Scenario 3:
You're deep in the middle of making what will surely be the greatest ARCore based 3D sticker app the world has ever seen. At the morning stand up you hear that the sound designer has made a pass updating the existing UI sound files while the art team have submitted 10 new 3D sticker prefabs bringing the total to 250! Excited to see the new changes you rush back to your desk and update. After making a new build you notice the size of the .apk has doubled in size, far too big for the changes mentioned!

- Questions:
  - How do you go about figuring out the cause of the increase?
  - How will you inform the team of your findings?
  - What steps do you take to reduce the size?

After a big change where we have 10 new prefabs bringing the total to 250 and having updated sound effects could increase the project size to an uncomfortable level this is what I would do to help the problem. I would separate the issue in sizes, what happens only after the audio file change not including the prefab additions and what happens if I only build with the prefab additions and not the audio. I would also have a control size meaning having a build before any of these changes were included so I can see a rate of how my project build and size is increasing. With these 2 separate build sizes compared to my combined build size and my original build size you can see the one that has the biggest impact on build size. While building I would save the editor log for each build to see each file size per asset that is being built in the build (by going to Window > General > Console > Editor Log). I will gather this data usually in steps showing the reason and numbers behind the build process for each iteration. Each iteration I would most likely make a google doc with a graph showing all 4 changes in build size (original build, build with new audio, build with new prefabs, build with everything). Each change would also have a list of the top files that create the biggest file size increase (by sorting through the log file and try to add up the differences from the new files in build size from new - original size) This gives clear path of what changed in the project and what are the primary causes of such build sizes so project stakeholders can make an informed decision on what to change next for the next build to have a size that's smaller. If I were to try to make the build smaller I would have 2 separate scenarios, 1 for the audio changes, 1 for the prefab additions. Since the audio isn't an addition I could possibly talk to the sound designer to maybe reduce the size of the file or change the file type that has better compression support. Otherwise I could maybe look into the import settings for the new audio files. While looking at the import settings there are settings that will change the size. One being the "Audio Format" and "Load Type". I would change these based on what type of audio file I'm changing, if it's a small sound file I would tend to decrease the size more that might decrease the quality since the user has less interaction with that sound while looking at the lifetime of the application. If the sound is more like a music file, I would likely less modifications due to the fact that the user tends to hear more of the music over the lifetime of the app. I would adjust these so that sound effects heard more often would have less modifications. All modifications will likely be shown to the sound designer so it passes their approval. For the additional prefabs it sounds intimidating to have 250 similar prefabs in the project before build. I would try to analyze what are the major differences between the prefabs and what could be combined into a handful of starting prefab templates or just prefab variants.

Since the changes were made by the art team I would ask to see what was the reason behind adding them in the first place and seeing what was not already in the project that they needed to have. If it comes down to different variations in color or size or any other minor visual modification to the original prefabI would most likely bring up a system design that allows data to pulled in from a server or database that gives instructions to the game to construct the prefabs at runtime so we can take in a color mod as an example and apply this to the base prefab. Almost like a style sheet with names and values for each type of prefab that could be declared in json format so that the change is not affecting build size but would have to include a pipeline structure to get this working. When using a cloud based system like this, the artist could just change the outcome of those prefab visual styles without having to change the project files in any way. As a plus I could add a tool, separate from the main build that in the editor, allows the artist to see this change in a tool scene so they can preview the change in visual style. A step higher would be a straight asset bundle tool that would allow the artist still work in unity and modify these prefab variants natively, but instead have them built as an asset bundle and have the bundle uploading to a cloud bucket like GCS or AWS which would also have the need for a end to end system that pulls these assets at run time and has a caching system that allows assets to be loaded faster that downloading them. These assets would not end up in the project at build time and would be ignored using the Assets/AssetBundle workflow. Picking up one of these systems has risk and engineering overhead that must be accounted for when trying to decide the best change to avert build size problems. I would prefer to do the first solution as it has less overhead and involves the least amount of risk into the asset pipeline. You would also need to train the artist on the new process to include an asset or asset variant in the project correctly.

Recourses:
https://docs.unity3d.com/352/Documentation/Manual/AudioFiles.html
https://docs.unity3d.com/Manual/ReducingFilesize.html
https://docs.unity3d.com/Manual/ReducingFilesize.html
Googles: “unity audio files make project bigger” to try and see if I was in the shoes of implementing audio and ran into a problem if there any solutions people mentioned online somewhere like the unity forum