and analysis, design, prototype building, user evaluation, prototype refinement, implementation product and maintain phase. Fig. 1 will show about Prototyping Model. All activities involved in the prototyping model was shown Table 2.

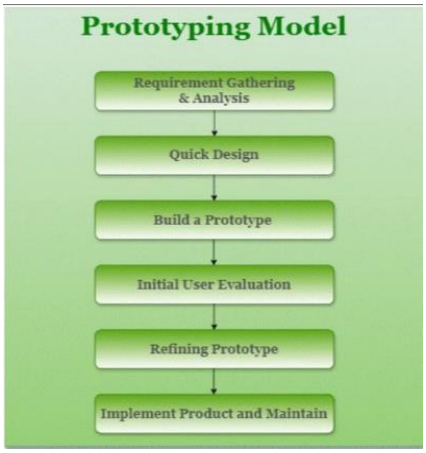


Fig. 1 *Prototyping model [12]*

Table 2 *Software development activities and tasks*

Phase	Task	Output
Requirement Gathering and Analysis	Determining the problem and project timeline. Analyze the requirements, interviews, and observations.	Project proposal Project requirements Project scope
Design	Designing the system architecture, user interface, database schema, and mobile structure based on the requirements.	Use case diagram Activity diagram Sequence diagram Database design User interface design System architecture
Building Prototype	Develop the prototype according to the gathered requirements and design	Prototype Backend infrastructures
User Evaluation	Interact with the prototype to evaluate its functionality, usability, and overall user experience.	User feedback
Prototype Refinement	Review the feedback and make an update for prototype according to the feedback.	Updated prototype
Implementation Product and Maintain	Develop the final product based on all user's requirements, feedback, and prototype. Deploy the finalized system in its operational environment. Test the validity of system	Project full report Production-ready system prototype Testing Result User Acceptance Testing

4. Analysis and Design

This topic explains the analysis and design of the proposed system. It includes functional requirement, non-functional requirement, user requirement, system architecture, database design, and user interface design. This section also describes the Unified Modelling Language (UML), such as use case, activity, sequence, and class diagram. The outcomes of the analysis and design of this application may ensure that its functional process is more easily understood and that it can satisfy user requirements need.

4.1 Requirement Analysis

The process of identifying the needs that a constructed system must meet or the results that users anticipate from the suggested system is known as requirement analysis. System requirements consist of user requirements, functional and non-functional requirements, and system prerequisites. Table 3 shows the functional requirements while Table 4 shows non-functional requirements for Traserve: Tradesman Service Application.

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 1119-1137

Table 3 *Functional Requirement*

Module	Functionality
User Authentication	- Allows users to register for the system. - Registered users can log in to access the mobile app.
Manage Account	- Allows users to add, and update their personal information. - Allows users to delete their own account. - Allows users to view order history for tracking.
Manage Service	- Allows crew to create, update, and delete the service details information. - Allows customer to browse, view, and select service available.
Manage Order	- Allows customer to create, update, and cancel order. - Allows crew to accept, reject, update service price, and manage order workflows. - Allows admin to track order process status.
Manage Payment	- Allows admin to setup payment methods for customer. - Allows customer to make a payment for admin to verify. - Allows admin to verify the payment record to update order process. - Allows admin to pay salary to crew.
Manage Chat	- Allows users to have communication through messaging function

Table 4 *Non-functional Requirement*

No	Requirement	Description
1	Performance	- The system should always be usable. - The system should allow users to place orders in a short period of time. - The system should be able to have real-time system process.
2	Operational	- The loading time required for a mobile app is no more than 1 minutes. - The system can be updated and maintained with easy.
3	Security	- The system should be able to handle user authentication. - The Firebase rules should be strictly.
4	Usable	- The user interface should be responsive and user-friendly. - The system should be easy to understand. - The system should be able to work on any device.

4.2 Use Case Diagram

The use case diagram was created to illustrate the overall functionality and components of the application. The use case diagram is a UML diagram that shows the relationship between the actors, use cases and the system [11]. There are three actors in the system which are Customer, Crew, and Admin. There are 6 main functionalities in this system which are Manage User Authentication, Manage Account, Manage Service, Manage Order, Manage Payment and Manage Chat. The main functionalities and overall activity in this system as shown as use case in the Fig. 2.

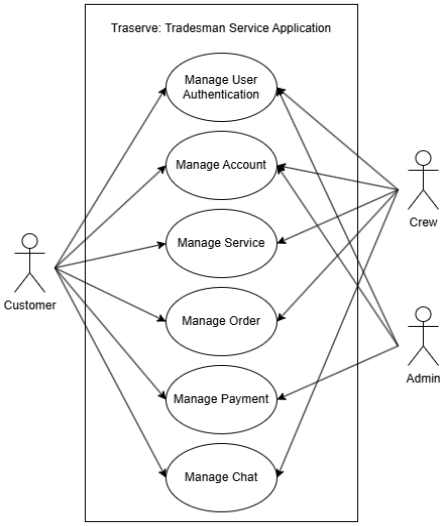


Fig. 2 *Use Case Diagram of Traserve: Tradesman Service Application*

4.3 System Architecture

Traserve: Tradesman Service Application uses a centralized unstructured Cloud Firestore Database to manage data and supports two main user roles: customers and crew. Each user accesses role-specific interfaces. Customers register with a username, password, and verified email, then log in to browse and book services via category or search. Crew members register with assigned credentials and access a dedicated dashboard. They manage service details, order statuses, and profiles. Orders placed by customers are confirmed by crew, with both parties able to communicate through an in-app chat. Crew sets service pricing, while admins verify payment methods. All users can manage their profile information, and all system interactions are handled through the UI, backed by the Firestore database. Traserve: Tradesman Service Application's system architecture is as shown in Fig. 3.

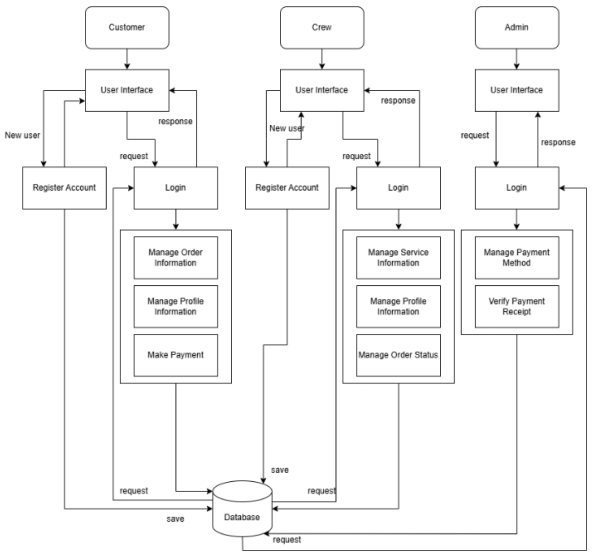


Fig. 3 *System Architecture*

4.4 Class Diagram

UML Class Diagram is a visual notation for the design and representation of object-oriented systems. Such a diagram represents the structure of the system by showing the classes, attributes, functions, and relationships between its objects. The Traserve: Tradesman Service Application's class diagram is as shown in Fig. 4.

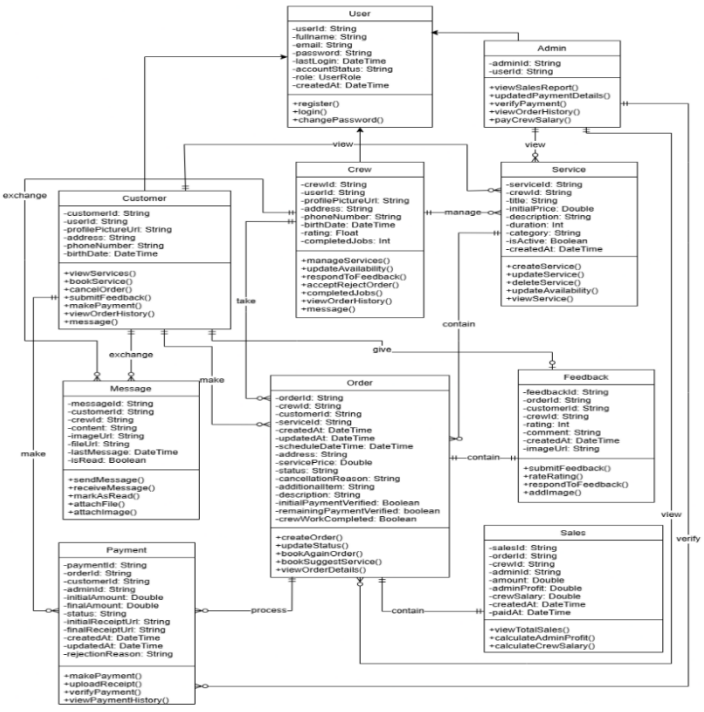


Fig. 4 *Class Diagram*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 1119-1137

4.5 Database Design

A Schema Table and data dictionary in table 5 are derived from the database design, which is based on the class diagram created during the system analysis phase. These tables represent the entities and their relationships, reflecting the core functionalities of the application.

Table 5 *Schema Table of Traserve*

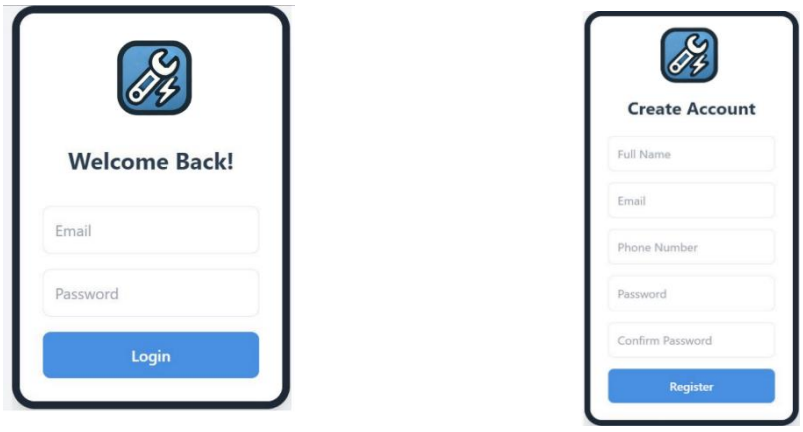
Collection	Attributes
User	userId, fullname, email, password, role, lastLogin, accountStatus, createdAt
Customer	customerId, profilePictureUrl, address, phoneNumber, birthDate, userId
Crew	crewId, profilePictureUrl, address, phoneNumber, birthDate, rating, completedJobs, userId
Admin	userId, adminId
Service	serviceId, title, initialPrice, description, duration, category, isActive, createdAt, crewId
Order	orderId, scheduleDateTime, address, servicePrice, status, cancellationReason, additionalItem, description, initialPaymentVerified, remainingPaymentVerified, crewWorkCompleted, createdAt, updatedAt, crewId, customerId, serviceId
Payment	paymentId, initialAmount, finalAmount, status, initialReceiptUrl, finalReceiptUrl, createdAt, updatedAt, rejectionReason, orderId, customerId, adminId
Message	messageId, content, imageUrl, fileUrl, lastMessage, isRead, customerId, crewId
Feedback	feedbackId, rating, comment, createdAt, imageUrl, orderId, customerId, crewId
Sales	salesId, amount, adminProfit, cewSalary, createdAt, paidAt, orderId, crewId, adminId

4.6 Interface Design

The design of the system interface is one of the important and necessary things for every system development that is made. It aims to give an idea of how the interface of the system is to be developed. The interface design focuses on creating an intuitive and user-friendly experience, incorporating modern design principles and responsive layouts. The following are the interfaces that have been designed based on each module.

4.6.1 User Authentication

The user interface of login as shown in fig. 5 and user interface of (a) signin and (b) signup that need to fill in Email and Password. The information will be validated. If the information is correct, it will be redirected to the homepage. If the information is not correct, the error message will pop up. If a user needs to register, they need to click the sign-up button.



(a) **(b)**

Fig. 5 *User Authentication Sign in (a) Sign in; (b) Sign Up*

4.6.2 Manage Account

User can manage personal information, update account information, address and phone number on account in fig. 6.

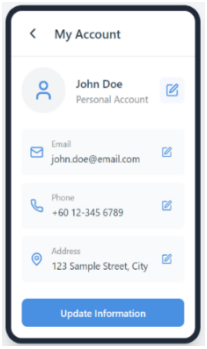


Fig. 6 *Manage Account*

4.6.3 Manage Service

Manage Interface will be in role crew side because the crew only can manage CRUD of services. The crew will fill in the details of the service and click the button create to add it. Then, its display the list of crew service. The user interface is shown in fig. 7.

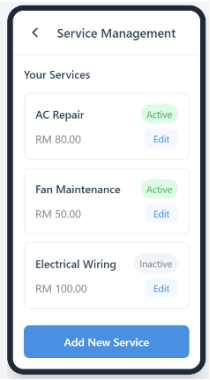


Fig. 7 Manage Service

4.6.4 Manage Order

Manage Interface will be in role customer because customers are the only user that can book the order. Customers will add the service and fill the details. They need to click on the join button. If successful, it will redirect to the order page where customers can track order process. The user interface is shown in fig. 8.

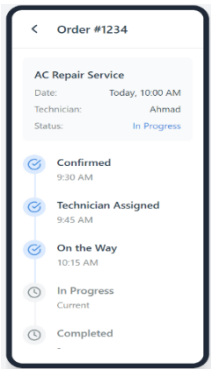


Fig. 8 *Manage Order*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 1119-1137

4.6.5 Manage Payment

After the service done, customer need to make a payment. There are two payment phases, first in initial payment, customer need to pay after confirm the order. Second is after the service fully done. They will make the payment to the admin account. The details of the bank will be displayed in the interface. Then click upload to continue. The user interface is shown in fig. 9.

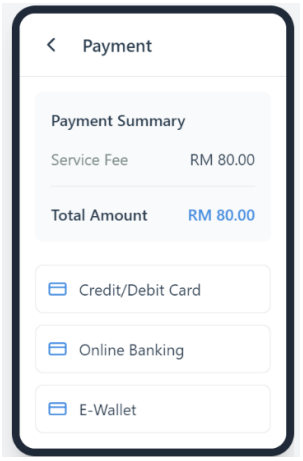


Fig. 9 *Manage Payment*

4.6.6 Manage Chat

Manage chat is one of the modules for customers and crews can interact with each other for any inquiries or deal. The user interface of the QR code is shown in fig. 10.

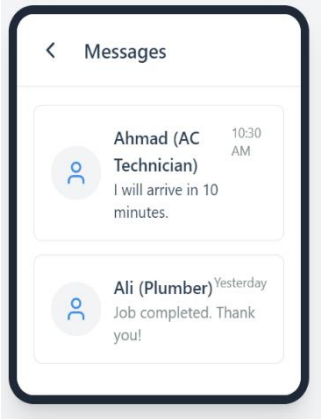


Fig. 10 *Manage Chat*

5. Result and Discussion

This section focuses on system implementation and testing for the mobile-based Traserve: Tradesman Service Application. It covers the implementation of User Authentication, Manage Account, Manage Service, Manage Order, Manage Payment, Manage Chat module functionalities. Additionally, it discusses the comprehensive functional testing conducted to the system's performance. The system's database has been designed and implemented using Firebase to ensure real-time data synchronization and reliability.

5.1 Implementation

This section describes the development of functional modules in a system. Program code is provided to aid clarification.

5.1.1 User Authentication Interface

Fig. 11 *User Interface Sign In*

```
Future<void> signin({
  required String email,
  required String password,
  required BuildContext context
}) async {
  try {
    final userCredential = await _auth.signInWithEmailAndPassword(
      email: email.trim(),
      password: password.trim()
    );
    final user = userCredential.user!;

    // Get user role from Firestore
    final userDoc = await _firestore.collection('users').doc(user.uid).get();
    final userRole = userDoc.get('role');

    // Check email verification, but bypass for admin
    if (!user.emailVerified && userRole != 'admin') {
      await _auth.signOut();
      if (context.mounted) {
        ScaffoldMessenger.of(context).showSnackBar(
          SnackBar(
            content: const Text('Please verify your email first'),
            action: SnackBarAction(
              label: 'Resend',
              onPressed: () => sendVerificationEmail(user, context),
            ), // SnackBarAction
          ), // SnackBar
        );
      }
      return;
    }
```

Fig. 12 *Source code Sign In*

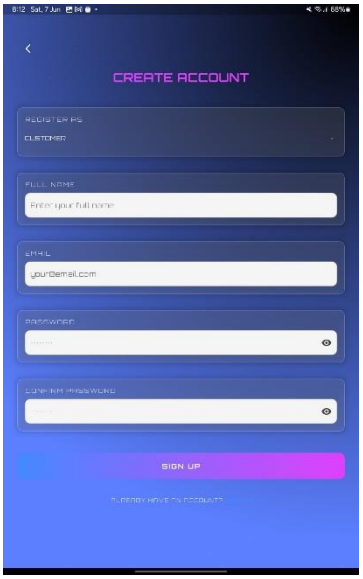


Fig. 13 *User Interface Sign Up*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 1119-1137

```
// Create user in Firebase Auth
UserCredential userCredential = await _auth.createUserWithEmailAndPassword(
  email: email.trim(),
  password: password.trim()
);

final user = userCredential.user!;
final userData = {
  'uid': user.uid,
  'email': email.trim(),
  'fullName': fullName.trim(),
  'role': role.toString().split('.').last, // Stores 'customer' or 'crew'
  'emailVerified': false,
  'createdAt': FieldValue.serverTimestamp(),
  'updatedAt': FieldValue.serverTimestamp(),
  'lastLogin': null,
};

// Create user document in Firestore
await _firestore.collection('users').doc(user.uid).set(userData);

// Send verification email
await sendVerificationEmail(user, context);
```

Fig. 14 *Source code Sign Up*

Fig. 11 shows the user interface of the Sign In. Fig. 12 shows the source code for Sign In where have the verification of user. The verification will use Firebase Authentication method to connect with the server side. The successful verification will redirect to home page. The failed verification will return to login again with error pop up. Fig. 13 shows the user interface of the Sign Up. Fig. 14 shows the source code for Sign Up. The Sign Up start with checking the email user if it already exists or not. If the user not exist, then it will proceed to insert the data into Firestore. If the user exists, it will redirect login page with user exist message.

5.1.2 Manage Account Interface

Fig. 15 *User Interface Manage Account*

```
late TextEditingController _fullNameController;
late TextEditingController _phoneController;
late TextEditingController _addressController;
DateTime? _selectedDate;
File? _profileImageFile;
String? _profileImageUrl;
bool _isEditing = false;
bool _isLoading = false;
bool _isImageUploading = false;

@override
void initState() {
  super.initState();
  _fullNameController = TextEditingController(text: widget.userData['fullName'] ?? '');
  _phoneController = TextEditingController(text: widget.userData['phoneNumber'] ?? '');
  _addressController = TextEditingController(text: widget.userData['address'] ?? '');
  _profileImageUrl = widget.userData['profilePictureUrl'];
  if (widget.userData['birthDate'] != null) {
    _selectedDate = (widget.userData['birthDate'] as Timestamp).toDate();
  }
}
```

Fig. 16 *Source Code Manage Personal Information*

Fig. 15 shows the user interface of Manage Account. Fig. 16 shows the source code of Manage Account. The user needs to fill the input to update the information. After fill and check the input, click the save button to save the information. It will execute the Collection update query through the Firestore method to update at the database.

5.1.3 Manage Service Interface

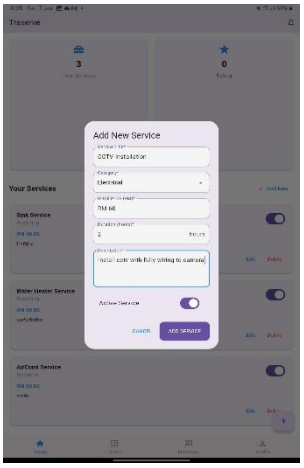


Fig. 17 *User Interface Manage Service*

```dart
class _ServiceDialogState extends State<ServiceDialog> {
  final _formKey = GlobalKey<FormState>();
  late TextEditingController _titleController;
  late TextEditingController _priceController;
  late TextEditingController _descriptionController;
  late TextEditingController _durationController;
  late String _selectedCategory;
  late bool _isActive;

  final List<String> _categories = ['Electrical', 'Plumbing'];

  @override
  void initState() {
    super.initState();
    _titleController = TextEditingController(text: widget.service?.title ?? '');
    _priceController = TextEditingController(
      text: widget.service?.initialPrice.toString() ?? '',
    );
    _descriptionController = TextEditingController(
      text: widget.service?.description ?? '',
    );
    _durationController = TextEditingController(
      text: widget.service?.duration.toString() ?? '1', // Default 1 hour
    );
    _selectedCategory = widget.service?.category ?? _categories[0];
    _isActive = widget.service?.isActive ?? true;
  }
```

Fig. 18 *Source Code Manage Service*

Fig. 17 shows the user interface of Manage Service. Fig. 18 shows the source code of the Manage Service. The crew must fill the form of service detail to create the service. The service can be view to the customer. The service details will be store in services collection in Firestore.

5.1.4 Manage Order Interface

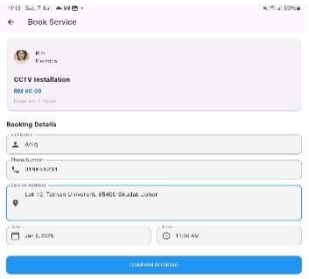


Fig. 19 *User Interface Manage Order*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 1119-1137

```dart
class _CustomerBookingPageState extends State<CustomerBookingPage> {
  final _formKey = GlobalKey<FormState>();
  final _fullNameController = TextEditingController();
  final _phoneController = TextEditingController();
  final _addressController = TextEditingController();
  DateTime? _selectedDate;
  TimeOfDay? _selectedTime;
  bool _isLoading = false;
  late Future<app_user.User?> _crewFuture;

  @override
  void initState() {
    super.initState();
    _crewFuture = _getCrewInfo();
  }

  Future<app_user.User?> _getCrewInfo() async {
    try {
      final doc = await FirebaseFirestore.instance
          .collection('users')
          .doc(widget.crewId)
          .get();
      return doc.exists ? app_user.User.fromMap(doc.data()!, widget.crewId) : null;
    } catch (e) {
      return null;
    }
  }
```

Fig. 20 *Source Code Manage Order*

Fig. 19 shows the user interface of Manage Order. Fig. 20 shows the source code of Manage Order. The customer can create the order by clicking the service card. Customer must fill every detail of order and it will be store as bookings in Firestore's collection. Customer can view, update, and cancel the order.

5.1.5 Manage Payment Interface

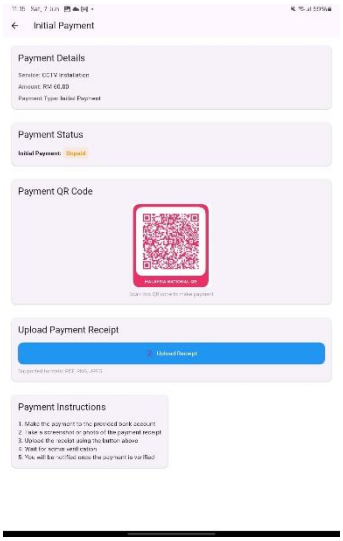


Fig. 21 *User Interface Manage Payment*

```dart
class _CustomerPaymentPageState extends State<CustomerPaymentPage> {
  final PaymentService _paymentService = PaymentService();
  final OrderService _orderService = OrderService();
  final ImagePicker _picker = ImagePicker();
  File? _receiptFile;
  bool _isLoading = true;
  String? _qrUrl;
  Payment? _payment;
  order.Order? _order;
  bool _isUploading = false;
  String? _uploadedReceiptUrl;

  @override
  void initState() {
    super.initState();
    _loadQrCode();
    _loadPayment();
    _loadOrder();
  }
```

Fig. 22 *Source Code Manage Payment*

Fig. 21 shows the user interface of Manage Payment. Fig. 22 shows the source code of Manage Payment. Customer needs to pay the bill through the app via admin payment method provider. There's two phase of payment which is initial and remaining that both need admin's verification.

5.1.6 Manage Chat Interface

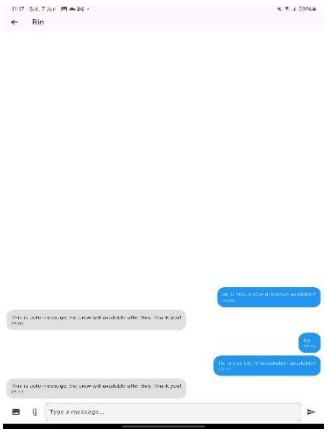


Fig. 23 *User Interface Manage Chat*

```
class _CustomerChatScreenState extends State<CustomerChatScreen> {
  final MessageService _messageService = MessageService();
  final TextEditingController _messageController = TextEditingController();
  final currentUser = FirebaseAuth.instance.currentUser;
  File? _imageFile;
  File? _file;
  String? _fileName;
  final ImagePicker _picker = ImagePicker();

  @override
  void initState() {
    super.initState();
    if (currentUser != null) {
      _messageService.markMessagesAsRead(currentUser!.uid, widget.crewId);
      _maybeSendAutoMessage();
    }
  }

  Future<void> _maybeSendAutoMessage() async {
    if (widget.serviceName == null) return;
    // Customer sends the first message
    await _messageService.sendMessage(
      senderId: currentUser!.uid,
      receiverId: widget.crewId,
      content: 'Hi, is this ${widget.serviceName} available?',
    );
    // Crew/system sends the auto-reply
    await _messageService.sendMessage(
      senderId: widget.crewId,
      receiverId: currentUser!.uid,
      content: 'This is auto message, the crew will available after this, Thank you!',
    );
  }
```

Fig. 24 *Source Code Manage Chat*

Fig. 23 shows the user interface of Manage Chat. The customer can start the chat by clicking the chat button in crew service list. Fig. 24 shows the source code of the Manage Chat where the system will store the latest interaction in Firestore's collection.

5.2 Functional Testing

This section focuses on verifying that the Traserve's features work according to the user requirements. It checks each functional module of the application by providing appropriate inputs and examining the result, ensuring that the app performs its intended tasks correctly. Table 6 shows the testing result of Traserve.

Table 6 *Testing Results*

Module	Description	Expected Result	Actual Results	Pass/Fail
User Authentication	To check whether customer, crew and can register and verify and account	Customer and crew are able create account	Customer and crew are successfully creating account	Pass

Table 6 *Continued*

	To check whether customer, crew, and admin can log in to app	Customer, crew, and admin able to login to app	Customer, crew, and admin successfully login	Pass
Manage Account	To check whether customer and crew can edit their profile	Customer and crew able to edit their profile	Customer and crew successfully edit their profile	Pass
Manage Service	To check whether crew can add, update, delete service	Crew able to add, update, delete the event	Crew successfully add, update, delete the service	Pass
Manage Order	To check whether customer can create the order	Customer able to create the order	Customer successfully create the order	Pass
Manage Payment	To check whether customer can submit the payment receipt	Customer able to submit the payment receipt	Customer successfully submit the payment receipt	Pass
Manage Chat	To check whether customer and crew can send and receive message	Customer and crew able to send and receive message	Customer and crew successfully send and receive message	Pass

Every module was tested using the proper inputs, and the expected and actual results were contrasted. According to the findings, every essential feature such as user authentication, profile management, service management in-app messaging, order placement, payment submission, in-app messaging that operated flawlessly. Every tested module passed, demonstrating that the application successfully satisfies its functional requirements.

5.3 User Acceptance Testing

User Acceptance Testing (UAT) is the phase of the testing process where real users test the software to ensure it meets user requirements. UAT evaluates whether the system is ready for deployment by checking its usability, functionality, and overall performance in real-world scenarios. Traserve App has been tests by ten users related which is six customers, three crews, and an admin.

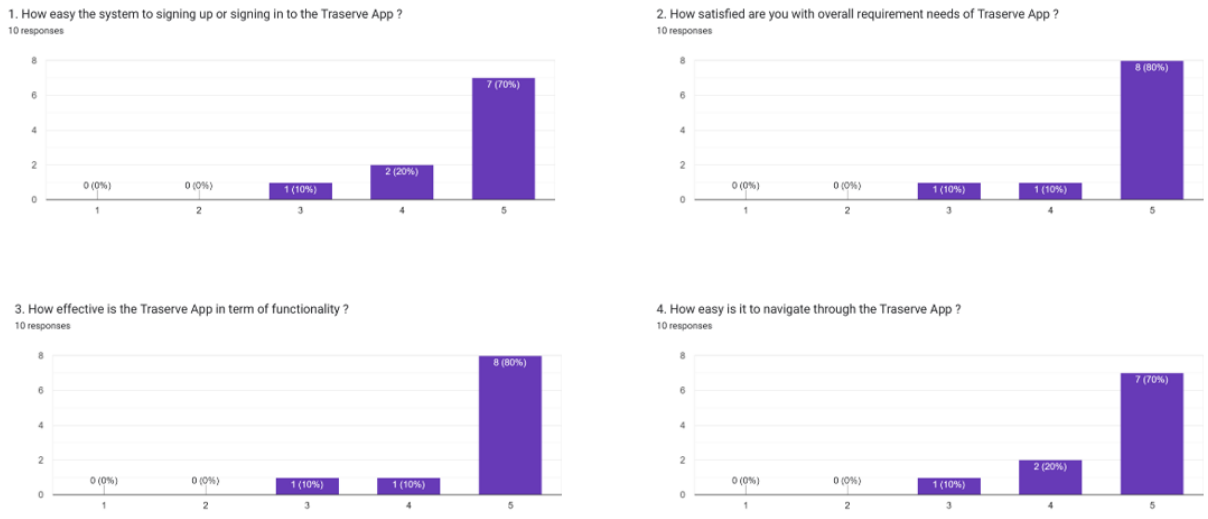


Fig. 25 *Graphs of Testing Result for System Functionality*

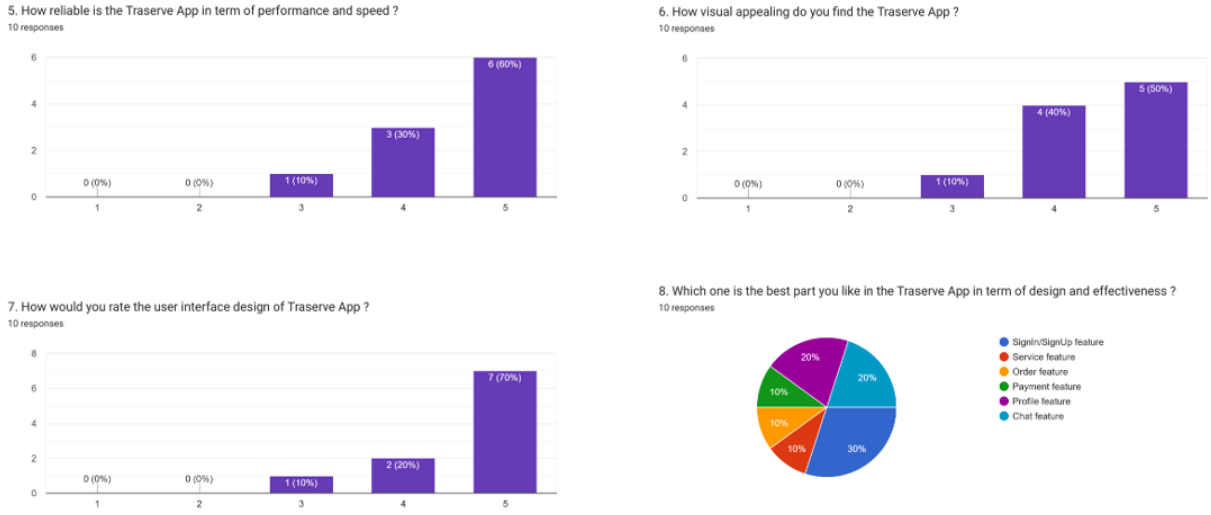


Fig. 26 *Graphs and Pie Chart of Testing Result for System Functionality*

Fig. 25 and Fig. 26 presents the results from user acceptance testing for the Traserve: Tradesman Service Application. The graph indicates as scale from 1(Strongly Disagree) to 5(Strongly Agree). It yielded positive results across all evaluated areas, with 10 respondents participating in the survey. The ease of signing up and signing in was also highly rated, with 70% giving it a 5, 20% a 4, and 10% a 3. In terms of overall requirements need filled, 80% of respondents rated the system a 5, and 10% rated it a 4, and 10% rated it a 3. Effectiveness saw 80% of respondents rating it a 5, 10% a 4, and 10% a 3. Reliability in performance and speed was rated a 5 by 60% and a 4 by 30%, and 10% for a 3. No respondents rated the system below a 3 in any category, indicating a strong overall approval. Based on this feedback, it is recommended to maintain and enhance the current system functionality, address any minor concerns with the sign in and signup process, and ensure ongoing system reliability and performance to sustain user confidence. The user interface design received a rating of 5 from 70% of respondents, 4 from 20%, and 3 from 10%. Navigation ease was rated a 5 by 70%, a 4 by 20%, and a 3 by 10%. Visual appeal was rated a 5 by 50%, a 4 by 40%, and a 3 by 10%. Recommendations include maintaining the current design standards, improving navigation ease, and ensuring the system continues to meet user needs effectively. Lastly, the pie chart statistic on the respondents on which on the respondents choose as the best module/ features of the app. Its shows all the module being chosen with SignIn and SignUp feature at the top with 30%, Chat feature with 20%, Profile feature with 20%, Service feature, Order feature, and Payment feature with 10% respectively.

6. Conclusion & Future Works

In conclusion, mobile based booking system Traserve: Tradesman Service Application is developed for overcome a few problems that occur which is overlapping order, lack of centralized data, miscommunication, and non-streamlined system. This system has a platform that is friendly to the user and easy to use. Also, the system can perform sign in and signup with authentication, manage account, manage service, manage order, manage payment, manage chat. It improves operational efficiency by supports robust data management with secure databases for user profiles, service detail, order process event and other. This initiative is expected to improve participant and crew experience, increase customer, and significantly improve the overall management and sustainability of Traserve booking app.

For future improvements, the Traserve: Tradesman Service Application could be enhanced to better capture and reflect a wider array of user preferences. Additionally, integrating real-time data updates for more feature such as overview, would make the recommendations more accurate and timelier. Lastly, adding notification functions would help users stay updated on new service and system updates.

Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support.

Conflict of Interest

Authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

This journal requires that all authors take public responsibility for the content of the work submitted for review. The contributions of all authors must be described in the following manner:

The authors confirm contribution to the paper as follows: **study conception and design**: Muhammad Ashraff Bin Azhan, Ruhaya Binti Ab Aziz; **data collection**: Muhammad Ashraff Bin Azhan, Ruhaya Binti Ab Aziz; **analysis and interpretation of results**: Muhammad Ashraff Bin Azhan, Ruhaya Binti Ab Aziz; **draft manuscript preparation**: Muhammad Ashraff Bin Azhan, Ruhaya Binti Ab Aziz. All authors reviewed the results and approved the final version of the manuscript.

References

Journal

[1] K. L. Tan and Y. M. Wong, (2021). Digital Transformation in Malaysian Service Industries: A Case Study Analysis, Journal of Southeast Asian Economies, vol. 38, no. 2, pp. 67-82.

[2] M. H. Rahman, R. Jerry and T. Williams, (2023). Real-time Communication Systems in Mobile Applications: A Comprehensive Survey, Journal of Systems and Software, vol. 180, pp. 111-122.

[3] P. Johnson and R. Miller, (2022). Firebase as Backend Service for Mobile Applications: Security and Performance Analysis, IEEE Transactions on Mobile Computing, vol. 20, no. 4, pp. 892-904.

[4] V. Kumar, S. Patel and R. Joshi, (2022) Real-time Database Management in Service-Based Mobile Applications, International Journal of Advanced Computer Science and Applications, vol. 13, no. 1, pp. 156-169.

[5] S. Ahmad, C. Z. Yang and J. S. Lee, (2021) Performance Analysis of Flutter Applications on Cross-Platform Mobile Development, International Journal of Advanced Computer Science and Applications, vol. 12, no. 3, pp. 98-107.

[6] Mamat, N., & Azizan, N. (2024). The planning process of "HOMERESC" home service system. Malaysian Journal of Science Health & Technology, vol. 10, no 2, pp. 152–164.

[7] Khan, N. R., Shaikh, M. A., & Ansari, M. H. (2023). Online service booking platform with payment integration. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, vol 9, no. 2, pp. 123–129.

Webpage

[8] Zonar. (2025) Zonar Malaysia. Available: https://zonar.my/

[9] TukangGo. (2025) Join as a Service Provider. Available: https://www.tukanggo.my/join.html

[10] Recommend.my. (2025) Find Recommended Services in Malaysia. Available: https://www.recommend.my/

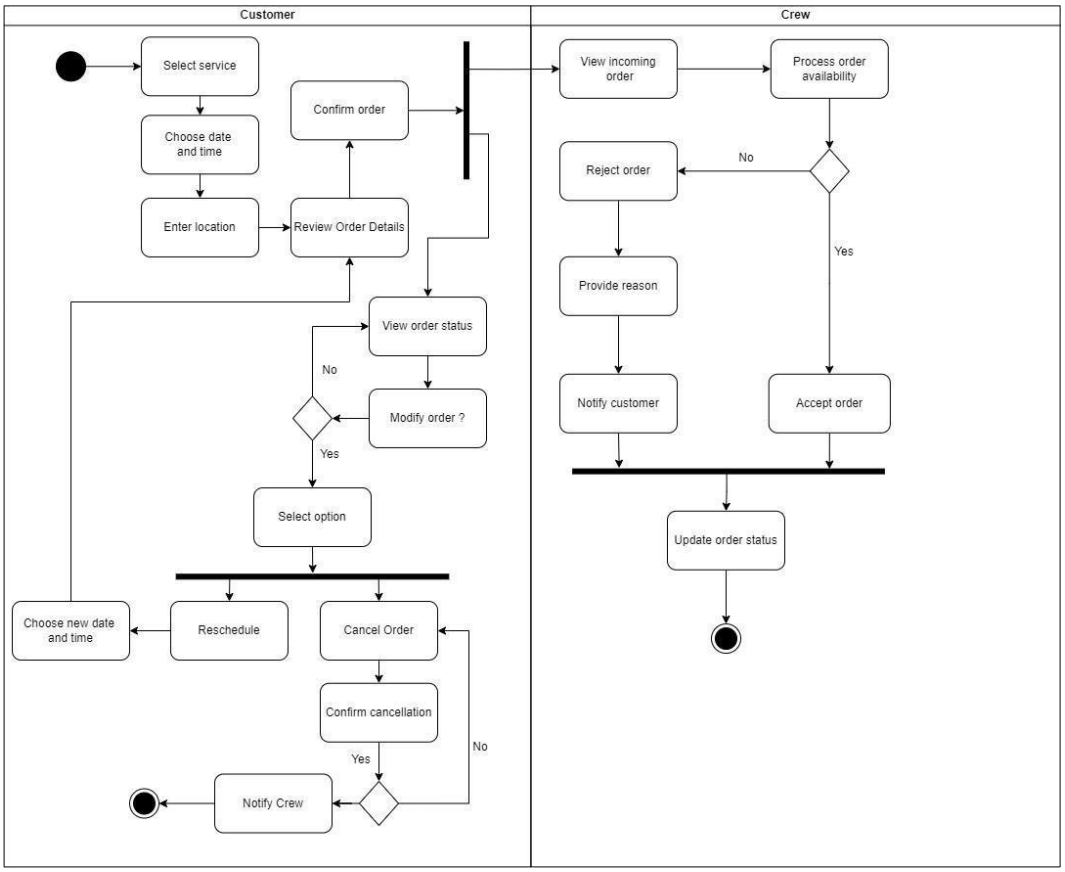
[11] D S Maylawati et al (2018). IOP Conf. Ser.: Mater. Sci. Eng. 288 012047 Available: https://www.researchgate.net/publication/322693037_Systematic_Design_of_Expert_System_Using_Unified_Modelling_Language

[12] Breja. M (2024). Prototyping Model – Software Engineering. Available: https://www.geeksforgeeks.org/software-engineering-prototyping-model/

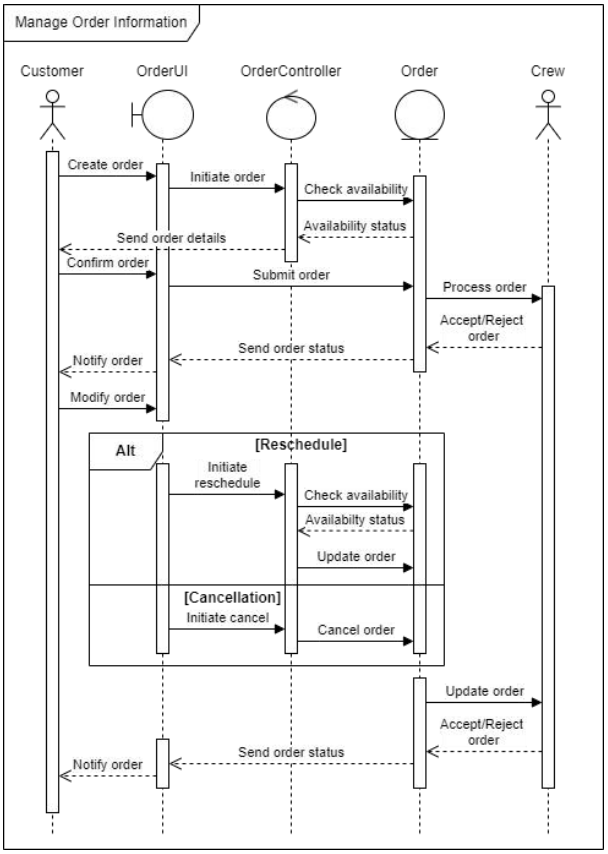
[13] Rathod, R., Dhuri, A., & Kadam, M. (2021). An online system for household services. International Journal of Engineering Research & Technology (IJERT). Available: https://www.ijert.org/an-online-system-for-household-services

[14] MarketBox. (2022). Advantages of using online booking systems for service businesses. MarketBox Blog. Available: https://www.gomarketbox.com

Appendix A: Activity Diagram

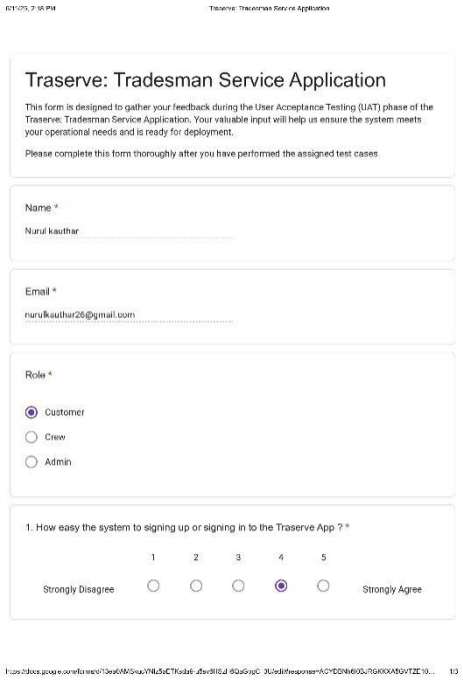


Appendix B: Sequence Diagram



Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 1119-1137

Appendix C: User Testing Form



Traserve: Tradesman Service Application

This form is designed to gather your feedback during the User Acceptance Testing (UAT) phase of the Traserve: Tradesman Service Application. Your valuable input will help us ensure the system meets your operational needs and is ready for deployment.

Please complete this form thoroughly after you have performed the assigned test cases.

Name *

Nurul kauthar

Email *

nurulkauthar26@gmail.com

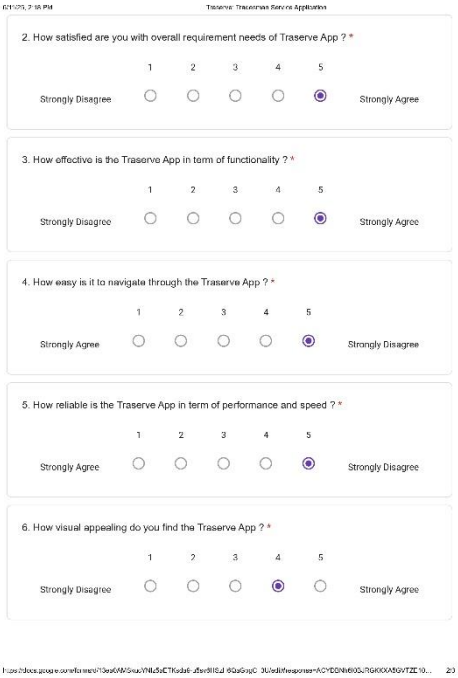
Role *

- Customer
- Crew
- Admin

1. How easy the system to signing up or signing in to the Traserve App ? *

1 2 3 4 5

Strongly Disagree — Strongly Agree



2. How satisfied are you with overall requirement needs of Traserve App ? *

1 2 3 4 5

Strongly Disagree — Strongly Agree

3. How effective is the Traserve App in term of functionality ? *

1 2 3 4 5

Strongly Disagree — Strongly Agree

4. How easy is it to navigate through the Traserve App ? *

1 2 3 4 5

Strongly Agree — Strongly Disagree

5. How reliable is the Traserve App in term of performance and speed ? *

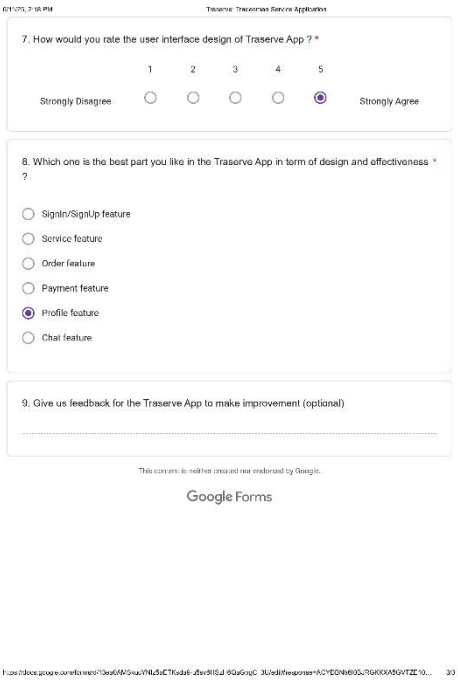
1 2 3 4 5

Strongly Agree — Strongly Disagree

6. How visual appealing do you find the Traserve App ? *

1 2 3 4 5

Strongly Disagree — Strongly Agree



7. How would you rate the user interface design of Traserve App ? *

1 2 3 4 5

Strongly Disagree — Strongly Agree

8. Which one is the best part you like in the Traserve App in term of design and effectiveness * ?

- SignIn/SignUp feature
- Service feature
- Order feature
- Payment feature
- Profile feature
- Chat feature

9. Give us feedback for the Traserve App to make improvement (optional)

This content is neither created nor endorsed by Google.

Google Forms