MEHANI WEBSITE REQUERMENTS

MEHANI is an e-commerce services website that caters to normal users, service providers, and administrators involves several essential requirements to ensure that the platform functions effectively and meets the needs of all stakeholders. Here are some of the key requirements for such a project:

1. **User Registration and Authentication:**

   - User registration with email and password.

   - Social media login options.(optional)

   - Two-factor authentication for enhanced security.

2. **User Profiles:**

   - Allow users to create and manage their profiles.

   - Users can add personal information, shipping addresses, and payment methods.

3. **Search and Browse:**

   - A robust search feature to allow users to find services or products easily.

   - Filter and sort options.

4. **Service Listings:**

   - Service providers should be able to create and manage their service listings.

   - Include details such as service descriptions, pricing, availability, images, and reviews.

5. **Booking System:**

   - Users should be able to book services from providers.

   - Integration with a shopping cart for product-based services.

6. **Payment Processing:**

   - Secure payment gateway integration to facilitate transactions.

   - Support for various payment methods.

7. **Reviews and Ratings:**

   - Allow users to leave reviews and ratings for services or products.

   - Provide a rating system for service providers.

8. **Messaging and Notifications:**

   - In-site messaging system for communication between users and service providers.

   - Notifications for bookings, order updates, and messages.(optional)

9. **User Dashboard:**

   - Personalized dashboard for users to track their activity, orders, and messages.

10. **Service Provider Dashboard:**

    - Dashboard for service providers to manage listings, bookings, and earnings.

11. **Administrator Panel:**

    - An admin panel to manage users, service providers, listings, and resolve disputes.

    - Access control and user management.

12. **Security and Privacy:**(optional)

    - Data encryption and secure storage of user information.

    - Compliance with data protection regulations.

13. **SEO-Friendly:**

    - Search engine optimization features to improve the website's visibility.

14. **Responsive Design:**

    - Ensure the website works well on various devices and screen sizes.

15. **Analytics and Reporting:**

    - Collect and analyze user activity and sales data. (optional)

    - Generate reports for administrators and service providers.

16. **Performance and Scalability:**

    - Optimize the website for speed and scalability as traffic grows.

17. **Customer Support:**

    - Provide a mechanism for users to seek help or report issues.

18. **Terms and Conditions, Privacy Policy:**

    - Include legal documents that protect the interests of all stakeholders.

19. **Marketing and Promotion:**(optional)

    - Tools for promoting services and products.

    - Integration with social media and email marketing.

20. **Localization and Multiple Languages:**(optional)

    - Support for multiple languages and currencies if needed.

21. **Mobile App Integration (Optional):**

    - Develop mobile apps for iOS and Android to expand the user base.

22. **Feedback and Improvement System:**()

    - Collect feedback from users to continually improve the platform.

23. **Content Management System (CMS):**

    - A user-friendly CMS for adding and updating content.

24. **Legal Compliance:**

- Ensure compliance with relevant e-commerce and business laws and regulations.

25. **Testing and Quality Assurance:**

- Rigorous testing of the website to identify and resolve any bugs or issues.

26. **Backup and Data Recovery:**

- Regular data backups and a data recovery plan.

27. **Hosting and Infrastructure:**

- Choose a reliable hosting provider and infrastructure to ensure website stability.

These requirements serve as a starting point for my e-commerce services website project. we should refine these requirements, considering the specific needs of my target audience and the unique features of wer platform.

simplified database schema earlier:

Tables:

User Table.

Service Table.

Order Table.

Review Table.

Relationships in terms of one-to-many (1 to M), many-to-one (M to 1), and many-to-many (M to M) for the simplified database schema:

1. **One-to-Many (1 to M) Relationships**:

- **User to Service** (1 to M):

  - Each user (both normal users and service providers) can offer multiple services, but each service is associated with only one user. This is a one-to-many relationship.

- **User to Order** (1 to M):

- Each user can place multiple orders, but each order is associated with only one user. This is a one-to-many relationship.

- **Service to Order** (1 to M):

- Each service can be ordered multiple times, but each order is associated with only one service. This is a one-to-many relationship.

- **User to Review** (1 to M):

- Each user can leave multiple reviews, but each review is associated with only one user. This is a one-to-many relationship.

- **Service to Review** (1 to M):

- Each service can have multiple reviews, but each review is associated with only one service. This is a one-to-many relationship.

2. **Many-to-One (M to 1) Relationships**:

- **Service to User** (M to 1):

- Multiple services can be offered by a single user (service provider). This is a many-to-one relationship.

- **Order to User** (M to 1):

- Multiple orders can be placed by a single user. This is a many-to-one relationship.

- **Order to Service** (M to 1):

  - Multiple orders can be associated with a single service. This is a many-to-one relationship.

- **Review to User** (M to 1):

  - Multiple reviews can be left by a single user. This is a many-to-one relationship.

- **Review to Service** (M to 1):

  - Multiple reviews can be associated with a single service. This is a many-to-one relationship.

3. **Many-to-Many (M to M) Relationships**:

  - In the simplified schema, there isn't a straightforward many-to-many relationship between the main entities (User, Service, Order, and Review). However, we can introduce additional tables to represent many-to-many relationships if needed. For example:

  - **Many-to-Many between Users and Services**:

    - We can create a table (e.g., `User_Service`) that includes two foreign keys, one referencing the User and the other referencing the Service. This table can represent which users offer which services, allowing for many-to-many relationships between users and services.

  - **Many-to-Many between Users and Orders**:

    - We can create a table (e.g., `User_Order`) with foreign keys for User and Order. This table can represent the relationship between users and orders, indicating which users placed which orders.

- **Many-to-Many between Services and Reviews**:

  - We can create a table (e.g., `Service_Review`) with foreign keys for Service and Review. This table can represent the relationship between services and the reviews they receive.

These additional tables allow us to model many-to-many relationships by breaking them down into multiple one-to-many or many-to-one relationships.