Plagiarism Detector .net

Apr 22, 2025

Plagiarism Scan Report

Characters:4912	Words:720
Sentences:33	Speak Time: 6 Min

9% Exact Matched | 0% Partial Matched

Excluded URL: None

Content Checked for Plagiarism

Description The system utilizes structured data gathered from users and ambulance providers. The key datasets involved include: - User Data: Name, contact number, age, medical needs - Provider Data: Ambulance type, status, driver details - Booking Data: Timestamp, user ID, ambulance ID, booking status The system uses MySQL for persistent data storage, chosen for its reliability and ease of integration with the backend. Data integrity is ensured using constraints and validations at both the application and database levels. Data is exchanged in JSON format between frontend and backend for flexibility and compatibility across different platforms. 3.3 Exploratory Data Analysis (EDA) Although not a data-centric project, EDA was conducted during testing to analyze usage patterns. The following insights were gathered: - Peak usage hours (8 AM - 12 PM and 6 PM - 9 PM) - Most frequently requested ambulance types - Average provider response time This data was used to optimize notification frequency and improve system load handling during peak periods. Charts were created using matplotlib for internal presentation. 3.4 Procedure / Development Life Cycle We followed the Agile methodology for this project, which allowed us to adapt the features based on continuous feedback and iterative testing. The process included: 1. Planning and Requirement Analysis: Stakeholders (users, providers, admin) were consulted to collect functional and non-functional requirements. 2. Design: System design involved creating wireframes, use case diagrams, and database schema. The architecture was documented and reviewed. 3. Implementation: Development was split into four major modules: - User Module - Provider Module - Admin Dashboard - Notification System 4. Integration: Individual modules were combined and tested as a complete system. 5. Testing: Manual and automated testing was performed for each module. 6. Deployment: Hosted the application on a cloud platform after pilot testing. 3.5 Modules and Their Implementation 1 . Each module has been designed to serve a distinct role in the ecosystem. User Module: - Register/Login with validation - Raise ambulance request with form inputs - View past bookings and status updates Ambulance Provider Module: - Register/Login after approval by admin - Mark ambulance availability - Accept or reject bookings with comments Admin Module: - View system activity dashboard - Manage and verify ambulance provider profiles - Review booking

logs and user reports 3.6 Notifications & Alerts To keep all parties informed in real-time, we implemented a robust notification system using Firebase Cloud Messaging (FCM). Notifications are triggered on the following events: - Booking request confirmation - Ambulance en route - Booking cancellation or delay Each message is timestamped and logged for reference. 3.7 UI/UX Development The frontend is developed using Flutter, which allows cross-platform deployment. UI elements follow a clean, modern design principle, with a focus on accessibility and speed. Key features include: - Dark mode - Button hierarchy for emergencies - Clear distinction between booking statuses The admin dashboard was built using React for dynamic content rendering and ease of integration with APIs. 3.8 Security & Data Privacy Security was a key focus area. Measures implemented include: - Role-based access control for users, providers, and admins - SSL encryption for data in transit - Encrypted user passwords using bcrypt - Input sanitization to prevent injection attacks All user data is stored securely in a managed database, and regular backups are scheduled. 3.9 Testing and Evaluation A multi-tiered approach to testing was adopted: - Unit Testing: Every function was independently tested - Integration Testing: Checked inter-module interaction - User Acceptance Testing (UAT): Conducted with a closed group of users Evaluation Metrics: - Booking success rate - Average response time - User feedback rating These metrics were used to iterate and improve the app before full deployment. 3.10 Deployment Strategy The application was deployed in three phases: Phase1: Internal testing among team members Phase2: Pilot launch in a local urban region Phase 3: Full deployment via cloud with analytics 2 . support We used Heroku for deployment and Firebase for analytics and crash reporting. Conclusion The methodology adopted in the development of the Ambulance Booking System ensured robust planning, design, implementation, and validation. Following agile practices, combined with continuous feedback and secure coding principles, the app is now a ready-to-use solution. EXPERIMENTAL SETUP 1. Software Requirements: 3 . - Web Browser (Chrome/Firefox) – To access MIT App Inventor. - Python 3.x – For backend logic (if applicable). - Firebase Account – For cloud-based data storag

Sources

33% Plagiarized

There has been a growing public interest in the role and value of natural ecosystems and how they contribute to our quality of life and to human wellbeing.Missing: distinct | Show results with:

https://www.agriculture.gov.au/sites/default/files/documents/ecosystem-services.pdf

33% Plagiarized

Aug 9, 2020 • Prior to firebase, I've used ActiveRecord and PostgresQL just for data management alone and platforms such Heroku for deployment/hosting.Missing: support | Show results with:

https://samanbatool08.medium.com/what-i-learned-using-firebase-for-data-management-on-my-react-application-259232d40e3d

33% Plagiarized

... your Web Browser (Chrome, Firefox or Safari, no other Browsers are compatible.) So, in your browser, use your gmail account to log into App Inventor: http...

https://groups.google.com/g/mitappinventortest/c/UgmDchJ1Prg