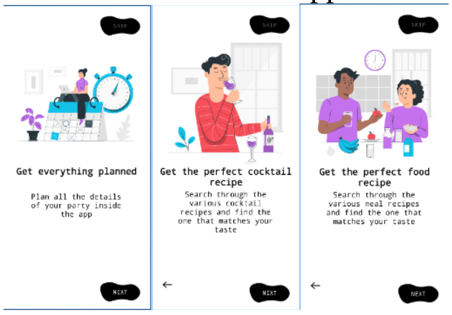
Background

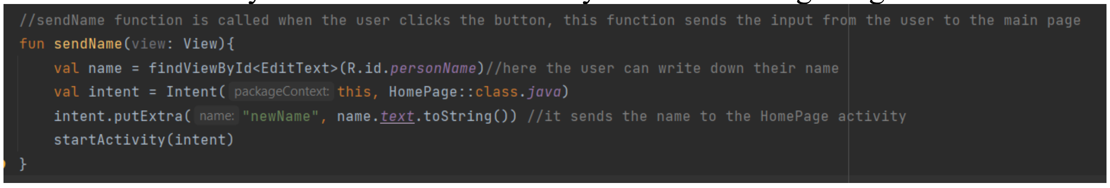
MeetUp is an application made for people that want to organize a party with their friends. MeetUp is developed to be used on mobile devices. The application has multiple features: gives the user various food and cocktail recipes, displays the current position of the user, allows to user to create to do lists inside the app and provides them with various pieces of advice. This application helps planning different parties, by having everything the user needs in one single application. Such an application is full of design and technical opportunities. From a design perspective, the application uses three designated colors that help in creating the visual identity of the application. The colors are: #12c2e9, #c471ed and #f64f59. These colors are not only used for the background, but also the images displayed inside the app which creates a consistent design. The consistency of the design helps in creating a brand identity. The text font used through the entire application is Monospace. The design of the application is helpful in creating a user friendly interface. For example, the application starts by guiding the user through the main features of the application.


SKIP
Get everything planned
Plan all the details of your party inside the app
NEXT
SKIP
Get the perfect cocktail recipe
Search through the various cocktail recipes and find the one that matches your taste
NEXT
SKIP
Get the perfect food recipe
Search through the various meal recipes and find the one that matches your taste
NEXT

From a technical perspective the app uses three APIs, a location sensor and passing data from an activity to another using the following methods: putExtra() and getStringExtra(). PutExtra() is used for getting the name of the user and sending it to the main activity called “HomePage.kt”. The getStringExtra() is used for placing the name inside the TextView. MeetUp is inspired by multiple existing apps such as: Event Planner (https://play.google.com/store/apps/details?id=com.mmobile.app.event&hl=en), BigNight (https://bignightapp.com/) and Cocktail Flow (https://cocktailflow.com/). However, MeetUp is incorporating all three features of these applications to inspire my application, providing the user with food and cocktail recipes and a page where the user can write down their to do lists. In addition, MeetUp also provides the user with their current position that can be sent to their guests and a section with pieces of advice that can be shared with the guests at the dinner table.

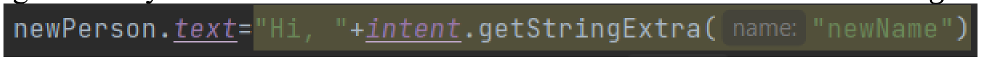
Description and Justification

The application starts with “MainActivity” which contains two image views, two text views and a button that has a background image. The method “buttonPressed” is attached to the button. By clicking this button, the user is sent to the next activity “MainActivity2”. Attached to the skip button, there is the “goToName” method that sends the user directly to the “AskName” activity. The “AskName” activity has the role of getting the user’s name.



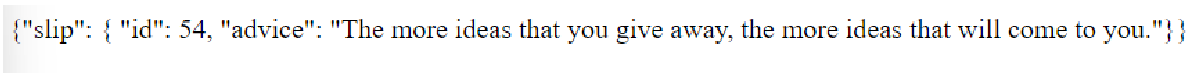
```
//sendName function is called when the user clicks the button, this function sends the input from the user to the main page
fun sendName(view: View){
    val name = findViewById<EditText>(R.id.personName)//here the user can write down their name
    val intent = Intent( packageContext: this, HomePage::class.java)
    intent.putExtra( name: "newName", name.text.toString()) //it sends the name to the HomePage activity
    startActivity(intent)
}
```

The putExtra() method sends the input to the next activity. Name.text.toString() converts the input to be a string and to be able to displayed inside “HomePage” activity. The “HomePage” activity contains the newPerson variable that has the TextView assigned to it.



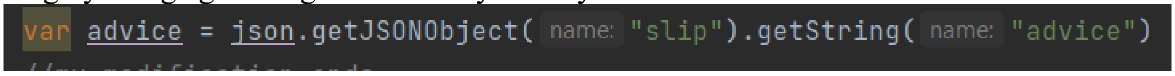
```
newPerson.text="Hi, "+intent.getStringExtra( name: "newName")
```

The prior line of code concatenates two strings: the greeting and the name that has been passed and displays it onto the screen. The “HomePage” activity also contains a section that is designated to show pieces of advice. This section uses the Advice Slip API, for this feature the application needs internet permission. The structure of the API is the following.



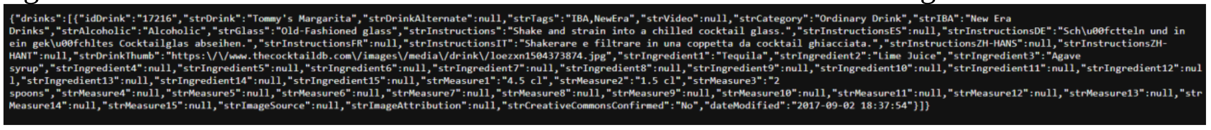
{"slip": { "id": 54, "advice": "The more ideas that you give away, the more ideas that will come to you."}}

In order to access the advice, I have extracted it from the object named "slip" and then got the string by using "getString" followed by the key name "advice".



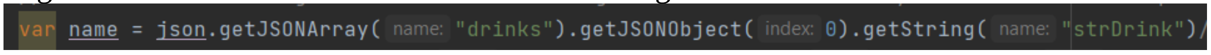
```
var advice = json.getJSONObject( name: "slip").getString( name: "advice")
```

The "HomePage" also displays the current location of the user, by accessing the location sensor. In order for this sensor to be accessed the application requires two permissions: "android.permission.ACCESS_FINE_LOCATION" and "android.permission.ACCESS_COARSE _ LOCATION". Once the permission is granted, the sensor starts which displays the latitude and longitude of the user.
The menu containing the following options: "Plan", "Cocktail recipe" and "Food recipe" is inside a horizontal scroll view. Each option is inside a frame layout that contains a descriptive image button, the title and a description of the feature. Attached to each of the image buttons, there is a method that sends the user to the respective activity page.
The "CocktailPage" uses TheCocktailDB API, for this the application needs access to Internet. The button with the id "getDrink" has attached to it the "getdrink" method. This method calls the "getDataFromServer(url)" function that connects to the server. "getDataFromServer()" connects to the server using OkHTTP. The "onFailure" function prints the error to the console in case there is one. The "readJSONFact" function extracts the right information from the API. The structure of the API is the following.



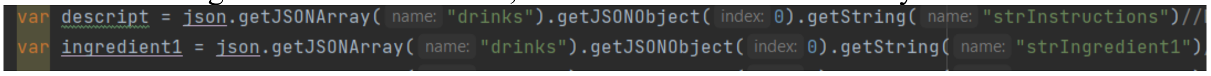
{"drinks":[{"idDrink":"17216","strDrink":"Tommy's Margarita","strDrinkAlternate":null,"strTags":"IBA,NewEra","strVideo":null,"strCategory":"Ordinary Drink","strIBA":"New Era
Drinks","strAlcoholic":"Alcoholic","strGlass":"Old-Fashioned glass","strInstructions":"Shake and strain into a chilled cocktail glass.","strInstructionsES":null,"strInstructionsDE":"Sch\u00fctteln und in
ein gek\u00fchltes Cocktailglas abseihen.","strInstructionsFR":null,"strInstructionsIT":"Shakerare e filtrare in una coppetta da cocktail ghiacciata.","strInstructionsZH-HANS":null,"strInstructionsZH-
HANT":null,"strDrinkThumb":"https:\/\/www.thecocktaildb.com\/images\/media\/drink\/loezxn1504373874.jpg","strIngredient1":"Tequila","strIngredient2":"Lime Juice","strIngredient3":"Agave
syrup","strIngredient4":null,"strIngredient5":null,"strIngredient6":null,"strIngredient7":null,"strIngredient8":null,"strIngredient9":null,"strIngredient10":null,"strIngredient11":null,"strIngredient12":nul
l,"strIngredient13":null,"strIngredient14":null,"strIngredient15":null,"strMeasure1":"4.5 cl","strMeasure2":"1.5 cl","strMeasure3":"2
spoons","strMeasure4":null,"strMeasure5":null,"strMeasure6":null,"strMeasure7":null,"strMeasure8":null,"strMeasure9":null,"strMeasure10":null,"strMeasure11":null,"strMeasure12":null,"strMeasure13":null,"str
Measure14":null,"strMeasure15":null,"strImageSource":null,"strImageAttribution":null,"strCreativeCommonsConfirmed":"No","dateModified":"2017-09-02 18:37:54"}]}

From this API, I needed the title of the drink, the instructions and the ingredients. As a result, to get the title of the drink I have used the following line of code.



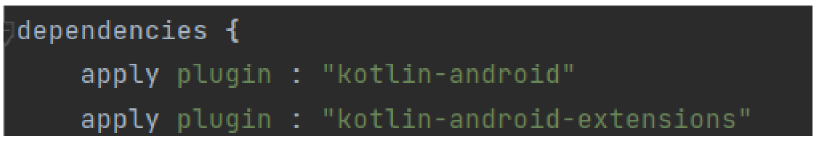
```
var name = json.getJSONArray( name: "drinks").getJSONObject( index: 0).getString( name: "strDrink")/
```

"drinks" is the name of the array that is formed of one object; therefore, it is indexed as 0 and then in order to get the title of the drink, I have used the "strDrink" key.



```
var descript = json.getJSONArray( name: "drinks").getJSONObject( index: 0).getString( name: "strInstructions")//
var ingredient1 = json.getJSONArray( name: "drinks").getJSONObject( index: 0).getString( name: "strIngredient1")
```

The same method was used to get the instructions and the ingredients, but I have used "strInstructions" and "strIngredient1" keys.
The "FoodPage" activity uses the "TheMealDB" API, the structure of this API is the same as the one used for the "CocktailPage", thus the same methods were used.
The "PlanPage" activity allows the user to create a to do list. The class named "TodoAdapter" creates the logic behind the "PlanPage" activity. Inside the "TodoAdapter" class, the "onCreateHolder" function creates the view holder. The LayoutInflater is used to get a reference to the specific view that is in the xml file. "addTodo" function is used for adding a new item and it is updating the list of tasks. The "deleteDoneTodos" function deletes the tasks that are checked as done. The "toggleStrikeThrough" function crosses out the tasks that are done. The "onBindViewHolder" function sets the right text inside the correct view item. The "getItemCount()" function helps in updating the size of the todos list. The "Todo" class contains data for the title of the task and the variable isChecked of type Boolean that is initially assigned as false. Inside the "PlanPage" class are two "setOnClickListener" methods. "btnAddTodo.setOnClickListener" is called when the user presses the "ADD" button. This method displays the next task on the screen after checking if the data added is not empty. The "btnDeleteDoneTodos.setOnClickListener" is called when the user presses the "DELETE" button. This method deletes the tasks that are marked as done.
In order to access the IDs of XML items, without using the "findviewbyid", I have added the following plugins.


```
dependencies {
    apply plugin : "kotlin-android"
    apply plugin : "kotlin-android-extensions"
```
As mentioned before these features are supposed to be used on a mobile device that can be connected to internet and has a location sensor. The application does not send notifications, therefore the user is not distracted by the application when they are outside of it.
From an ethical perspective, the app asks the user for permission to use the location sensor so they know the application is using it.
From a security perspective, the application does not contain any confidential information. In case the user would write something confidential on the “PlanPage”, that data is not stored inside the app.

Critical Reflection

In my opinion, the project met its primary aims; it provides the user with pieces of advice, food recipes, cocktail recipes, current location and a page where the user can organize their party. However, the application was supposed to have a page where the user could take pictures, but some of the functions that I have tried using were deprecated, such as “startActivityForResult()” and I could not find a solution for this issue. I have tried replacing it with a page where the user can draw their ideas and even plan where each guest can sit at the table, however this idea did not go as planned either. Thus, I finally decided on using a location sensor. While this was not my first choice, it proved to be one of the most important features of the application, providing the user with their exact location in longitude and latitude rather than their address. This is beneficial when events are held in remote/unnamed destinations. However, a future version of this app would also have these two functionalities because they would be helpful to the user.
Another issue is when an item retrieved from the APIs is equal to “null”, the page displays “null”, instead of displaying an empty string. In addition, the application also displays “null” instead of the user’s name after using the APIs.
In terms of features that worked well, I want to point out the feature that asks the user for their name, and places it on the top of the main page. This creates a personalized experience for each user, making it feel more intimate and specific to their needs. In addition, the display of information requested from the APIs is very beneficial. The displays used makes the advice easy to read, but most importantly makes the recipes easy for the user to follow at home. This is particularly important because the user can save time in preparation for and during the party they have planned. Hence, while there are some improvements that could be made, the benefits of this application outweigh these minor issues.

Word count: 1512

Reference List:
Square (2019) OkHTTP (v4.x) [Software] Available at: https://square.github.io/okhttp/
Advice slip JSON API (no date) Advice Slip JSON API. Available at: https://api.adviceslip.com/
Welcome to themealdb (no date) TheMealDB.com. Available at: https://themealdb.com/
Welcome to thecocktaildb (no date) TheCocktailDB.com. Available at: https://www.thecocktaildb.com/
SS-MIConvenor22 (no date) SensorExamples/MainActivity.kt at main · SS-Miconvenor22/Sensorexamples, GitHub. Available at: https://github.com/SS-MIConvenor22/SensorExamples/blob/main/Position/app/src/main/java/com/example/position/MainActivity.kt (Accessed: December 27, 2022).
SS-MIConvenor22 (no date) Mi-practical4/MainActivity.kt at master · SS-Miconvenor22/Mi-practical4, GitHub. Available at: https://github.com/SS-MIConvenor22/MI-practical4/blob/master/app/src/main/java/com/example/practical4/MainActivity.kt (Accessed: December 1, 2022).
Build a simple Android app with Kotlin (2020) YouTube. Available at: https://youtu.be/BBWyXo-3JGQ (Accessed: January 05, 2023).

Description of Asset	Source	Location	License/Permission
	https://www.freepik.com/free-vector/multicultural-people-standing-together_9176081.htm#page=2&position=28&from_view=author	Activity_main.xml	Used under Illustration for Instruction principle.
	https://www.freepik.com/free-vector/soft-skills-concept-illustration_24237558.htm#&position=8&from_view=undefined	Activity_home_page.xml	Used under Illustration for Instruction principle.
	https://www.freepik.com/free-vector/tasting-illustration-concept_6193929.htm#query=drinking&position=5&from_view=search&track=sph	Activity_main3.xml Activity_home_page.xml	Used under Illustration for Instruction principle.
	https://www.freepik.com/free-vector/time-management-concept-illustration_6263624.htm#&position=16&from_view=undefined	Activity_main2.xml Actvity_home_page.xml	Used under Illustration for Instruction principle.
	https://www.freepik.com/free-vector/set-healthy-dishes_4667459.htm#&position=2&from_view=undefined	Activity_food_page.xml	Used under Illustration for Instruction principle.
	https://www.freepik.com/free-vector/eat-breakfast-concept-illustration_26581437.htm#&position=18&from_view=undefined	Activity_main4.xml Activity_home_page.xml	Used under Illustration for Instruction principle.

	https://www.freepik.com/free-vector/150-arrow-icons_1082385.htm#query=arrow&position=31&from_view=search&track=sph	Activity_main3.xml Activity_main4.xml Activity_food_page.xml Activity_cocktail_page.xml	Used under Illustration for Instruction principle.
	https://www.freepik.com/free-vector/cocktail_2921828.htm#&position=6&from_view=undefined	Activity_cocktail_page.xml	Used under Illustration for Instruction principle.
API for pieces of advice	Advice Slip JSON API	Activity_home_page.xml	Used under Illustration for Instruction principle.
API for food recipes	TheMealDB.com	Activity_food_page.xml	Used under Illustration for Instruction principle.
API for drink recipes	TheCocktailDB.com	Activity_cocktail_page.xml	Used under Illustration for Instruction principle.