Technical Foundation Plan for Insight Artistry

1. Define Technical Requirements

Frontend Requirements:

- A responsive interface that includes:
    - Home Page: Displays featured artworks, categories, and promotions.
    - Product Listing Page: Allows filtering by categories (e.g., Paintings, Sculptures) and price range.
    - Product Details Page: Displays artwork details, customization options, and related items.
    - Cart and Checkout Pages: Simplify the purchasing process with real-time updates.
    - User Profile Page: Shows order history, customization requests, and saved items.

Backend Requirements:

- **Database:** Firebase for storing:
    - Products (details, stock levels, and images).
    - Customers (registration details, preferences).
    - Orders (status, payment history).
- **APIs:** Integration for:
    - Payment processing (Stripe, PayPal).
    - Shipping updates (Third-party API).
    - Customization management.

Third-Party APIs:

- Payment Gateway API for secure transactions.
- Shipping API for real-time delivery tracking.
- Customization API to process user modification requests.

2. Design System Architecture

High-Level Architecture Diagram:

```
[Frontend (Next.JS)]
     |
[Firebase Database] <--> [Third-Party APIs]
     |                   |
[Authentication]     [Payment Gateway, Shipping API]
```

Workflows:

1. **User Registration:**
   - **Input:** User details.
   - **Action:** Data stored in Firebase.
   - **Output:** Account creation and login credentials.
2. **Product Browsing:**
   - **Input:** User selects a category or applies filters.
   - **Action:** Firebase query fetches matching artworks.
   - **Output:** Artworks displayed dynamically.
3. **Order Placement:**
   - **Input:** Items added to cart and checkout initiated.
   - **Action:** Firebase records order details; payment API processes transaction.
   - **Output:** Order confirmation sent to user.
4. **Customization Request:**
   - **Input:** User submits customization details.
   - **Action:** Request logged in Firebase; admin notified for approval.
   - **Output:** Cost estimate shared with user.
5. **Shipping Updates:**
   - **Input:** Order ID queried via Shipping API.
   - **Action:** Fetch real-time status.
   - **Output:** Delivery updates displayed in the user profile.

3. Plan API Requirements

Endpoints:

1. **Fetch Products:**
   - Endpoint: /products
   - Method: GET
   - **Response:**

```
{
  "id": "1",
  "name": "Abstract Painting",
  "price": 500,
  "category": "Painting",
  "stock": 10,
  "image": "url"
}
```

2. **Place Order:**

- Endpoint: /orders
- Method: POST

Payload:

```
{
  "userId": "123",
  "products": [{"id": "1", "quantity": 2}],
  "paymentStatus": "Paid"
}
```

Response:

```
{"orderId": "456", "status": "Confirmed"}
```

3. **Track Shipment:**

- Endpoint: /shipment
- Method: GET
- **Response:**

```
{
  "orderId": "456",
  "status": "In Transit",
  "eta": "3 days"
}
```

4. **Customization Request:**

- Endpoint: /customization
- Method: POST
- **Payload:**

```
{
  "productId": "123",
  "size": "24x36",
  "material": "Canvas",
  "colorScheme": "Black and White"
}
```

Response:

```
{"requestId": "789", "status": "Received"}
```

4. Write Technical Documentation

System Architecture Overview:

- Frontend: Built using Nest.JS for a responsive, user-friendly interface.
- Backend: Firebase database for real-time data synchronization.
- Third-Party APIs: Integrated for payment processing, shipping updates, and customization handling.

Key Workflows:

1. User Onboarding:
    - ○ New users register, and data is stored in Firebase.
    - ○ User accounts enable browsing, purchasing, and customization.
2. Order Management:

   - Products are added to the cart and processed at checkout.
   - Payment confirmation triggers order recording and shipping.
3. Customization Handling:
   - User modifications are logged and forwarded to admins for processing.