|  |  |  |
| --- | --- | --- |
|  | **2, LEBUH ACHEH, GEORGE TOWN**  **10300 GEORGE TOWN**  **PULAU PINANG**  **INFORMATION SHEET** | |
| **PROGRAM’S CODE & NAME** | J620-002-4:2020 FRONT-END SOFTWARE DEVELOPMENT | |
| **LEVEL** | FOUR (4) | |
| **COMPETENCY UNIT NO. AND TITLE** | J620-002-4:2020-C04: LOCAL MOBILE DEVELOPMENT | |
| **WORK ACTIVITIES NO. AND STATEMENT** | 1. CREATE MOBILE APP DESIGN MOCK-UP ELEMENTS. 2. PLAN MOBILE APP DESIGN STRUCTURE. 3. TRANSFORM MOCK-UP TO MOBILE APP. 4. INTEGRATE MOBILE APP WITH DATA SOURCE. 5. VERIFY SUCCESSFUL API INTEGRATION 6. VERIFY DEVELOPED MOBILE APP. 7. **VERIFY MOBILE APP ACCESSIBLE GLOBALLY.** | |
| **CODE NO.** | J620-002-4:2020-C04/IS(14/15) | Page: 1 of18 |

**TITLE**:

**APPLICATION SUBMISSION GUIDELINES**

**PURPOSE**:

This information sheet is intended to provide insight and knowledge to trainees with regards to the fundamentals of application submission guidelines.

**INFORMATION:**

This information sheet provides useful notes and explanations on fundamental concepts of application submission guidelines.

# **APPLICATION SUBMISSION CHECKLIST**

This checklist brings together the processes you should follow to ensure your app is launched successfully. The tasks are arranged to give you an idea of the normal sequence in which they would be actioned. However, you can handle the tasks in any sequence that works for you and skip steps as appropriate.

Table 1: Application Submission Checklist Tasks

|  |  |
| --- | --- |
| Tasks | Details |
| Understand the Developer Program Policies | The Developer Program Policies are designed to ensure that the Play store remains a trusted resource for Android users. Review the policies thoroughly as there are consequences for violations. |
| Prepare your developer account | To publish Android apps on Google Play, you will need to create a Google Play Developer account. Follow these steps to register for a Google Play Developer account:  Sign up for a Google Play Developer account. Once you have a Developer account, you can use the Play Console to publish and manage your apps.  Accept the Developer Distribution Agreement.  Pay registration fee.  Complete your account details. To process your request for a Play Developer account you may be asked for a valid government ID and a credit card, both under your legal name. If this information is determined to be invalid, your registration fee will not be refunded. |
| Plan for simultaneous releases | Releasing your app on multiple platforms and devices maximizes your promotion activities and the number of installs, so include it in your development plans upfront. If you cannot launch your app on all platforms at once, ask for users' contact details and let them know when your app is ready. |
| Target a recent API level | Every new Android version introduces changes that bring significant security and performance improvements as well as enhance the user experience of Android overall. Some of these changes only apply to apps that explicitly declare support through their targetSdkVersion manifest attribute (also known as the target API level).  Configuring your app to target a recent API level ensures that users can benefit from these improvements, while still allowing it to run on older Android versions. Targeting a recent API level also allows your app to take advantage of the platform's latest features to delight your users. Furthermore, as of Android 10 (API level 29), users see a warning when starting an app for the first time if the app targets Android 5.1 (API level 22) or lower. |
| Build your Android App Bundle | When you are ready to make your app available to users, either for testing or as a final product, build your Android App Bundle in Android Studio. Google Play will use your app bundle to generate, sign and serve optimized APKs for each user's device, resulting in a smaller app to download and install. |
| Run internal tests | Use the internal test track to push your app to up to 100 internal testers to get feedback before making your app available to external users in the closed, open, or production tracks. Use multiple closed test tracks for different versions of your app before pushing them to open test tracks or production. |
| Plan your app's Play store listing | Prepare the descriptions, promotional graphics, screenshots, and videos you will add to your app's Play store page. Make sure you include a link to your privacy policy if required. Localize your store listing in all the languages your app supports and for the countries you are targeting. |
| Upload your Android App Bundle to the closed or an open test track | Closed and open testing tracks can help greatly in discovering issues with your app, giving you the opportunity to fix those issues and raise the quality of your initial release. |
| Define your app's device compatibility | Let the Play store know which Android versions and device screen sizes your app is designed to work on. |
| Setup your app's price and countries of distribution | Once you have determined your monetization model, setup your app as free or paid and select the countries in which it will be distributed. |
| Opt-in to the right distribution options | From the pricing & distribution page, opt-in to specific devices and programs, such as Wear OS by Google, Android TV, and Designed for families. Google Play can then review your app and once approved, make it more discoverable for users. |
| Set up your in-app products and subscriptions | Confirm the countries you can sell into and currency and tax issues to consider. Then set up your merchant account and add details and prices for your in-app products and subscriptions. |
| Determine your app's content rating | Providing an appropriate rating for your app is a requirement of the Developer Program Policies but, it also ensures your app gets seen by the right age-based audiences. |
| Final checks and publishing | First, go back and double check you have done everything on this list. Now you are ready to publish your app to the production channel. If you are releasing an app update, use staged rollouts to release your update to progressively more users. This allows you to halt the update if you find an issue, so you can limit the number of users it affects. |

# **METHOD TO UPLOAD APP TO PLAY STORE**

After you build and sign the release version of your app, the next step is to upload it to Google Play to inspect, test, and publish your app. Before you get started, you might want to make sure you satisfy the following:

* If you have not already done so, enrol into Play App Signing, which is the recommended way to upload and sign your app. If you build and upload an Android App Bundle, you must enrol in app Play App Signing.
* Google Play supports compressed app downloads of only 150 MB or less. To learn more, read Compressed download size restriction.

## Inspecting APKs using bundle explorer

If you upload your app as an Android App Bundle, the Play Console automatically generates split APKs and multi-APKs for all device configurations your app supports. In the Play Console, you can use the App Bundle Explorer to see all APK artifacts that Google Play generates, inspect data such as supported devices and APK size savings, and download generated APKs to deploy and test locally.

## Testing using the internal test track

The easiest and fastest way to test your app after you upload an app bundle is by using Play Console’s internal test track. This new test track is aimed towards quickly sharing your app with a limited number of internal testers. And, because the only way to test downloading installing feature modules on demand is through Google Play, the internal test track is where you will want to upload your app bundle to fine-tune this type of user experience.

## Preparing and rolling out a release

Update your app bundle. After you upload your app to the Play Console, updating your app requires you to increase the version code you include in the base module, and build and upload a new app bundle. Google Play then generates updated APKs with new version codes and serves them to users as needed.

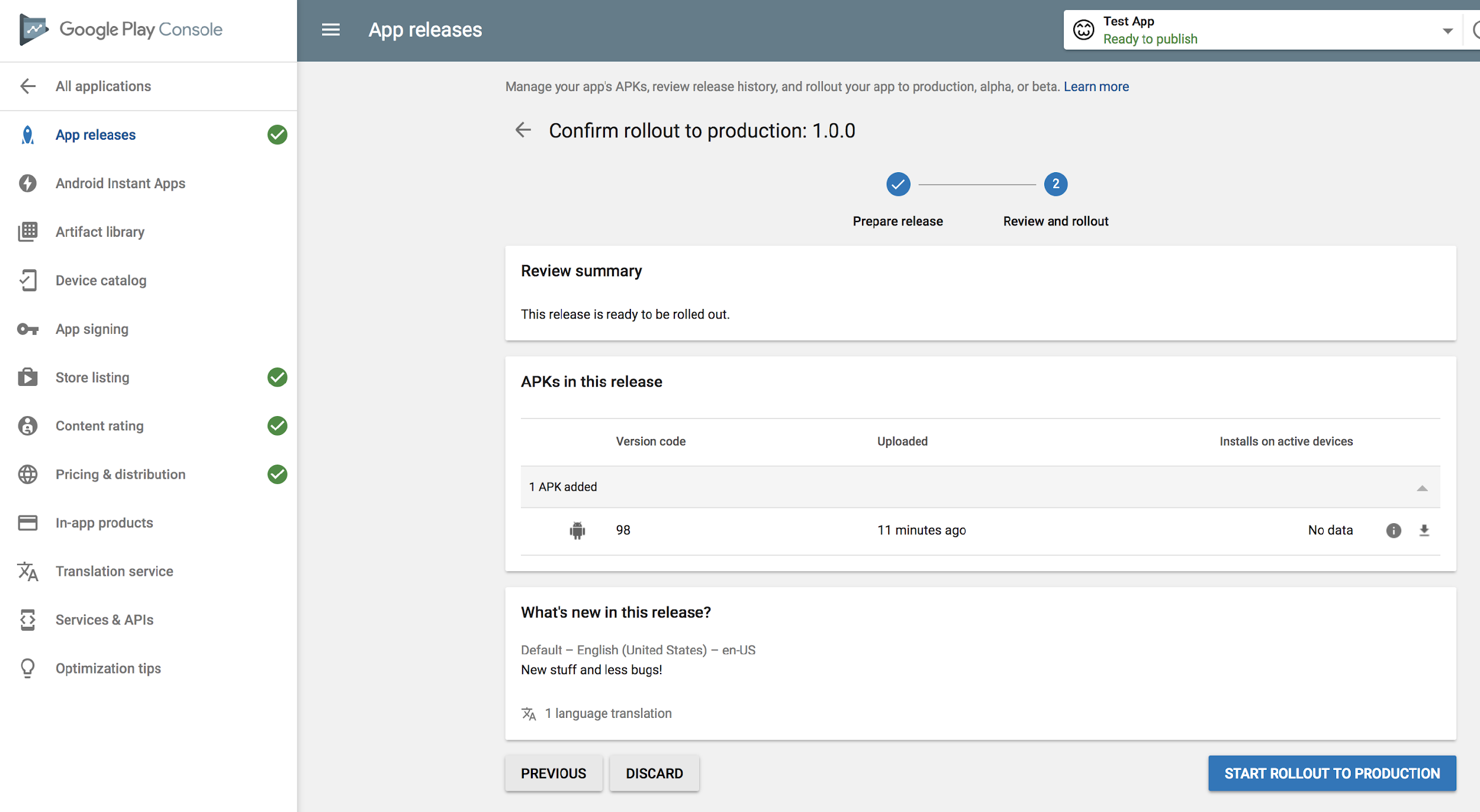


Figure 1: Roll out a release

1. Creating a release

A release is a combination of one or more build artifacts that you will prepare to roll out an app or an app update. You can create a release on three different testing tracks or to production:

1. **Open testing:** Open testing releases are available to testers on Google Play. Users can join tests from your store listing.
2. **Closed testing:** Closed testing releases are available to a limited number of testers that you choose, who can test a pre-release version of your app and submit feedback.
3. **Internal testing:** Internal testing releases are available to up to 100 testers that you choose.
4. **Production:** Production releases are available to all Google Play users in your chosen countries.
5. Preparing your app's release
6. Follow the on-screen instructions to prepare your release:
   * + If you want Google to protect your app’s app signing key, opt-in for app signing by Google Play.
     + Add your app bundles or APKs.
     + Name your release.
     + Enter release notes.
     + For more information on any of these fields, select the matching section heading under “Prepare” below.
7. To save any changes you make to your release, select Save.
8. When you have finished preparing your release, select Review release.
9. Reviewing and rolling out your release

Prerequisite: Before you can roll out your release, make sure you have set up your app's store listing, prepared your app for review on the App content page, and set up your app’s prices.

Once you are ready to roll out your app:

1. Open Play Console and go to the Releases Overview page.
2. Next to the release you want to roll out, select the right arrow to open the Release details page.
3. In the “Release overview” section, select View release dashboard.
4. Select the Releases tab, then Edit.
5. Review your draft release, make any necessary additional changes, and select Save.
6. Select Review release. You will be taken to the "Review and release" screen, where you can make sure there are not any issues with your release before rolling out to users.
7. If you see the heading “Errors summary” at the top of the page, click Show more to review the details and resolve any problems. Note that; when available, you can also view the recommended or required resolution. You cannot publish your app until errors have been resolved. If you only have warnings, minor issues, or a combination of the two, then you can still publish your app, but we recommend addressing them before publishing.
8. If you are updating an existing app, select a rollout percentage. If you are rolling out your first release, you will not see the option to select a rollout percentage.
9. Select Start rollout. If you are rolling out your app's first release on production, clicking Start rollout to production will also publish your app to all Google Play users in the countries you selected.
10. Reviewing release details

Once you have created a release, you will see the following information for the latest app release you rolled out to each track in a table under “Latest releases” on your Releases overview page.

1. **Release:** Name to identify release in the Play Console only, such as an internal code name or build version.
2. **Track:** The track that the release has been rolled out to.
3. **Release status:** The current status of your release.
4. **Last updated:** A date and timestamp indicating the last rollout event for your release.
5. **Countries / regions:** The number of countries / regions the last rollout of your release is available to.

You can view more in-depth information by selecting the right arrow to open the release’s Release details page, which includes:

1. **Release overview:** A set of metrics relating to your app's number of installs and updates, performance issues, and rating measured against previous releases.
2. **App bundles and APKs:** A list of new, retained, and deactivated app bundles and APKs associated with your release.
3. **Release notes:** A list of previous release notes.
4. **Rollout history:** A timeline that shows timestamps for when your app's release was halted, resumed, or served to a new percentage of users.
5. Tracking and managing releases on your Releases overview page

If you have rolled out multiple releases, the Releases overview page (under “Release” in the left menu) helps you stay on top of all releases in one place. It is a single location where you can monitor the availability of your apps across different tracks, view the countries/regions it’s available in, and select individual releases to view their specific details.

# **CORE APPLICATION QUALITY GUIDELINES**

Android users expect high-quality apps. App quality directly influences the long-term success of your app—in terms of installs, user rating and reviews, engagement, and user retention. All Android apps should meet these criteria. Before publishing your apps, test them against these criteria to ensure that they function well on many devices, meets Android standards for navigation and design, and are prepared for promotional opportunities in the Google Play store.

## Visual design and user interaction

These criteria ensure that your app provides standard Android visual design and interaction patterns where appropriate, for a consistent and intuitive user experience.

Table 2: Visual design and user interaction criteria.

|  |  |
| --- | --- |
| Area | Description |
| Standard design | The app follows Android Design guidelines and uses common UI patterns and icons:   * The app does not redefine the expected function of a system icon (such as the Back button). * The app does not replace a system icon with a completely different icon if it triggers the standard UI behaviour. * If the app provides a customized version of a standard system icon, the icon strongly resembles the system icon and triggers the standard system behaviour. * The app does not redefine or misuse Android UI patterns, such that icons or behaviours could be misleading or confusing to users. |
| Navigation | * The app supports standard system Back button navigation and does not make use of any custom, on-screen "Back button" prompts. * All dialogs are dismissible using the Back button. * Pressing the Home button at any point navigates to the Home screen of the device. |
| Notifications | Notifications follow Android Design guidelines. In particular:   * Multiple notifications are stacked into a single notification object, where possible. * Notifications are persistent only if related to ongoing events (such as music playback or a phone call). * Notifications do not contain advertising or content unrelated to the core function of the app, unless the user has opted in.   The app uses notifications only to:   * Indicate a change in context relating to the user personally (such as an incoming message), or * Expose information/controls relating to an ongoing event (such as music playback or a phone call). |

## Functionality

These criteria ensure that your app provides the expected functional behavior, with the appropriate level of permissions.

Table 3: Functionality criteria

|  |  |
| --- | --- |
| Area | Description |
| Permissions | * The app requests only the absolute minimum permissions that it needs to support core functionality. * The app does not request permissions to access sensitive data (such as Contacts or the System Log) or services that can cost the user money (such as the Dialer or SMS), unless related to a core capability of the app. |
| Install location | * The app functions normally when installed on SD card (if supported by app). * Supporting installation to SD card is recommended for most large apps (10MB+). |
| Audio | * Audio does not play when the screen is off, unless this is a core feature (for example, the app is a music player). * Audio does not play behind the lock screen unless this is a core feature. * Audio does not play on the home screen or over another app unless this is a core feature. * Audio resumes when the app returns to the foreground or indicates to the user that playback is in a paused state. |
| UI and Graphics | * The app supports both landscape and portrait orientations (if possible). Orientations expose largely the same features and actions and preserve functional parity. Minor changes in content or views are acceptable. * The app uses the whole screen in both orientations and does not letterbox to account for orientation changes. Minor letterboxing to compensate for small variations in screen geometry is acceptable. * The app correctly handles rapid transitions between display orientations without rendering problems. |
| User/app state | The app should not leave any services running when the app is in the background, unless related to a core capability of the app. For example, the app should not leave services running to maintain a network connection for notifications, to maintain a Bluetooth connection, or to keep the GPS powered-on.  The app correctly preserves and restores user or app state. The app preserves user or app state when leaving the foreground and prevents accidental data loss due to back-navigation and other state changes. When returning to the foreground, the app must restore the preserved state and any significant stateful transaction that was pending, such as changes to editable fields, game progress, menus, videos, and other sections of the app or game.   * When the app is resumed from the Recents app switcher, the app returns the user to the exact state in which it was last used. * When the app is resumed after the device wakes from sleep (locked) state, the app returns the user to the exact state in which it was last used. * When the app is relaunched from Home or All Apps, the app restores the app state as closely as possible to the previous state. * On Back keypresses, the app gives the user the option of saving any app or user state that would otherwise be lost on back-navigation. |

## Compatibility, performance and stability

These criteria ensure that apps provide the compatibility, performance, stability, and responsiveness expected by users.

Table 4: Compatibility, performance and stability criteria

|  |  |
| --- | --- |
| Area | Description |
| Stability | The app does not crash, force close, freeze, or otherwise function abnormally on any targeted device. |
| Performance | The app loads quickly or provides onscreen feedback to the user (a progress indicator or similar cue) if the app takes longer than two seconds to load. |
| SDK | * The app runs on the latest public version of the Android platform without crashing or loss of core function. * The app targets the latest SDK by setting the targetSdk value to minimize the use of any platform-provided compatibility fallbacks. * The app is built with the latest SDK by setting the compileSdk value. |
| Battery | The app supports power management features in Android 6.0+ (Doze and App Standby) properly. In the case where core functionality is disrupted by power management, only qualified apps may request an exemption. |
| Media | Music and video playback is smooth, without crackle, stutter, or other artifacts, during normal app use and load. |
| Visual quality | The app displays graphics, text, images, and other UI elements without noticeable distortion, blurring, or pixelation.   * The app provides high-quality graphics for all targeted screen sizes and form factors. * No aliasing at the edges of menus, buttons, and other UI elements is visible.   The app displays text and text blocks in an acceptable manner.   * Composition is acceptable in all supported form factors. * No cut-off letters or words are visible. * No improper word wraps within buttons or icons are visible. * Sufficient spacing between text and surrounding elements. |

## Security

These criteria ensure that apps handle user data and personal information safely. In addition to this checklist, applications published on the Google Play Store must also follow the User Data policies to protect users' privacy.

Table 5: Security criteria

|  |  |
| --- | --- |
| Area | Description |
| Data | * All private data is stored in the app's internal storage. * All data from external storage is verified before being accessed. * No personal or sensitive user data is logged to the system or app-specific log. |
| App Components | Only application components that share data with other apps, or components that should be invoked by other apps, are exported. This includes activities, services, broadcast receivers, and especially content providers. Always set the android:exported attribute explicitly, regardless of whether or not you export any of your application's components.  All application components that share content with other apps define (and enforce) appropriate permissions. This includes activities, services, broadcast receivers, and especially content providers.  All content providers that share content between your apps use android:protectionLevel="signature". |
| Networking | * All network traffic is sent over SSL. * Application declares a network security configuration. * If the application uses Google Play services, the security provider is initialized at application startup. |
| Libraries | All libraries, SDKs, and dependencies are up to date. |
| WebViews | JavaScript is disabled in all WebViews (unless required).  WebViews only load allowlisted content if possible.  WebViews do not use addJavaScriptInterface() with untrusted content. On Android M and above, HTML message channels can be used instead. |
| Execution | The app does not dynamically load code from outside the app's APK. |
| Cryptography | * The app uses strong, platform-provided cryptographic algorithms and does not implement custom algorithms. * The app uses a properly secure random number generator, in particular to initialize cryptographic keys. |

## Google Play

These criteria ensure that your apps are ready to publish on Google Play.

Table 6: Google Play criteria

|  |  |
| --- | --- |
| Area | Description |
| Policies | * The app strictly adheres to the terms of the Google Play Developer Content Policy and does not offer inappropriate content, does not use the intellectual property or brand of others, and so on. * The app maturity level is set appropriately, based on the Content Rating Guidelines. * The app supports power management features in Android 6.0+ (Doze and App Standby) properly. In the case where core functionality is disrupted by power management, only qualified apps may request an exemption. See Support for other use cases in Doze and App Standby. |
| App Details Page | * The app listing includes a high-quality feature graphic. * The feature graphic does not contain device images, screenshots, or small text that will be illegible when scaled down and displayed on the smallest screen size that your app is targeting. * The feature graphic does not resemble an advertisement. * The app’s screenshots and videos do not show or reference non-Android devices. * The app’s screenshots or videos do not represent the content and experience of your app in a misleading way. |
| User Support | Common user-reported bugs in the Reviews tab of the Google Play page are addressed if they are reproducible and occur on many different devices. If a bug occurs on only a few devices, you should still address it if those devices are particularly popular or new. |

**QUESTIONS:**

1. Explain the advantages of uploading native apps to play store.

**Answer:**

If you upload your app as an Android App Bundle, the Play Console automatically generates split APKs and multi-APKs for all device configurations your app supports

**REFERENCE:**

* <https://support.google.com/googleplay/android-developer/answer/9859348#rollout&zippy=%2Capp-signing-by-google-play%2Cwhats-new-in-this-release>
* <https://developer.android.com/studio/publish/upload-bundle>
* <https://developer.android.com/distribute/best-practices/launch/launch-checklist>