Event Echo: AI Usage Statement 12/2

Overview

AI has been part of our application's development process. We have used it for generating screen wireframes as well as assisting with project code directly, such as generating, debugging, and troubleshooting. In many ways, it is quite helpful and accelerates the development process and clears up confusion, but it also has limitations and almost all of its suggestions require thorough evaluation and restructuring.

Tools Used, Where & How

- **Figma**
  - We used Figma for screen wireframes, most of which were created manually using the drag-and-drop system but a couple were generated by Figma's AI, which we used as a skeleton/mockup to base some of our screen designs off of.
- **ChatGPT**
  - We used ChatGPT for generating, debugging, and troubleshooting code. Almost all of its suggestions require thorough evaluation and restructuring.
- **Firebase Gemini Assistant**
  - We used firebase gemini assistant to evaluate which firebase products best suited our project and how to best integrate them with our app. Although it provided good suggestions, the official firebase Kotlin documentation was much more helpful.
- **Gemini**
  - Like ChatGPT, we used Gemini (Android Studio's built-in version and Google's web version) for generating, debugging, and troubleshooting code. Almost all of its suggestions require thorough evaluation and restructuring.

Example Prompts

- **General Examples**
  - How can I design a system that uses both live location and custom google maps navigation so that we don't fetch from ticketmaster api too often?
  - Would a high accuracy or low accuracy priority be better?
  - How do I interface with firestore within my kotlin project to upload data to our database?
  - Generate a simple wireframe for the profile page with generic fields for username, profile picture, and bio.

- **Detailed Examples**
  - Can you adapt this to a new composable so that it has a card on each side like this (Composable for a list of events & image of a desired Event Grid also provided as context)
    - **Response & Corrections Required:** ChatGPT responds with a new composable, but it has some extra whitespace, it's not fully integrated

        with the theme, and doesn't gel with our viewModel and data, so these need to be resolved manually.
    - Which color in the theme is the card taking after?
        - **Response & Corrections Required:** ChatGPT responds by saying that a Card uses MaterialTheme.colorScheme.surface, which can be used to figure out why the card displays as a certain color, eventually deciding to use a Surface instead.

Corrections Required (General)

- **Styling Fixes** - The screen wireframes generated very bare bones skeletons for our screens without the necessary styling our app required with our own custom theme.
- **Debugging Context** - Utilizing AI to help debug is useful but also difficult to do reliably since the context of our codebase is so large. It is usually faster to debug manually, unless the bug fix is simple to detect.
- **App-Specific Instructions -** Our app has some pretty specific functionality that AI does not quite understand. Thus, we have to adapt the output from AI to our own specific needs for what we want the app to do. For example, AI is good for showing the general structure for uploading data to firestore, but we had to adapt it to follow our specific schemas and app rules.

Analysis of Helpfulness and Limitations

These AI tools have been helpful as they accelerate the design and development process. For the UI, for example, the wireframes we generated using Figma guided how we styled certain pages. They have also been helpful for the project code because they can create samples of certain elements, especially considering these LLMs are aware of Jetpack Compose and its capabilities and structuring, which is helpful for generation but also debugging. If you are very specific about prompts and context, AI can help clear up confusion as long as its responses are critically evaluated and/or fact-checked.

However, these AI tools also come with many limitations, the most notable being that we have ideas in our head of how our codebase is structured and how we want certain elements to look and integrate with the rest of our project. So, any code generated using AI has to be adapted to gel with the rest of our project, like certain state variables or a ViewModel/NavController. These AIs also can't see what they're creating, so almost all of the time their responses, if visual elements are at play, will display strangely or not integrate with a theme and need adaptation.

It is also worth noting that often, the contexts of these LLMs are outdated or contain outdated information, which is problematic when the Android SDK is constantly evolving. These LLMs sometimes use elements that don't work with our development environment.