**Q1. Key differences between explicit and implicit intents**

**Answer:**

* **Explicit Intent**: Used when you know exactly which app or component (like an activity or service) should handle the request. Example: Opening a specific activity in your app.
* **Implicit Intent**: Used when you don’t specify the exact component but instead request an action, allowing any app that can handle it to respond. Example: Opening a web link in a browser.

**Q2. Handling a scenario where an application is unavailable to handle an implicit intent**

**Answer:**

Before sending an implicit intent, check if any app can handle it using **PackageManager.resolveActivity()**. If no app is found, show an error message or disable the feature to prevent crashes.

**Q3. Securing an intent to prevent unauthorized data access**

**Answer:**

* Use **intent filters** and permissions to restrict access.
* Mark sensitive data as **private** (e.g., FLAG\_SECURE).
* Validate the sender before processing data.
* Use **PendingIntents** with proper flags to prevent unauthorized execution.

**Q4. When to prefer Parcelable over Serializable, and why?**

**Answer:**

* Use **Parcelable** when passing data between activities because it is **faster** and optimized for Android.
* Use **Serializable** if you need to save objects to a file, but it's slower than Parcelable.

**Q5. Ensuring large data transfers do not affect performance or security**

**Answer:**

* Compress or **split** large data into smaller chunks.
* Use a **background thread** instead of the main thread.
* Store large data in **files** or a **database** instead of passing it through intents.
* Use **encryption** for sensitive data.

**Q6. Using implicit intents securely to interact with other apps**

**Answer:**

* Always **verify the receiving app** before sending data.
* Use **specific intent actions and categories** to avoid unwanted apps accessing the data.
* If sending sensitive data, use **permissions** and **content providers** with security checks.

**Q7. Best practices for ensuring data integrity during inter-app communication**

**Answer:**

* Use **digital signatures** or **hashing** to verify data has not been tampered with.
* Send data using **secure content providers** instead of direct intents.
* Validate **input and output** to prevent security risks like injection attacks.
* Use **encrypted communication** for sensitive data.