Flutter UI Reproduction Assessment Test

Objective

The objective of this assessment test is to evaluate the candidate's proficiency in Flutter development by reproducing a user interface (UI) provided in Figma. Additionally, the candidate will be required to integrate the UI with a catalog API to display relevant data and write tests to ensure code reliability.

Instructions

1. **UI Reproduction:**

   - You will be provided with a Figma link containing the UI design.
   - Reproduce the UI design using Flutter.
   - Ensure that the UI is visually accurate and responsive across different screen sizes.

2. **API Integration:**

   - Utilise the provided catalog API to fetch data.
   - Display the fetched data in relevant UI components as per the design.
   - Handle loading states and error states appropriately.

3. **Testing**:

   - Write unit tests for critical UI components and API integration logic.
   - Ensure adequate test coverage to validate the functionality and reliability of the code.
   - Use Flutter's testing framework or suitable packages for writing tests.

4. **Requirements**:

   - Use Flutter for UI development.
   - Use HTTP requests or suitable packages for API integration.
   - Implement proper error handling and loading indicators.
   - Ensure clean and maintainable code adhering to Flutter best practices.
   - Provide necessary comments to explain complex logic or functionality.

5. **Deliverables**:

   - Submit your Flutter project containing the reproduced UI, integrated API, and unit tests.
   - Include clear instructions on how to run the project and execute the tests.
   - Provide any additional notes or considerations regarding your implementation.

Resources:

- Figma UI Design Link: UI Figma
- Catalog API:

https://catalog-api.dev-cat.scalapay.com/v1/products/search?q=**${searchQuery}**&per_page=30&page=1&filter_by=&sort_by=**${sortType}**%3A**${sortDirection}**&**${otherSortingParam}**&partnerId=scalapayappit&source=trovaprezzi&language=it&country=IT

Or

https://catalog-api.dev.scalapay.com/v1/products/search?q=**${searchQuery}**&per_page=30&page=1&filter_by=&sort_by=**${sortType}**%3A**${sortDirection}**&**${otherSortingParam}**&partnerId=scalapayappit&source=trovaprezzi&language=it&country=IT

$searchQuery: any string
$sortType: _*text*_match OR selling_price
$sortDirection: asc OR desc
$otherSortingParam: minPrice AND maxPrice

Examples:

SORT SELLING PRICE DESC: (minPrice AND maxPrice)

https://catalog-api.dev-cat.scalapay.com/v1/products/search?q=nike&per_page=30&page=1&filter_by=&sort_by=selling_price%3Aasc&minPrice=13.0&maxPrice=278.0&partnerId=scalapayappit&source=trovaprezzi&language=it&country=IT

Or

https://catalog-api.dev.scalapay.com/v1/products/search?q=nike&per_page=30&page=1&filter_by=&sort_by=selling_price%3Aasc&minPrice=13.0&maxPrice=278.0&partnerId=scalapayappit&source=trovaprezzi&language=it&country=IT

GENERIC SORTING:

https://catalog-api.dev-cat.scalapay.com/v1/products/search?q=nike&per_page=30&page=1&filter_by=&sort_by=_text_match%3Adesc&partnerId=scalapayappit&source=trovaprezzi&language=it&country=IT

OR

https://catalog-api.dev.scalapay.com/v1/products/search?q=nike&per_page=30&page=1&filter_by=&sort_by=_text_match%3Adesc&partnerId=scalapayappit&source=trovaprezzi&language=it&country=IT

Note:

- You are allowed to use any additional packages or libraries deemed necessary for completing the task.

- The focus will be on the accuracy of UI reproduction, proper API integration, writing effective tests, and maintaining code quality.