1. **Gather and Finalize Requirements**

- **Understand the Form:** Review the attached registration form to list all required fields (name, contact info, etc.).
- **Define Registration Options:** Identify the different registration categories, packages, individual tickets, and souvenir ads.
- **User Flow Mapping:** Outline the attendee's journey:
    - Visit website → Fill out registration form → Choose packages/tickets/ads → Add to cart → Checkout & pay → Receive confirmation emails.

---

2. **Plan the Project Architecture**

- **Front-end vs. Back-end:** Decide on the structure (e.g., using HTML, CSS, and JavaScript for the front-end; PHP for back-end processing).
- **Database Design:** Sketch a schema that might include tables such as:
    - `users` (or `attendees`)
    - `registration_categories`
    - `packages` / `tickets` / `ads`
    - `orders` (to store overall order information)
    - `order_items` (for each package/ticket/ad purchased)
- **Security & Payment:** Outline secure payment handling (SSL, secure sessions) and determine the payment gateway (Stripe, PayPal, etc.).

---

3. **Design and Develop the Registration Form and Website Pages**

- **Home Page:** Introduce the convention, provide event details, and link to the registration page.
- **Registration Form:** Create a PHP-driven page with:
    - Input fields matching the attached registration form.
    - Validation both client-side (JavaScript) and server-side (PHP).
- **Package/Ticket Selection:**
    - Provide a clear interface for users to choose one or more items.
    - Use radio buttons, checkboxes, or a drop-down menu for selection.
- **Cart Page:**
    - Implement session-based cart functionality where users can review and update their selections.

---

4. **Implement the Checkout Process and Payment Integration**

- **Checkout Page:**
  - Display a summary of the order.
  - Capture billing/shipping details if necessary.
- **Payment Gateway Integration:**
  - Integrate your chosen payment gateway API using PHP.
  - Ensure proper handling of success and error scenarios.
- **Transaction Security:**
  - Use secure coding practices.
  - Ensure the site runs over HTTPS to protect sensitive information.

---

5. **Automate Confirmation Emails**

- **Email Setup:**
  - Configure PHP's mail function or use an external service (like PHPMailer, SendGrid, etc.).
- **Confirmation Email Content:**
  - Create a detailed confirmation email template that includes:
    - Order summary
    - Registration details
    - Any additional instructions
- **Multiple Recipients:**
  - Ensure the email is sent both to the attendee and to `treasurernluaa@gmail.com` .

---

6. **Develop an Administrative Backend (Optional but Recommended)**

- **Admin Dashboard:**
  - Build a secure admin panel where organizers can:
    - View registrations and orders.
    - Monitor payment statuses.
    - Export attendee lists if necessary.
- **Data Management:**

    - Implement features for order search, filtering, and detailed views.

---

7. **Testing and Quality Assurance**

- **End-to-End Testing:**
    - Test the entire registration flow from form submission to payment processing and email confirmation.
- **Security Testing:**
    - Ensure that all inputs are sanitized and secure.
    - Test for vulnerabilities (e.g., SQL injection, XSS).
- **User Acceptance Testing:**
    - Get feedback from a few users or stakeholders to ensure the process is intuitive.

---

8. **Deployment and Maintenance**

- **Deployment:**
    - Set up a production environment with PHP and your chosen database (MySQL, PostgreSQL, etc.).
    - Ensure that SSL is installed and configured.
- **Post-Deployment:**
    - Monitor the system for any issues.
    - Be prepared to perform regular maintenance and updates as needed.