Simula SSP SDK

Developer Assessment

Overview

This is a practical coding assessment designed to evaluate your ability to build production-ready SDK components for native advertising. You will build core SDK architecture for both React (Web) and Flutter (iOS/Android).

Time Allocation: 3 days

Difficulty Level: Senior Developer

Focus Areas: SDK Architecture, Performance Optimization, Cross-Platform Development

Background

Simula is building a native ad network that provides SDKs for publishers to integrate ads into their content feeds, explore pages, and chat interfaces. The ads must feel native to the app's UI while maintaining excellent performance.

Objective

Build the core SDK architecture and a NativeBanner component for both platforms:

- React (Web)
- Flutter (iOS + Android)

You must also implement a Provider/Manager component for API key authentication.

Critical Requirements

1. **Zero-Flicker Loading**
    a. Publisher feeds must NEVER jump or shift layouts
    b. Reserve ad container space BEFORE content arrives
    c. Maintain perfect layout stability during load
2. **Caching**
    a. If a user scrolls past an ad and returns, show it instantly
    b. NO second network request for the same ad in a session
3. **Impression Tracking**
    a. An impression is recorded ONLY when:
    b. Ad is ≥50% visible on screen
    c. For a continuous 1 second duration
    d. If user scrolls away before 1 sec → timer RESETS
    e. Send tracking request to endpoint using aid from fetch response
4. **Lifecycle Management**
    a. Cancel all active network requests when ad scrolls out of view
    b. Clear timers when page/screen closes
5. **Flexible Widths**
    a. Support fixed pixel values
    b. Support percentage widths
    c. Support fill-container layouts

    d. Enforce minimum width: 130px
6. **Loading States**
    a. Show loading spinner while fetching
    b. Graceful transitions to ad content
7. **Error Handling**
    a. Never crash the publisher's app
    b. Collapse ad container on session, fetch, track failures

API Endpoints

Create Session

Endpoint: POST https://simula-api-701226639755.us-central1.run.app/session/create

Headers:

```
{
  "Authorization": "Bearer <API_KEY>"
}
```

Request Body:

```
{}
```

Response Body:

```
{
  "sessionId": "abcd1234"
}
```

Fetch Ad

Endpoint: POST
https://simula-api-701226639755.us-central1.run.app/render_ad/ssp/native

Request Body:

```
{
  "session_id": "abcd1234",
  "slot": "explore",
  "position": 0
}
```

Response Headers:

```
{
  "aid": "1234"
}
```

Response Body: (raw HTML)

```
<iframe id="ad-iframe" srcdoc="{ad_content}" style="width: 100%; border: none; display: block;"
onload="this.style.height=this.contentDocument.documentElement.scrollHeight+'px';"></iframe>
```

Track Impression

Endpoint: POST
https://simula-api-701226639755.us-central1.run.app/track/engagement/impression/{ad_id}

Request Body:

```
{}
```

Example Implementation

React SDK

```
// Provider setup
import { SimulaProvider } from '@simula/react-sdk';

function App() {
  return (
    <SimulaProvider apiKey="your-api-key">
      <YourApp />
    </SimulaProvider>
  );
}
```

```
// Component usage
import { NativeBanner } from '@simula/react-sdk';

function Feed() {
  return (
    <NativeBanner
      slot="explore"
      position={0}
      width="100%"
```

```
      onError={(error) => console.error(error)}
      onImpression={() => console.log('Tracked!')}
    />
  );
}
```

Flutter SDK

```
// Provider setup
void main() {
  SimulaManager.initialize(apiKey: 'your-api-key');
  runApp(MyApp());
}
```

```
// Widget usage
class FeedScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        NativeBanner(
          slot: 'explore',
          position: 0,
          width: double.infinity,
          onError: (error) => print(error),
          onImpression: () => print('Tracked!'),
        ),
      ],
    );
  }
}
```

Test API Key

```
SIMULA_API_KEY=pub_a7f3c92d8b1e4f6fa2d4c1e9b73a5c4e
```

Works for both web and mobile.

Deliverables

1. Code Repositories

- Separate repos for React, Flutter SDKs.

- Clean, well-documented code
- Proper package structure (npm for React, pub for Flutter)

2. Package Setup

- React: Publishable npm package (test with npm link)
- Flutter: Publishable pub package (test with local path)
- Clear installation instructions

3. Testing Applications

- You can test your SDK inside these mock publisher apps:
    - **Web:** https://github.com/Simula-AI-SDK/react-mock-public
    - **Mobile:** https://github.com/Simula-AI-SDK/flutter-mock-public
- Demonstrate all features working
- Show edge cases handled gracefully

4. Demo Video

- Screen recording showing:
    - Zero-flicker loading in action
    - Impression tracking (use dev tools to verify timing)
    - Caching behavior on scroll
    - Error handling scenarios
    - Both desktop AND mobile views (iOS/Android for Flutter)
    - 3 minute max

Evaluation Criteria

1. **Code Quality (30%)**
    a. Clean, maintainable architecture
    b. Proper separation of concerns
    c. Type safety (TypeScript/Dart)
    d. Error handling
2. **Functionality (30%)**
    a. All requirements met
    b. Edge cases handled
    c. Cross-platform consistency
3. **Technical Depth (20%)**
    a. Caching works
    b. Good lifecycle management
4. **Developer Experience (20%)**
    a. Easy to integrate
    b. Helpful error messages
    c. Sensible defaults

Submission Instructions

1. GitHub Repository
    a. Grant access to: athreya@simula.ad
    b. Include all source code and documentation
2. Demo Video
    a. Upload to Loom/Google Drive
    b. Include link in README
3. Email Submission
    a. Send to: athreya@simula.ad
    b. Subject: "Simula SSP SDK Assessment - [Your Name]"
    c. Include Github link, demo link
    d. **Deadline: 3 days from receipt of this assessment**

Support

For technical clarifications or API issues:

- Email: athreya@simula.ad
- Response Time: Within 24 hours

Good luck! We're excited to see what you build.