

Article

Not peer-reviewed version

Leveraging ChatGPT in Android App Development: A Case Study on Supporting Novice Developers in Creating Successful Apps

[Mohamad Iyad Al-Khiami](#)^{*} and Martin Jaeger

Posted Date: 11 July 2023

doi: [10.20944/preprints202307.0660.v1](https://doi.org/10.20944/preprints202307.0660.v1)

Keywords: Chat Generative Pre-Trained Transformer (ChatGPT); Artificial Intelligence; Human-Computer Interaction; Human-AI Interaction Chatbot; App Development; Coding



Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Article

Leveraging ChatGPT in Android App Development: A Case Study on Supporting Novice Developers in Creating Successful Apps

Mohamad Iyad Al-Khiami ^{1,*} and Martin Jaeger ¹

¹ Civil Engineering Department, College of Engineering, Australian University, West Mishref, Kuwait

* Correspondence: Engineering Department, College of Engineering, Australian University, West Mishref
P.O. Box 1411, Kuwait Correspondence: m.alkhiami@au.edu.kw

Abstract: OpenAI has managed to turn 100 million heads in two months towards their new Language model tool (LM) ChatGPT. The third generation Generative Pretrained Transformer (GPT-3) has the capacity to tackle simple to complex and sophisticated problems, while providing reasoning behind its generated answers. ChatGPT can be used to increase productivity and improve efficiency and face challenging problems. Programming mobile applications is a challenging task that requires professional software engineers, skills and abilities to be developed. The following paper takes a case study approach to assess how novice app developers can use ChatGPT to generate Java scripts that will be used in Android studio to create a functional application. The results after, many iterations and ongoing conversations with ChatGPT managed to create an application for the anticipated function. Important insights have been drawn from the case study that could set the ground for any novice user seeking to create applications using Java scripting and Android Studio.

Keywords: Chat Generative Pre-trained transformer (ChatGPT); Artificial Intelligence; human-computer interaction; Human-AI Interaction Chatbot; App development; coding

1. Introduction

The world's traditional practices are gradually becoming outdated as newer technological advancements worldwide and within industries are emerging. The world is no longer the same and is forever changing. The integration of disruptive technologies into our daily lives is revolutionizing traditional practices. A major paradigm shift within these practices, caused by digital technologies, has already stepped into what is now called the digital era. Virtual reality, augmented reality and now a major advancement within artificial intelligence are only a few of the disruptive technologies emerging [1].

Information technologies has emerged and are integrated within our lives shifting the use of such technologies from an option to a compulsory component of our daily activities, thus achieving improved efficiency and productivity within all disciplines [1-3], renaming the current and upcoming generations as digital natives and past generations as digital immigrants [4].

Programming on the other hand, plays a vital role in the digital era we are living in, as most of these technologies involve computer languages and coding that can further be developed into the creation of mobile applications that provides individuals and communities with friendly and easy to use innovative solutions.

Though, learning programming, for novice users, to develop mobile applications can be very challenging, it involves learning computer languages such as Java, Python and C++ which would require a significant amount of time and effort to exit the experimental state and enter the state of productivity. in addition to understanding the different platforms and software, bugs and errors within the codes can be very challenging and difficult to interpret.

The emergence and advancement of digital technologies, mainly artificial intelligence (AI) and natural language processing (NLP) and advancements in Chatbots, known as intelligent agents, dates to the 1950s. Chatbots involves human-computer interactions trained based on simple techniques

such as pattern matching, artificial intelligence markup language, latent semantic analysis, and chat scripts [5]. Though, the techniques and training used to create AI has substantially been developed and improved, creating one of the most advanced Chatbots, known as Chat Generative Pre-trained transformer (ChatGPT). ChatGPT which was developed by OpenAI in 2020, has the ability to generate human like responses, caused by large language models (LLM), capable of responding to uncountable variations of problems [6,7].

The question remains, could ChatGPT aid novice developers create successful applications? The next section discusses what the literature has to say about the uses of ChatGPT in general and in developing codes for mobile applications in specific.

2. Literature Review

The literature review will first introduce ChatGPT, followed by previous and ongoing research on the applicability of ChatGPT as a technology in advancing overall practices and using it as a tool to generate codes towards a successful application.

2.1. *What is ChatGPT*

ChatGPT, a state of art natural language processing model that has been developed by an AI company known as OpenAI, based on GPT-3 [6–8] a successor to GPT and GPT-2 has managed to turn all heads towards it. The LLM model has been trained on more than 175 billion trainable parameters equivalent to 45 Terabytes (TB) of text data, reaching a cost of 12 million dollars [7,10]. According to Open AI ChatGPT serves as a successor to InstructGPT but with varying methods of AI training that uses Azure AI supercomputing infrastructure [10].

ChatGPT has managed to reach more than 100 million users only two months after its release, becoming the fastest growing platform in the world of internet [11]. The question remains, if such a disruptive technology does generate genuine and true replies, and if it will facilitate development of an Android smartphone application.

2.2. *Its Application and Limitations*

According to [9], ChatGPT can generate codes, stories and, poems in addition to text. It is described by [13] that ChatGPT is “Strikingly human-like. It understands text input and responds to it like a well-learned person. And it goes beyond generating language”. ChatGPT assistance includes problem-solving, reasoning, generating articles solving mathematical problems and beyond [9,13].

A study conducted by [14] discussed how Chatbots, mainly ChatGPT can be used to disrupt the traditional Google search for information to cancer relateFgd questions. Two basic and three complex questions were asked by the developer to assess its abilities. The findings showed that both Google's feature snippet and ChatGPT generated related answers to the questions. Though, ChatGPT generated responses minimized the likelihood of alarm in comparison to Google. The study also concluded that rewriting the exact same question to ChatGPT generates different responses, in addition, it does not provide with any sort of reference, thus, limiting the response to what it has been trained on up to 2021.

Another study related to the medical field was conducted by [15]. The study exposed ChatGPT to multiple choice medical examinations questions and recorded the results collected. ChatGPT managed to score between 42% and up to 64.4% depending on the question' difficulty. It was concluded by the author that ChatGPT current level corresponds to a third-year medical school student.

On the other hand, many researchers are investigating and questioning the capabilities and ability to use ChatGPT in education [6,16–18]. The studies indicate that ChatGPT has the potential to assist in many diverse aspects, including but not limited to an alternative search engine, academic writing, data interpretation, generating titles, and writing a research article draft. Its ethical usage and limitations remain in question. As mentioned previously by [14,15], ChatGPT is limited to the events up to 2021, it may also generate biased and sometimes fake information. It is worth noting that

the limitations collected by authors are not hidden by OpenAI, in fact, it is openly mentioned that ChatGPT has limited knowledge of world events after 2021, may occasionally produce harmful or biased content and generate incorrect information [10].

When it comes to the field of programming mobile applications using Java scripts and Android Studio, up to the author's knowledge, no testing was done by researchers. [19,20] discussed the potential of ChatGPT in generating codes that could be used for programming but did not mention any further details. Another study mentioned that ChatGPT managed to pass a programming test [2].

The closest was a study conducted by [21] that investigated the ability of ChatGPT in fixing bugs found in codes and suggest repairs. ChatGPT was found to fix 31 out of the 40 bugs tested. This is very promising as novice programmers can utilize ChatGPT to solve bugs within codes, but does ChatGPT generate codes according to specific needs and requirements?

3. Research Questions

As a result of the literature review, the following research questions (RQ) are inquired. RQ1 – Is ChatGPT capable of generating codes for programming mobile applications using Android Studio? RQ2 – Can novice programmers use ChatGPT as a disruptive tool to increase their learning curve and productivity levels? RQ3 – What sort of approach must users take when using ChatGPT to generate codes to avoid incorrect and unsuccessful attempts?

4. Methodology

When studying a specific case with the aim to derive generalizable insights, it needs to be clarified first if a single case study is sufficient, or if multiple cases need to be investigated. Each of the following five criteria justifies a single case study approach [22]:

1. The case is critical and allows testing a well formulated theory;
2. The case is unique or extreme;
3. The case is representative or typical;
4. The case allows analysis of a previously inaccessible phenomenon; and,
5. The case is longitudinal in nature (i.e. covering an extended period of time) and allows analysis at different points of time [22] (pp. 39–46).

The following aspects make the case considered here a unique case (criterion 2 above):

- Data related to experiences with Large Language Models (LLMs), such as ChatGPT, when coding android applications is not sufficiently available to allow a representative quantitative study; and,
- To the best of the authors' knowledge, experiences of using ChatGPT for Android App development have so far not been investigated.

In addition, anecdotal evidence shows that ChatGPT is used not only by students, in virtually all areas of life. Therefore, the case is also considered typical (criterion 3 above) since students may use it also for coding activities within a learning context. Furthermore, this case also allows the analysis of a previously inaccessible phenomenon (criterion 4 above) in that ChatGPT was previously not available. Hence, a single case study is considered appropriate.

4.1. Case Identification

The case considered here was identified within the framework of a graduation project (i.e. capstone project) of undergraduate civil engineering students. The scope of the graduation project included the development of a simple smartphone application which would allow the user to enter properties of a specific engineering material with the goal to receive a prediction concerning the usability of the material. Since students were not required to develop the application themselves (focus of their project was on material testing and analysis), they were permitted to outsource this task. However, developing such an application was beyond the budget of this graduation project, which led to the interesting question if ChatGPT can provide effective and efficient support.

The smartphone application is intended to function as per the following. First, it should start with a simple user interface that includes a drop-down menu in which the user can select the following options:

1. Disclaimer
 - Users that select the disclaimer should receive a disclaimer message.
2. Property Based Calculator (main function)
 - Users that select the Property Based Calculator should be promoted to add the properties requested to add three variables.
3. Copyright
 - Users that select the copyright option should receive the copyright message.

The application should use a specific formula that was the result of the civil engineering project and provide an output stating the estimated number of uses as a message.

Even though the application may sound uncomplex and easy for intermediate and professional programmers to develop, the developer of the smartphone application (one of the authors of this study) had no experience in programming and coding in any sort of computer language.

4.2. Data collection and data analysis

Comparing with the six principles of case study research [22] (pp. 83–108), the data collection and analysis approach is as follows:

1. Analyze “documentation”: Not applicable for this case, since no documentation related to coding of Android application using ChatGPT was identified.
2. Analyze “interviews and discussions”: All “discussion” with ChatGPT was analyzed regarding the usefulness of responses.
3. Analyze “physical artefacts”: ChatGPT generated code snippets (here considered as “artefacts”) were immediately transformed into Android MainActivity and tested. The output generated by the application was observed and analyzed. The analysis included one quantitative aspect and one qualitative aspect. The qualitative aspect refers to the difference between manually calculated values and application calculated values, whereas the qualitative aspect is concerned with the difference between expected application design and generated application design.
4. Analyze “direct observations”: Not applicable for this case, since only the results of coding (principle #5) and the developer carrying out coding activities (principle #4) can be observed.

This means, two of the four case study principles identified by [22] can and will be applied. Steps number two, and three of the data collection and analysis process will be carried out over the complete time of the case (i.e. longitudinal analysis) in order to allow the identification of pattern over time.

4.3. Analysis strategy

Further following [22] (pp. 109), the three general analysis strategies of cases are

1. Relying on theoretical propositions: These are derived from initial literature review, identification of issues and initial precursors for the case study. Final explanations will be based on comparing findings with these initial propositions. Since no body of knowledge could be identified in literature regarding the research question here, this study is based on the following two propositions:
 - The developer can critically observe himself during the time of coding; and,
 - “Archival records” allow a comparison with the experience of the developer.
2. Considering rival explanations: Alternative explanations might have caused the same phenomenon and need to be considered when discussing the results. For example, a code resulting in wrong values could be related to wrong information provided by ChatGPT, but it also could be related to a coding mistake although correct information was provided by ChatGPT.

3. Developing a descriptive framework (case description): This is principally used when the two proceeding strategies are not applicable [22] (pp. 109) and, therefore, is not further considered here.

Case study analysis may be based on one unit of analysis, or more than one unit of analysis [22]. A unit of analysis is a message or a message component which can be clearly identified [23] (pp. 71). Here four units of analysis are used as shown in Table 1.

Table 1. Units of analysis

#	Data analysis approach	Unit of analysis
1	"interviews and discussions"	Number of immediately useful ChatGPT response
		Number of not immediately useful ChatGPT response
2	"physical artefacts"	Frequency of wrong results generation Frequency of unmet expectation regarding application design

Analysis of case study data consists of examining, categorizing, tabulating testing, and otherwise recombining quantitative and qualitative data. The use of analytic software tools to facilitate coding and categorizing data is recommended [22] (pp. 109).

Based on a logic model [22] (pp. 127), the researcher stipulates a chain of events over time that link cause and effect. This means the outcome of an earlier stage becomes the causal event for the next stage. Predicted outcomes are then tested against evidence found in the data. Here, the usefulness of ChatGPT responses, observed developer emotions and the quality of the application (i.e. correctness of results and meeting expectations) present all parts of the logic model and are expected to cause effects on the following stages. This does not ignore potential direct effects of stages before the immediately preceding stage.

After summarizing the methodological basis of this study, the following section reports and discusses the results.

5. Results and Discussion

As part of an ongoing civil engineering project, the need to formulate a mobile application that would represent the findings of the research was a necessity. First a brief of the application function is described.

5.1. Defining the platform

Determining the platform to be used to serve the function intended was a challenge. ChatGPT was asked to recommend a platform to be used to create mobile applications that uses Java language.

The following shows the typical question from the developer (Q1) and response by ChatGPT (R1):

Developer: "What Platform for mobile app development do you recommend using that uses java?"

ChatGPT: "If you want to use Java for mobile app development, one of the most popular and widely used platforms is Android. Android is an open-source platform developed by Google and based on the Linux operating system. It supports Java for app development, as well as Kotlin, a newer programming language designed to be more concise and expressive than Java."

Android Studio is the official integrated development environment (IDE) for Android app development, and it is recommended for developing apps using Java. Android Studio includes a wide range of tools and features to help you build high-quality apps, including an XML layout editor, a code editor with intelligent code completion, a debugger, and a variety of testing and profiling tools.

In addition to Android Studio, there are several other tools and frameworks available for Java-based mobile app development, such as Cordova, PhoneGap, and Ionic. These platforms allow you

to build cross-platform mobile apps that can run on both Android and iOS devices using a single codebase. However, they may not offer the same level of performance or integration with the device hardware as native Android apps developed using Java and Android Studio.”

From the response above, even though the question was concise, with an expected outcome of a single platform recommendation, ChatGPT provided multiple different options including, Android Studio, Cordova, PhoneGap and Ionic. According to the unit of analysis mentioned in Table 1, the response is considered as “not immediately useful ChatGPT response”.

To further investigate the response of ChatGPT, the question (Q1) was asked again three times, with the exact same wording in the same chat. The responses (R2), (R3), and (R4) were similar with (R1). Though the third time (Q1) was asked, response (R4) was incomplete. In all responses ‘not immediately useful ChatGPT response’ was recorded.

Furthermore, (Q1) was asked again but in three different chats. In all three chats the responses (1R1), (2R1) and (3R1) generated were different, where the first number in (1R1), (2R1) and (3R1) refers to three different chats, and the number following the letter refers to the first response in that chat. ChatGPT recommended in the three new chats 5 new platforms:

1. Eclipse
2. IntelliJ IDEA
3. React Native
4. Xamarin
5. Netbeans

Even though ChatGPT recommended Android Studio as the first choice in all four chats, “not immediately useful ChatGPT response” is also recorded for the chats as the question intended was to precisely recommend a single platform that uses Java, rather than multiple platforms that uses different computer languages including Java.

To establish a trend on the responses of ChatGPT and frequency of the words used, an online word cloud was used where all the responses of ChatGPT (R1,1R1,2R1,3R1) were inserted. Figure 1 shows the results of the word cloud.



Figure 1. Word cloud generated based on all responses to (Q1)

Based on the font height of words in Figure 1 it can be perceived that the most common words ChatGPT used were Java, Android, and app as a response to Q1.

With the aim to receive a more useful response, the question's wording was then changed, and the following question (Q2) was asked, and response (R2) was given:

Developer : “can you recommend a single platform for mobile app development that uses Java only?”

ChatGPT: “Yes, there are several platforms for mobile app development that use Java as the programming language. One such platform is Android Studio, which is the official integrated development environment (IDE) for developing Android apps. Android Studio is based on the Java programming language and offers a range of tools and features to help developers create high-quality apps for Android devices.

Another option is the Codename One platform, which allows developers to write Java code that can be compiled to native code for multiple platforms, including iOS, Android, and Windows. Codename One also provides a range of tools and libraries to help developers build feature-rich mobile apps with Java.

Both Android Studio and Codename One are popular choices for mobile app development with Java, so it's worth considering which platform best fits your specific needs and preferences.”

Yet the response still provided two options, that is Android Studio and Codename which was not mentioned in any of the responses to (Q1). Again, a “not immediately useful ChatGPT response” was recorded. Finally, question (Q3) was changed to a command with the aim to receive a straightforward and immediately useful response. Question (Q3) and response (R3) were as follows:

Developer: “Recommend a single platform for mobile applications that uses java!”

ChatGPT: “One platform for mobile applications that uses Java is Android. Android is a mobile operating system based on a modified version of the Linux kernel and is designed primarily for touchscreen mobile devices such as smartphones and tablets. Android applications are typically written in Java and are compiled to bytecode that runs on the Dalvik virtual machine, which is specifically designed for mobile devices. Android also provides a rich set of APIs and tools for developing applications, including an integrated development environment (IDE) called Android Studio. Android is a popular platform for mobile application development, with a large user base and a vast array of third-party libraries and tools available to developers. It also provides a flexible and customizable platform for building applications that can run on a wide range of devices.”

Note that (Q3) used the word “Recommend” at the beginning of the question rather than “Can you” resulting in a single recommendation by ChatGPT as per (R3) response. Several other variations of (Q3) were asked including a verb such as “mention” “Provide” and “Give”, all in which resulted in a single platform recommendation. This important finding indicates that users should start with a verb in the question without the need of etiquette in asking, since this is a machine, it does not distinguish between asking kindly or not. Starting with “can you?” resulted in multiple platform recommendations. The results are that all responses for Q3 were recorded as a “immediately useful ChatGPT response”.

Although responses generated by ChatGPT that mentioned more than one platform and provided an extended description and reasoning behind the recommendations were considered as not immediately useful ChatGPT responses, it still provided insights and further understanding of the software available in the market.

To validate the recommendation of ChatGPT, G2, a website used for reviewing software [24], was checked. Table 2 shows the ratings of the software platforms.

Table 2. Software ratings based on G2 (source: [24])

Platform	Ease of use	Quality of support	Ease of setup
Android studio	8.6	8.4	7.8
Cordova	8	6.9	7.6
Ionic	8.9	8.2	8.5
PhoneGap	-	-	-
Eclipse	8.2	7.8	8.1
IntelliJ IDEA	8.7	8.5	8.6
React Native	8.2	8.1	8.4
Xamarin	7.7	7.6	7.6

Netbeans	8.5	7.8	8.4
----------	-----	-----	-----

According to [24], and based on the ratings of users only, the results show that Ionic is the best platform for mobile apps, followed by Eclipse, IntelliJ IDEA and React Native, almost on the same level as Android Studio. An important finding to document is PhoneGap, a platform that was also mentioned by ChatGPT, is outdated and is no longer being used. This corresponds to the first physical artefact referred to in Table 1 as a “wrong result generation”. Also, Ionic and Cordova only received 139 and 52 reviews respectively in [24], whereas Eclipse and IntelliJ IDEA received more than 2000 reviews, making Android Studio the most suitable platform as it received more than 500 reviews. The findings based on [24] are in line with ChatGPT most generated response, that is, Android Studio.

The choice of moving forward with Android Studio was made and the software was then installed.

The previous dialogue can be summarized as follows. When asking the same question with the exact same wordings in the same chat, ChatGPT generated the same response three times and once incomplete. On the other hand, asking the same question in different chats generated different responses all three times. Rewording the question to a command, rather than a question, resulted in more concise and direct responses aligned with the needs of the developer.

5.2. Creating the application

Initially, some YouTube videos [25–27] were reviewed by the developer to familiarize himself with the platform user interface which seemed overwhelming to begin with, how to add a logo and how to change the icon of the application. The YouTube videos involved identifying the main features of the platform, understanding what layouts are, how to add images, where to place the code of the main activity and setting up the emulator to test the code.

Following watching the videos, a description of the intended app function was sent to ChatGPT requesting a code that can be used.

The following shows the request by the developer (Q4) and the response by ChatGPT (R4):

Developer: “I want to prepare a script for a smartphone application in Android Studio. The script shall describe how to program a material reusability calculator. The application should start with the title "Material Calculator". Then it should show the logo of the "University". It should have a drop-down menu with Disclaimer, Calculator and copyright. Clicking on the Disclaimer should present: "Disclaimer message" Clicking on Calculator shall open a dialogue asking for the following input: input 1, input 2 and input 3. input 2 shall be entered in kg/m3. Based on the following equation, the number of uses U shall be displayed based on a formula that uses the three inputs . The result of U shall follow the statement "Estimated max. number of uses:" If any input field was not filled, the following statement shall be displayed: "Input missing" Clicking on copyright shall display "Copyright: University "".

ChatGPT: “Here is a possible script for a material reusability calculator Android application in Android Studio:”

The script provided by ChatGPT in response to Q4, R4 can be found in Appendix A, Table A1.

Notice that the first response (R4) by ChatGPT was incomplete, corresponding to a wrong result generation. The question (Q4) was then asked three more times in three different chats, resulting in responses (1R4), (2R4) and (3R4), as an attempt to generate a complete code for the required purpose. (1R4) and (2R4) were also incomplete while testing (3R4) resulted in an immediate error message.

In (Q4) a query like follow up question on the reason why ChatGPT did not complete the code, was asked rather than a question starting with a command to assess if ChatGPT would admit its mistake:

Developer : “Why did you stop?”

ChatGPT: “Apologies, it seems that my response got cut off. Here is the rest of the script for the Material Calculator:”

The script provided by ChatGPT in response to Q4 follow up, R4 follow up can be found in Appendix A, Table A2.

ChatGPT response continues beyond the code stating the follow "You will need to create a new Android Studio project and replace the MainActivity.java file with this script. The script defines the UI elements (EditText, Spinner, Button, TextView) and sets up the calculate Button to perform the necessary calculations based on the inputs provided by the user. Please note that the formula used to calculate the estimated max. number of uses is just an example, and you may need to adjust it based on your specific requirements. Additionally, you will need to create the UI layout (XML) file for the activity, which should include the necessary UI elements and the "Material Calculator" title, "University" logo, and drop-down menu with "Disclaimer", "Calculator", and "Copyright" options."

What is interesting is that ChatGPT replied with an apology, admitting the mistake it had made. When testing the code in Android Studio, it revealed errors, and the application did not run. Since the developer is only a novice programmer, the first impression was that the code is completely correct, and the main issue revolves around placing the code in the software. However, all responses R4,1R4,2R4,3R4 were not useful. The approach of requesting the complete code all at once seemed not convenient, neither for the developer nor for ChatGPT.

As a result, a different approach was applied in a new chat, requesting ChatGPT (Q5) to provide a step-by-step guide (R5) on how to insert the code mentioned previously.

Developer: "Provide a step-by-step guide for a novice developer to prepare a mobile application on Android Studio. The script shall describe how to program a material reusability calculator. The application should start with the title "Material Calculator". Then it should show the logo of the "University". It should have a drop-down menu with Disclaimer, Calculator and copyright. Clicking on the Disclaimer should present: "Disclaimer message" Clicking on Calculator shall open a dialogue asking for the following input: input 1, input 2 and input 3. input 2 shall be entered in kg/m3. Based on the following equation, the number of uses U shall be displayed based on a formula that uses the three inputs . The result of U shall follow the statement "Estimated max. number of uses:" If any input field was not filled, the following statement shall be displayed: "Input missing" Clicking on copyright shall display "Copyright: University ."

ChatGPT: "Sure, here's a step-by-step guide to creating a Material Reusability Calculator mobile application on Android Studio:

1. Open Android Studio and create a new project.
2. Select "Empty Activity" as the project template.
3. Give your project a name and package name.
4. Click "Finish" to create the project.
5. Open the activity_main.xml file, located in the res/layout folder.
6. Add the following code to create the layout for the application:"

Step 6 in ChatGPT response included the code to create the layout for the application. The complete code can be found in Appendix A, Table A3.

Question (Q5) was then asked two more times in two different chats (1R5), (2R5) to ensure consistency and validity of the code generated by ChatGPT before testing it in Android Studio. Interestingly (1R5) provided five steps rather than seven which was seen in (R5). Also, the code in (1R5) was different from what was generated in (R5). As for (2R5) that is, the second response of ChatGPT of question 5 in a different chat, the response generated was on a high, generic level and did not provide any specific codes.

From the three different responses generated, the developer was able to gain more insights on the steps towards successful app development. For example, steps like creating an empty activity, preparing the layout, uploading logos were all new to the developer.

Even with the steps provided by ChatGPT, it was still difficult for the developer to complete the code. Therefore, several follow up questions referred to iterations followed the response generated in R5. Steps one to four were simple steps that did not require any follow-up and was directly and successfully completed. Step five was followed with a single iteration to be completed. Step six was completed in three iterations.

Table 3 shows the steps generated by ChatGPT in response to Q5, the number of iterations, and if the step was successfully completed or not.

Table 3. Number of Iterations required per step for R5

ChatGPT generated Steps in R5	Number of Iterations	Success
1	0	yes
2	0	yes
3	0	yes
4	0	yes
5	1	yes
6	3	yes

Steps one to five were relatively easy and were easily comprehended by the developer and considered as immediately useful. As for step six, the developer required three follow up questions requesting where and how to place the code.

As result, the following was concluded from all the trials above:

1. ChatGPT, when given a very long description, can generate wrong or incomplete codes.
2. A step-by-step guide can enhance the developer's understanding of the workflow used by Android Studio but does not necessarily allow for successful app development. Though simple steps in the early process of the app development can be successfully completed through a step-by-step request.
3. Several follow-up questions must be asked by the developer to complete specific steps based on the complexity.
4. None of the physical "artefacts", were considered useful and required varying number of iterations.

As a result, a third approach was applied. Instead of asking for a step-by-step guide, the developer broke down the application furthermore, to breaking it down into specific functions rather than the whole application to avoid incomplete or wrong codes.

One of the main parts of the app is to prepare a drop-down menu with multiple options. The following section shows the approach used.

5.2.1. Creating the spinner

The developer asked ChatGPT (Q6) on how to prepare a dropdown menu using Android Studio. The question was prepared based on the lessons learned from previous responses. The input should be a command (rather than a question) and include a request for a step-by-step guide. The word "detailed" following "step-by-step" was added as an attempt to improve the description generated by ChatGPT. The following R6 shows the response:

Developer : "Provide a step-by-step detailed guide on how to prepare a dropdown menu using Android Studio"

ChatGPT: "Sure, I can provide a step-by-step guide on how to prepare a dropdown menu using Android Studio.

Step 1: Create a new project in Android Studio Launch Android Studio, and create a new project by selecting "Empty Activity" and clicking "Next." Give your project a name and choose the appropriate package name, then click "Finish" to create the project.

Step 2: Design the layout In the layout XML file, add a spinner view. The spinner view is the drop-down menu that we'll be creating"

Script for step 2 can be found in Appendix B, Table B1.

ChatGPT then continues "Here we're using an "entries" attribute to define the options for the spinner. We'll create an array of countries in the next step.

Step 3: Define the options Create a new XML file in the "res/values" directory named "strings.xml". Add an array of countries to this file:"

Script for step3 can be found in Appendix B, Table B1.

ChatGPT further continues to step 4, "Step 4: Populate the Spinner In your Java code, create an ArrayAdapter to populate the spinner with the options defined in the "countries" array."

Similarly, the script for the specific step is found in Appendix B, Table B1. ChatGPT response continues explaining thoroughly what each step and code represents.

"Here we're setting the adapter for the spinner using the "setAdapter" method. We're also setting the layout resource for the drop-down list using "setDropDownViewResource".

Step 5: Add a Listener If you want to perform an action when the user selects an item from the dropdown menu, you can add a listener to the spinner. Here's an example:"

Step 5 codes found in Appendix B, Table B1. ChatGPT follows up by stating the following "That's it! Your dropdown menu should now be working in your Android Studio project".

Steps one and two were successfully completed without any follow-up questions as they were similar to the responses in R5. Steps three and four were attempted without any follow-up question but were unsuccessful. Some errors occurred not allowing the app to run. These errors were then sent to ChatGPT requesting to solve the problem. Question (Q6a) and Response (R6a) represent the first iteration concerning step three and four:

Developer: "I received this error:

```
C:\Users\miyad\AndroidStudioProjects\research3\app\src\main\java\com\research\research3\MainActivity.java:17: error: class, interface, or enum expected Spinner spinner = findViewById(R.id.spinner);"
```

ChatGPT: "This error message typically occurs when there is a syntax error or misplaced code in your Java file. It seems like the code snippet you provided may be incomplete or not placed within a class or method.

Here's an example of how you could define a class with a spinner in your MainActivity.java file:". The response of ChatGPT can be found in Appendix B, Table B2. ChatGPT follows by stating the following, "Make sure that your code is placed inside the class definition for MainActivity, and that you have imported the necessary packages. Also, ensure that you have properly defined the layout file with the ID "spinner" in your activity_main.xml file."

The code was then tested again, and a spinner was successfully created. Though, it is important to note that some extra steps, such as importing class that were shown as solutions by Android Studio due to errors generated. Android Studio usually marks a red line under classes that requires to be imported. The steps were done by the developer inspite of it being mentioned by ChatGPT. The options in the spinner, the disclaimer and copy right messages were then modified based on the needs of the developer and was also successful.

For a spinner, a single follow up question was enough to help the developer complete the simple task.

The next part of the application was that upon selecting one of the options, a specific action must be taken.

5.2.2. Setting up the functionality of dropdown menu

As described in the Application function section, each option in the dropdown requires a specific action to be taken by the application. A new chat was created and (Q7) was asked to ChatGPT and the following response was recorded (R7):

Developer: "Provide a step-by-step detailed guide on how to prepare a dropdown menu using Android Studio. The drop-down menu shall have three options. The first is a disclaimer message, the second is the material calculator and the third is the copyright. Selecting the disclaimer message opens up the message. selecting the calculator opens a new layout requesting user to put in three inputs. selecting the copyright shows the copy right message."

ChatGPT: "Sure, here's a step-by-step guide on how to create a dropdown menu with three options in Android Studio:

Step 1: Create a new project in Android Studio

First, create a new project in Android Studio. Choose a project name and package name, and select your desired minimum SDK version.

Step 2: Create a new layout for the dropdown menu

Create a new layout file for the dropdown menu by right-clicking on the "res" folder in the project view and selecting "New" > "Layout resource file". Name the file "dropdown_menu.xml".

Step 3: Add a Spinner to the layout

In the new "dropdown_menu.xml" layout file, add a Spinner view to the layout. The Spinner will be used to display the dropdown menu options. Here's an example code snippet:". The code snippet is found in Appendix C Table C1.

Chat GPT continues "Step 4: Create an array of dropdown menu options

Create an array of strings to hold the three dropdown menu options. You can do this in the "strings.xml" file in the "values" folder. Here's an example code snippet:". The code is found in Table C1, Appendix C. The response then follows " Step 5: Set the dropdown menu options in the Spinner

In the activity or fragment that will use the dropdown menu, create a reference to the Spinner view and set the dropdown menu options in the Spinner using an ArrayAdapter. Here's an example code snippet:" Refer to Appendix C, Table C1 for the code snippet.

Step 6, the final step generated by ChatGPT included the following, " Step 6: Handle Spinner item selection

Finally, handle the Spinner item selection to perform the appropriate action when an item is selected. You can do this using a setOnItemSelectedListener. Here's an example code snippet:". For the code snippet of step 6, refer to Appendix C, Table C1. ChatGPT further elaborates on step 6 mentioning the following, "The case where "Material Calculator" is selected, you would need to create a new activity and layout file for the calculator interface. You can do this by right-clicking on the project folder, selecting "New" > "Activity" > "Empty Activity". Name the new activity "CalculatorActivity". Then create a new layout file for the calculator interface by right-clicking on the "res" folder in the project view and selecting "New" > "Layout resource file". Name the file "calculator_layout.xml". In the "CalculatorActivity.java" file, set the content view to the new layout file in the onCreate method:

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.calculator_layout);  
}
```

Then add EditText views for the three inputs you require for the calculator, and handle the button click to perform the calculation."

A similar approach was proposed by ChatGPT in R6 (Step one to five) regarding the spinner. Step six was the extra step that handled the selection options. Once again, the code was tested directly without any iterations but unsuccessful. The developer then asked follow-up questions in the same chat. The first question (Q7a) was to further explain steps one to four. Once again, one iteration for the steps was enough to get those done with. Step five required three iterations for the developer to understand how to complete it. Though, step six, which is what the application is supposed to do once the user selects a specific option, took more than 72 follow up questions to complete the code and get it working, this shows an exponential increase in the number of iterations due to the increase in complexity when compared with earlier steps. The conversation was like a one-on-one specific tutorial session. Questions on where to place the code, why did you use this function and how to implement it were asked. Follow up questions based on the errors received on Android Studio were also part of the conversation. The follow up questions resulted in corrective actions by ChatGPT. Specific questions related to what each option must do when selected were asked which furthermore increased the complexity of the code and the answers generated.

The back-and-forth conversation resulted in an application that was working perfectly fine. It is important to note that it took an average of 20 hours to complete the mobile application. An article written by [28] mentioned that it would take around 3-4 months to develop an application using

Android Studio. Another article [29] divided applications based on the complexity level. According to [29] a simple application would require a duration of 2-4 months to complete.

Figure 2 shows how the application looked using a Pixel 6 API 30 emulator when selecting different functions.

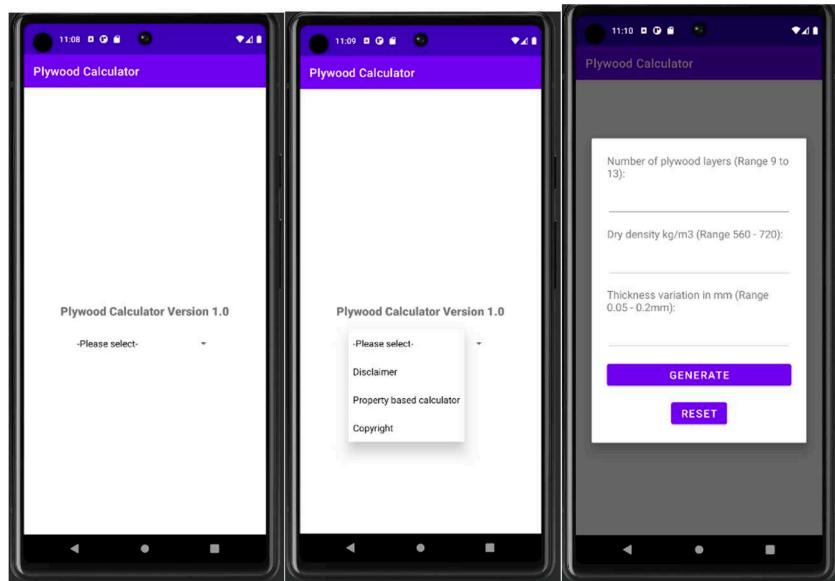


Figure 2. Pixel 6 API Emulator showing the functions of the application (left to right: opening screen, select options, data entry fields)ord cloud generated based on all responses to (Q1)

Once the main code of the application was developed, complexity and difficulty in handling modifications in the application was substantially reduced for both, ChatGPT and the developer. New chats were created where the complete code was pasted first, and then several follow-up questions were asked to add extra functions such as adding a reset button and setting up ranges for the accepted input. Even though “artefacts” received errors when applied in Android Studio, through several iterations, a novice developer can manage to overcome the issues and solve the problem, most of the added options when the code was fed to ChatGPT first, required an average of five iterations until the code ran error-free.

The insertion of the code first substantially reduced the number of iterations to achieve the results. For example, a reset button was required in the material calculator that could reset all the inputs provided by the user. The code was first inserted and then the question was asked. To get the reset button working, seven iterations were required, which is considerably lower than the number of iterations required in the previous requests. It is important to explain to ChatGPT prior to sending the code not to comment on script unless you ask a question. The reason behind this is that ChatGPT is programmed to automatically respond to any input provided by the user. A response to the code gives a description of what the code is intended to do.

6. Conclusion and Recommendations

The paper presented how ChatGPT can be used to aid novice mobile app developers in creating successful applications. A case study approach was considered to collect qualitative insights in this matter. ChatGPT and the developer conducted a one-on-one teaching pedagogy where the developer continuously asked questions on how to create specific functions using Android Studio towards the development a mobile application. The following was found. Requesting ChatGPT to provide the whole code from the start ended up in wrong or incomplete codes. Users intending to use ChatGPT must break down the functions of their application into phases, then requesting the ChatGPT to program each part specifically and gradually building on it towards the complete development of

the app. ChatGPT capabilities in remembering what users said earlier in the conversion allows for such an approach to be implemented.

It was also perceived that as the complexity of steps is increased, more follow-up questions are required to successfully complete the app. The relationship seen is considered exponential. However, once the code is completed, any extra functions required to be added are made easy by sending ChatGPT the code first, followed by the question. This was seen to substantially reduce the number of follow-up questions.

It is also highly recommended that novice users should start with a few tutorial videos about coding before using ChatGPT. Doing so has improved the developer's understanding of how to get started.

Appropriate vocabulary and statements with specific and direct commands rather than questions is a substantial contributor to receiving useful answers and reducing the number of iterations.

While the paper is limited to a specific and unique application functionality, which may be a very simple app for intermediate and professional programmers, it yet provides insights on how to utilize ChatGPT in this regard and opens the floor to further investigations on how ChatGPT would react and solve more complex applications. An application with the complexity of the one developed took an average of 20 hours to complete using Android Studio. A comparison between traditional methods of learning coding to develop the same specific mobile application by a novice developer would be of a great interest. Such a comparison would provide substantial insights and allow for further understanding of ChatGPT capabilities.

Author Contributions: Conceptualization, M.I.K and M.J.; methodology, M.J.; validation, M.I.K and M.J.; formal analysis, M.I.K.; investigation, M.I.K and M.J.; writing—original draft preparation, M.I.K.; writing—review and editing, M.I.K and M.J.; supervision, M.J.;. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is available upon the request and approval of the authors.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Creating the App script based on questions

Developer's Question	ChatGPT Response	Line number	Code
Q4	R4	1	import android.os.Bundle;
		2	import android.view.Menu;
		3	import android.view.MenuInflater;
		4	import android.view.MenuItem;
		5	import android.widget.ArrayAdapter;
		6	import android.widget.Spinner;
		7	import androidx.appcompat.app.AlertDialog;
		8	import androidx.appcompat.app.AppCompatActivity;
		9	public class MainActivity extends AppCompatActivity {
		10	private Spinner spinner;
		11	private String[] options = {"Disclaimer", "Calculator", "Copyright"};

```
12  private double input1 = 0.0;
13  private double input2 = 0.0;
14  private double input3 = 0.0;
15  private double U = 0.0;
16  @Override
17  protected void onCreate(Bundle savedInstanceState) {
18      super.onCreate(savedInstanceState);
19      setContentView(R.layout.activity_main);
20      setTitle("Material Calculator");
21      // Set up the spinner for the drop-down menu
22      spinner = findViewById(R.id.spinner);
23      ArrayAdapter<String>adapter = new ArrayAdapter<>
24          (this, android.R.layout.simple_spinner_item, options);
25      adapter.setDropDownViewResource
26          (android.R.layout.simple_spinner_dropdown_item);
27      spinner.setAdapter(adapter);
28      // Set the logo of the University
29      getSupportActionBar().setDisplayShowHomeEnabled(true);
30      getSupportActionBar().setLogo(R.drawable.university_logo);
31      getSupportActionBar().setDisplayUseLogoEnabled(true);
32      }
33      @Override
34      public boolean onCreateOptionsMenu(Menu menu) {
35          MenuInflater inflater = getMenuInflater();
36          inflater.inflate(R.menu.main_menu, menu);
37          return true;
38      }
39      @Override
40      public boolean onOptionsItemSelected(MenuItem item) {
41          switch (item.getItemId()) {
42          case R.id.action_disclaimer:
43              showDisclaimer();
44              return true;
45          case R.id.action_calculator:
46              showCalculatorDialog();
47              return true;
48          case R.id.action_copyright:
49              showCopyright();
50              return true;
51          default:
52              return super.onOptionsItemSelected(item);
53          }
54      }
55  }
56  
```

```
52    }
53    private void showDisclaimer() {
54        AlertDialog.Builder builder = new AlertDialog.Builder(this);
55        builder.setTitle("Disclaimer");
56        builder.setMessage("Disclaimer message");
57        builder.setPositiveButton("OK", null);
58        builder.show();
59    }
60    private void showCalculatorDialog() {
61        AlertDialog.Builder builder = new AlertDialog.Builder(this);
62        builder.setTitle("Material Reusability Calculator");
63        // Set up the input fields
64        final EditText input1EditText = new EditText(this);
65        input1EditText.setInputType(InputType.
66        TYPE_NUMBER_FLAG_DECIMAL);
67        input1EditText.setHint("Input 1");
68        final EditText input2EditText = new EditText(this);
69        input2EditText.setInputType(InputType.
70        TYPE_NUMBER_FLAG_DECIMAL);
71        input2EditText.setHint("Input 2 (kg/m3)");
72        final EditText input3EditText = new EditText(this);
73        input3EditText.setInputType(InputType.
74        TYPE_NUMBER_FLAG_DECIMAL);
75        input3EditText.setHint("Input 3");
76        LinearLayout layout = new LinearLayout(this);
77        layout.setOrientation(LinearLayout.VERTICAL);
78        layout.addView(input1EditText);
79        layout.addView(input2EditText);
80        layout.addView(input3EditText);
81        builder.setView(layout);
82        builder.setPositiveButton("Calculate", new
83        DialogInterface.OnClickListener() {
84            @Override
85            public void onClick(DialogInterface dialogInterface, int i) {
86                // Retrieve the values from the input fields
87                try {
88                    input1 = Double.parseDouble(input1EditText.getText().
89                    toString());
89                } catch (NumberFormatException e) {
90                    input1 = 0.0;
91                }
92                try {
93                    input2 = Double.parseDouble(input2EditText.getText().
94                    toString());
94                } catch (NumberFormatException e) {
95                    input2 = 0.0;
96                }
97                try {
98                    input3 = Double.parseDouble(input3EditText.getText().
99                    toString());
99                } catch (NumberFormatException e) {
100                    input3 = 0.0;
101                }
102                calculate();
103            }
104        });
105        builder.show();
106    }
107    private void calculate() {
108        double result = (input1 * input2 * input3) / 1000;
109        String resultString = String.format("Result: %.2f", result);
110        Toast.makeText(this, resultString, Toast.LENGTH_SHORT).show();
111    }
112}
```

		input2 = Double.parseDouble(input2EditText.getText(). toString());
	89	}
	90	catch (NumberFormatException e) {
	91	input2 = 0.0;
	92	}
	93	try {
	94	input3 = Double.parseDouble(input3EditText.getText(). toString());
	95	}
	96	catch (NumberFormatException e) {
	97	input3 = 0.0;
	98	// Calculate the number of uses
	99	if (input1"

Table A2. Creating the App script based on questions

Developer's Question	ChatGPT Response	Line number	Code
Q4-Followup	R4-Followup	1	// Get the values of input 1, input 2 and input 3
		2	String input1Value = input1.getText().toString();
		3	String input2Value = input2.getText().toString();
		4	String input3Value = input3.getText().toString();
		5	// Check if any input field is empty
		6	if (input1Value.isEmpty() input2Value.isEmpty() input3Value.isEmpty()) {
		7	resultView.setText("Input missing");
		8	} else {
		9	// Parse input 2 as a float value
		10	float input2Float = Float.parseFloat(input2Value);
		11	// Calculate the estimated max. number of uses using the formula
		12	float U = (1000 * input2Float * 0.7f) / (Float.parseFloat(input1Value) * Float.parseFloat(input3Value));
		13	// Display the result in the resultView
		14	resultView.setText("Estimated max. number of uses: " + String.format("%.2f", U));
		15	}
		16	}
		17	});
		18	}
		19	}

Table A3. Creating the App script based on questions

Developer's Question	ChatGPT Response	Line number	Code
Q5	R5	1	<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
		2	xmlns:app="http://schemas.android.com/apk/res-auto"
		3	android:layout_width="match_parent"
		4	android:layout_height="match_parent">
		5	<TextView
		6	android:id="@+id/title"
		7	android:text="Material Calculator"
		8	android:textSize="20sp"
		9	android:layout_width="wrap_content"
		10	android:layout_height="wrap_content"
		11	android:layout_centerHorizontal="true"
		12	android:layout_marginTop="10dp" />
		13	<ImageView
		14	android:id="@+id/logo"
		15	android:src="@drawable/university_logo"
		16	android:layout_width="wrap_content"
		17	android:layout_height="wrap_content"
		18	android:layout_below="@+id/title"
		19	android:layout_centerHorizontal="true"
		20	android:layout_marginTop="20dp" />
		21	<Spinner
		22	android:id="@+id/menu"
		23	android:layout_width="wrap_content"
		24	android:layout_height="wrap_content"
		25	android:layout_below="@+id/logo"
		26	android:layout_centerHorizontal="true"
		27	android:layout_marginTop="20dp" />
		28	Add the following code to the MainActivity.java file to set up the spinner:
			public class MainActivity extends AppCompatActivity implements
		29	AdapterView.OnItemSelectedListener {
		30	private Spinner menuSpinner;
		31	@Override
		32	protected void onCreate(Bundle savedInstanceState) {
		33	super.onCreate(savedInstanceState);
		34	setContentView(R.layout.activity_main);
		35	menuSpinner = (Spinner) findViewById(R.id.menu);
		36	ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this, R.array. menu_options, android.R.layout.simple_spinner_item);
		37	adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

```
38     menuSpinner.setAdapter(adapter);
39     menuSpinner.setOnItemSelectedListener(this);
40 }
41     @Override
42     public void onItemSelected(AdapterView<?> adapterView, View view, int position, long l) {
43     switch (position) {
44     case 0:
45     showDisclaimer();
46     break;
47     case 1:
48     showCalculatorDialog();
49     break;
50     case 2:
51     showCopyright();
52     break;
53   }
54 }
55     @Override
56     public void onNothingSelected(AdapterView<?> adapterView) {
57   }
58     private void showDisclaimer() {
59     AlertDialog.Builder builder = new AlertDialog.Builder(this);
60     builder.setTitle("Disclaimer");
61     builder.setMessage("This application is for informational purposes only.
Please consult a professional for accurate calculations.");
62     builder.setPositiveButton("OK", null);
63     builder.create().show();
64   }
65     private void showCalculatorDialog() {
66     AlertDialog.Builder builder = new AlertDialog.Builder(this);
67     builder.setTitle("Calculator");
68     LayoutInflator inflater = getLayoutInflater();
69     View dialogView = inflater.inflate(R.layout.dialog_calculator, null);
70     builder.setView(dialogView);
71     final EditText input1EditText = (EditText) dialogView.findViewById(R.id.input1EditText);
72     final EditText input2EditText = (EditText) dialogView.findViewById(R.id.input2EditText);
73     final EditText input3EditText = (EditText) dialogView.findViewById(R.id.input3EditText);
74     builder.setPositiveButton("Calculate", new DialogInterface.OnClickListener() {
75     @Override
76     public void onClick(DialogInterface dialogInterface, int i) {
77     String input1 = input1EditText.getText()
```

Appendix B

Table B1. Creating the spinner

Developer's Question	ChatGPT Response	Steps	Line number	Code
Q6	R6	2	1	<Spinner
			2	android:id="@+id/spinner"
			3	android:layout_width="match_parent"
			4	android:layout_height="wrap_content"
			5	android:entries="@array/countries" />
		3	6	<string-array name="countries">
			7	<item>USA</item>
			8	<item>Canada</item>
			9	<item>Mexico</item>
			10	<item>Brazil</item>
			11	</string-array>
		4	12	Spinner spinner = findViewById(R.id.spinner);
			13	ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,
			14	R.array.countries, android.R.layout.simple_spinner_item);
			15	adapter.setDropDownViewResource
			16	(android.R.layout.simple_spinner_dropdown_item);
		5	17	spinner.setAdapter(adapter);
			18	spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
			19	@Override
			20	public void onItemSelected(AdapterView<?>
			21	parent, View view, int position, long id) {
			22	String selectedCountry = parent.getItemAtPosition(position).toString();
			23	Toast.makeText(MainActivity.this, "Selected: "
			24	+ selectedCountry, Toast.LENGTH_SHORT).show();
			25	}
			26	@Override
				public void onNothingSelected(AdapterView<?> parent) {
				}
				});

Table B2. Creating the spinner

Developer's Question	ChatGPT Response	Line number	Code
Q6(a)	R6(a)	1	import android.os.Bundle;
		2	import android.view.View;
		3	import android.widget.AdapterView;
		4	import android.widget.ArrayAdapter;

				5 import android.widget.Spinner;
				6 import android.widget.Toast;
				7 import androidx.appcompat.app.AppCompatActivity;
				8 public class MainActivity extends AppCompatActivity {
				9 @Override
				10 protected void onCreate(Bundle savedInstanceState) {
				11 super.onCreate(savedInstanceState);
				12 setContentView(R.layout.activity_main);
				13 // Find the spinner view by its ID
				14 Spinner spinner = findViewById(R.id.spinner);
				15 // Create an ArrayAdapter using the string array and a default spinner layout
				16 ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,
				17 R.array.countries, android.R.layout.simple_spinner_item);
				18 // Specify the layout to use when the list of choices appears
				19 adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
				20 // Apply the adapter to the spinner
				21 spinner.setAdapter(adapter);
				22 // Add a listener to the spinner
				23 spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
				24 @Override
				25 public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
				26 // Get the selected item from the spinner
				27 String selectedCountry = parent.getItemAtPosition(position).toString();
				28 // Display a Toast message with the selected item
				29 Toast.makeText(MainActivity.this, "Selected: " + selectedCountry, Toast.LENGTH_SHORT).show();
				30 }
				31 @Override
				32 public void onNothingSelected(AdapterView<?> parent) {
				33 }
				34 });
				35 }
				36 }

Appendix C

Table C1. Setting up the functionality of dropdown menu

Developer's Question	ChatGP Response	Step	Line number	Code
Q7	R7	3	1	<Spinner

		2	2 android:id="@+id/dropdown_menu"
		3	3 android:layout_width="wrap_content"
		4	4 android:layout_height="wrap_content" />
	4	5	5 <string-array name="menu_options">
		6	6 <item>Disclaimer Message</item>
		7	7 <item>Material Calculator</item>
		8	8 <item>Copyright</item>
		9	9 </string-array>
	5	10	10 Spinner dropdownMenu = findViewById(R.id.dropdown_menu);
		11	11 ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
		12	12 android.R.layout.simple_spinner_item, getResources().getStringArray(R.array.menu_options));
		13	13 adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
		14	14 dropdownMenu.setAdapter(adapter);
	6	15	15 dropdownMenu.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
		16	16 @Override
		17	17 public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
		18	18 switch (position) {
		19	19 case 0:
		20	20 // Display disclaimer message
		21	21 break;
		22	22 case 1:
		23	23 // Launch material calculator layout
		24	24 Intent intent = new Intent(MainActivity.this, CalculatorActivity.class);
		25	25 startActivity(intent);
		26	26 break;
		27	27 case 2:
		28	28 // Display copyright message
		29	29 break;
		30	30 }
		31	31 }
		32	32 @Override
		33	33 public void onNothingSelected(AdapterView<?> parent) {
		34	34 // Do nothing
		35	35 }
		36	36 });
		37	37 }

References

1. Aleksandra Fostikov. 2023. First impressions on using AI powered chatbots, tools and search engines: ChatGPT, Perplexity and other – possibilities and usage problems. (2023). DOI:<http://dx.doi.org/https://doi.org/10.17613/9t6p-2229>
2. Azam Syukur Rahmatullah, E. Mulyasa, Syahrani Syahrani, Fien Pongpalilu, and Riana Eka Putri. 2022. Digital Era 4.0. Linguistics and Culture Review 6 (2022), 89–107. DOI:<http://dx.doi.org/10.21744/lingcure.v6ns3.2064>
3. Aline F.S. Borges, Fernando J.B. Laurindo, Mauro M. Spínola, Rodrigo F. Gonçalves, and Claudia A. Mattos. 2021. The strategic use of artificial intelligence in the Digital Era: Systematic Literature Review and Future Research Directions. International Journal of Information Management 57 (2021), 102225. DOI:<http://dx.doi.org/10.1016/j.ijinfomgt.2020.102225>
4. Ankit Kesharwani. 2020. Do (how) Digital Natives adopt a new technology differently than digital immigrants? A longitudinal study. Information & Management 57, 2 (2020), 103170. DOI:<http://dx.doi.org/10.1016/j.im.2019.103170>
5. Eleni Adamopoulou and Lefteris Moussiades. 2020. An overview of chatbot technology. IFIP Advances in Information and Communication Technology (2020), 373–383. DOI:http://dx.doi.org/10.1007/978-3-030-49186-4_31
6. Jürgen Rudolph, Samson Tan, and Shannon Tan. 2023. Chatgpt: Bullshit spewer or the end of traditional assessments in higher education? 1 6, 1 (2023). DOI:<http://dx.doi.org/10.37074/jalt.2023.6.1.9>
7. Jie Zhou, Pei Ke, Xipeng Qiu, Minlie Huang, and Junping Zhang. 2023. CHATGPT: Potential, prospects, and limitations. Frontiers of Information Technology & Electronic Engineering (2023). DOI:<http://dx.doi.org/10.1631/fitee.2300089>
8. Michael Liebrenz, Roman Schleifer, Anna Buadze, Dinesh Bhugra, and Alexander Smith. 2023. Generating scholarly content with chatgpt: Ethical challenges for medical publishing. The Lancet Digital Health 5, 3 (2023). DOI:[http://dx.doi.org/10.1016/s2589-7500\(23\)00019-5](http://dx.doi.org/10.1016/s2589-7500(23)00019-5)
9. Kindra Cooper. 2022. OpenAI GPT-3: Everything you need to know. (October 2022). Retrieved March 12, 2023 from <https://www.springboard.com/blog/data-science/machine-learning-gpt-3-open-ai/>
10. OpenAI. 2015. Introducing chatgpt. (2015). Retrieved March 12, 2023 from <https://openai.com/blog/chatgpt>
11. Dan Milmo. 2023. CHATGPT reaches 100 million users two months after launch. (February 2023). Retrieved March 12, 2023 from <https://www.theguardian.com/technology/2023/feb/02/chatgpt-100-million-users-open-ai-fastest-growing-app>
12. Zhicheng Lin. 2023. Why and how to embrace AI such as CHATGPT in your academic life. (2023). DOI:<http://dx.doi.org/10.31234/osf.io/sdx3j>
13. Marcel Binz and Eric Schulz. 2023. Using cognitive psychology to understand GPT-3. Proceedings of the National Academy of Sciences 120, 6 (2023). DOI:<http://dx.doi.org/10.1073/pnas.2218523120>
14. Ashley M. Hopkins, Jessica M. Logan, Ganessan Kichenadasse, and Michael J. Sorich. 2023. AI chatbots will revolutionize how cancer patients access information: CHATGPT represents a paradigm-shift. JNCI Cancer Spectrum (2023). DOI:<http://dx.doi.org/10.1093/jncics/pkad010>
15. Aidan Gilson et al. 2023. How does CHATGPT perform on the United States Medical Licensing Examination? the implications of large language models for medical education and knowledge assessment. JMIR Medical Education 9 (2023). DOI:<http://dx.doi.org/10.2196/45312>
16. Travis Ryan Pickell and Brian R. Doak. 2023. Five Ideas for How Professors Can Deal with GPT-3 ... For Now (2023).
17. Muneer Alshater. 2022. Exploring the role of artificial intelligence in enhancing academic performance: A case study of chatgpt. SSRN Electronic Journal (2022). DOI:<http://dx.doi.org/10.2139/ssrn.4312358>
18. Calum Macdonald, Davies Adeloye, Aziz Sheikh, and Igor Rudan. 2023. Can chatgpt draft a research article? an example of population-level vaccine effectiveness analysis. Journal of Global Health 13 (2023). DOI:<http://dx.doi.org/10.7189/jogh.13.01003>
19. Ali Borji. 2023. A Categorical Archive of ChatGPT Failures. arXiv preprint (2023). DOI:<http://dx.doi.org/https://doi.org/10.48550/arXiv.2302.03494>
20. Sakib Shahriar and Kadhim Hayawi. 2023. Let's have a chat! A Conversation with ChatGPT: Technology, Applications, and Limitations. arXiv preprint (2023). DOI:<http://dx.doi.org/https://doi.org/10.48550/arXiv.2302.13817>

21. Domonik Sobania, Martin Briesch, Carol Hanna, and Justyna Petke. 2023. An Analysis of the Automatic Bug Fixing Performance of ChatGPT. arXiv preprint (2023). DOI:<http://dx.doi.org/https://doi.org/10.48550/arXiv.2301.08653>
22. Robert K. Yin. 2003. Case study research: Design and methods, Thousand Oaks, CA: Sage Publications.
23. Kimberly A. Neuendorf. 2002. The Content Analysis Guidebook, Thousand Oaks (California), Cleveland State University: Sage.
24. G2. 2012. Business software and services reviews | G2. (2012). Retrieved March 10, 2023 from <https://www.g2.com/>
25. [Codes Easy]. (2017, December 29). Creating First Application In Android Studio in 2023 [Video]. Youtube. <https://www.youtube.com/watch?v=p0ItPcqXog>
26. [Wasay Tech Tips]. (2020, September 29). How to Install Emulator in Android Studio 2020 [Video]. Youtube. <https://www.youtube.com/watch?v=P1LF15CYulc>
27. [Coding in Flow]. (2017, December 29). How to Change the App Icon in Android Studio (With Adaptive Icons) [Video]. Youtube. <https://www.youtube.com/watch?v=ts98gL1JCQU>
28. Alex Moazed. 2017. How long does it take to build an IOS or Android mobile app? (May 2017). Retrieved March 21, 2023 from <https://www.applicoinc.com/blog/long-take-build-ios-android-mobile-app/#:~:text=It%20will%20usually%20take%203,of%20building%20a%20mobile%20app>.
29. Spdload. 2023. Find out how long does it take to build an app in 2023. (January 2023). Retrieved March 21, 2023 from <https://spdload.com/blog/how-long-does-it-take-to-develop-an-app/>

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.