Assignment 1: Designing & Rapid Prototyping

The application that I will be improving upon is Android Auto, Google's recent solution of navigation. This application is installed and run off an android device but can also be connected to any supported vehicle's center infotainment display system. Android Auto contains features such as navigation, music selection, phone and messaging which can be controlled over voice commands through Google Assistance, physical screen touches or steering wheel controls if connected to the car. The major problem with this existing application is that navigating between features and using such features whilst driving is quite cumbersome. Each feature requires you to switch tabs to it, meaning more time having your eyes off the road, increasing distracted driving that is dangerous for the driver and others on the road. As this application is already in use by many people in the world, improving upon its design for safer and easier usage is of high importance.

User Groups/Tasks

The main user groups that will be using this application will be drivers and passengers. Firstly, drivers would mainly need navigational information displayed that would show them a map, current position, route to take, and next major turn. The navigation page should also allow the driver to create a route, cancel current route, set new route or find alternative route. Secondary to that would be music selection, which would show the current track, start and pause, seeking options, and playlist options. This should not hinder the navigation aspect at all by being a splitscreen page alongside the navigation page. The driver will also rarely use the phone and messaging page, both of which will also open in splitscreen alongside the navigation page.

Passengers on the other hand will mostly only need controls for the music selection. As mentioned above, this would display the current track, start pause and seeking options, and playlist options. Choosing this page should not hinder the driver in anyway so it should be displayed alongside the navigation page. They would also likely use the messaging page and phone page more frequently than the driver so these would also be displayed in splitscreen alongside the navigation page when selected. The phone page should have a numpad that allows users to dial a number, and a contacts subpage that will show their contacts list. The messaging page would show a list of current chats and selecting one of them will bring it up. As an added feature for passengers, if they do not want to navigate the pages in splitscreen view and if Android Auto is connected and being used on the cars infotainment system, then passengers may choose to navigate the pages on the phone and have it fullscreen on there. This allows the driver to retain full navigation whilst the passengers enjoy full usability of the features such as music, phone and messaging.

Scenarios

User 1 is a driver with no passengers. He is using Android Auto connected to his car (using the car's infotainment screen as the main screen). Upon connection, he is met with the default home screen which shows the time and any planned drives he has, and proceeds to select the navigation page. Google Maps (main navigation component) in Android Auto opens fullscreen on the display, where he sees a map of his surrounding area and a blue dot indicating his current position. He wishes to go to the nearest McDonalds, so he searches for it by typing it into the search bar on the top left (an onscreen keyboard opens when search bar is selected). After he selects the closest one from the search list, a quick overview of the drive is given, and he confirms the routes selection on screen and sets off. Along the way he decides to put on music, so he taps on the music icon from the menu bar. The music component opens in splitscreen, alongside the already open navigation component. He can select the playlist option and his desired playlist to start playing.

User 2 is a passenger in the car. The driver is using Android Auto on his phone only. The passenger decides that she wants to choose a different song to play. She selects the music icon from the menu bar and the music page opens in splitscreen beside the open navigation page. She selects the next song from the playlist and returns the display to navigation fullscreen by tapping the navigation icon.

User 3 is a passenger in the car, and the driver has Android Auto connected to the car. He needs phone his parents, but the driver needs the entire navigation screen that's being displayed on the car's infotainment screen. User 3 picks up the phone that is connected to the car and wakes it from sleep, presenting a secondary screen of Android Auto. He selects the phone icon from its menu bar and the phone page opens on the phone in fullscreen. This does not affect the navigation screen on the car's display.

User 4 is a driver with her phone connected to the car with Android Auto running on the car's infotainment display. She already has the navigation page up with a route already actively giving instructions. She wants to skip to the next song quickly, so she engages the steering wheel switch that changes to the next song. A small notification style overlay shows up at the top of the screen signaling the change. She then wants to message her boyfriend to tell him that she'll be arriving in the next minute or two but is too busy focusing on the road to select the messaging page, so she uses voice commands. She engages Google Assistant by toggling the voice button on the steering wheel and says, "message Sam I will be arriving in a minute". The voice assistant rereads the command and asks for confirmation to which user 4 replies "send". The message is then sent at that point and the entire time the screen displaying navigation information has not been hindered nor altered in any way.

Prototyping Techniques

For prototypes, the most simplistic low fidelity technique would be card-based prototyping, which allows each card to represent a screen of a possible application. As this is low fidelity, not all details will be included in the sketches and interactions between screens will not be accurately displayed. This brings me to the second prototyping technique I used called storyboarding. This allows me to link the card-based prototypes with a story or scenario that users may find themselves in when using the application and shows how they would progress through each screen. As a final prototyping technique, I will convert some of the low fidelity card sketches into medium fidelity wireframes that would better illustrate how the interface would look like.

Functionalities

Since we are improving upon the current Android Auto's design and functions, most of its existing main functions such as navigation, music, phone and messaging will be retained. The home page of Android Auto looks like this in phone and connected car respectively:

Figure 1: Phone Homescreen

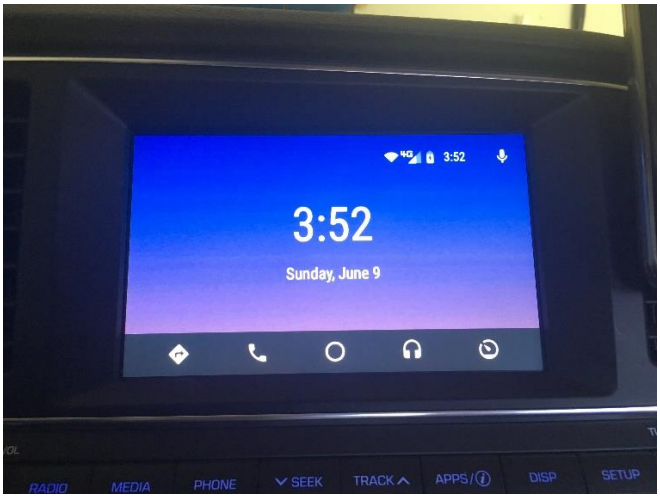

Figure 2: Car Homescreen

Navigation

One of the main functionalities that Android Auto will have is navigation. Currently the navigation layout Android Auto has is functional at best but can be drastically improved upon.

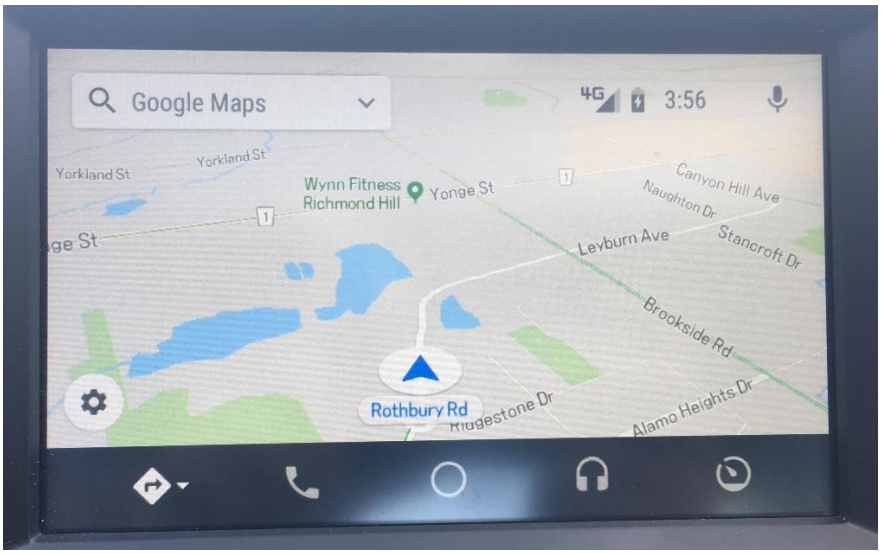

Figure 3: Car Display of Navigation Page

As you can see from the picture, the navigation page is a basic Google Maps view of your current surrounding area with a small blue dot showing your current location. At the top there is a search bar to search for a destination. This is slightly different when connected to the car's display, where the search bar is at the top left with two destination options for Home and Work right below it. When a route has been selected, the route instructions is displayed at the top when Android Auto is on the phone and to the left when connected to the car's display as seen below:

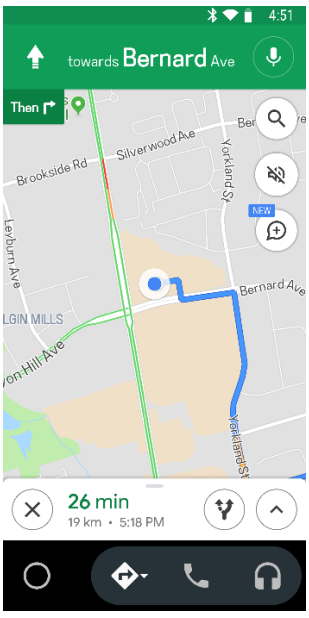

Figure 4: Navigation

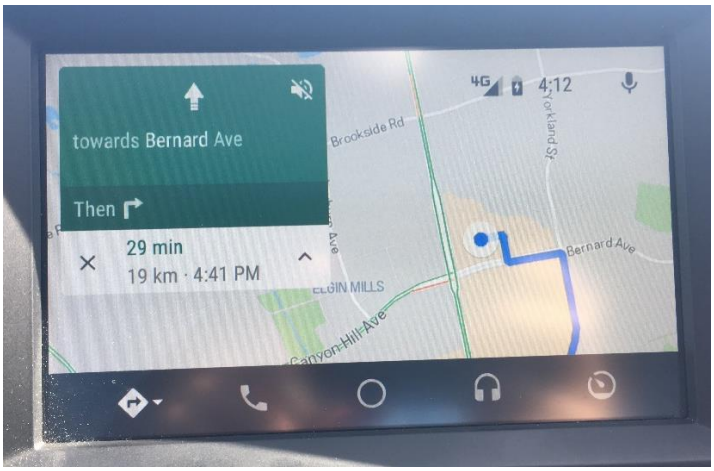

Figure 5: Navigation Route on Phone

In terms of memorability, this is not a consistent design aspect as the location has changed even though it is the same function. The graphical element on the phone's screen is also less clear and concise than that found on the car's display.

A simple solution to this is to have the same standard regardless of whether it is solely running on the phone's screen or on the car's display. Since the car's display is the cleaner design, we will use that on the phone's screen too. Below is my low fidelity sketch of what it would look like on the phone.

Figure 6: Sketch of Phone Navigation page

That would mean the phone would have to be oriented the same way as the screen, a horizontal screen instead of the default vertical. This is fine as many existing clips that mounts the phone to the car's dashboard allows for such rotational adjustments. From a usability standpoint, users will not have a hassle with this altered orientation of the phone and will find that it is satisfying that the user interface looks the same on their phone screen and the car's display, therefore encouraging familiarity, efficiency and overall usage satisfaction.

Music

Music control is by far one of the most used features other than navigation in Android Auto. The basic music interface that is currently being used on Android Auto is the following:

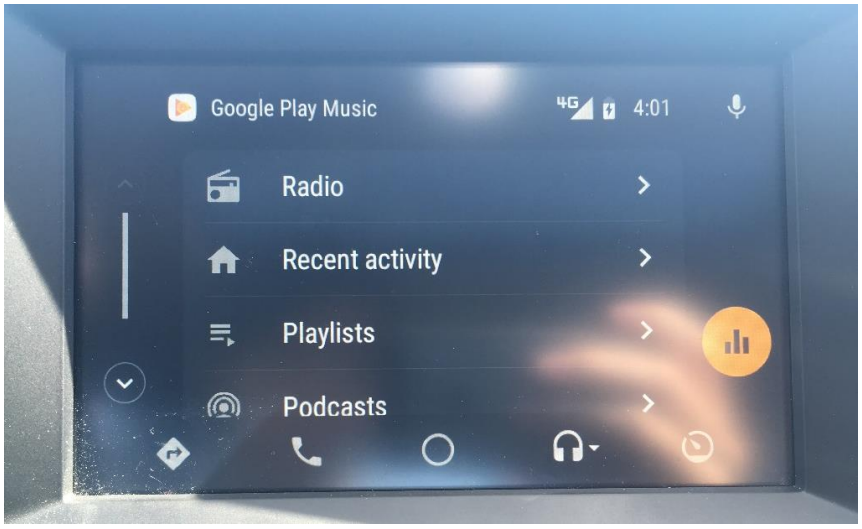

Figure 7: Music control page

As is clearly seen, the page is still too cluttered that it requires a scroll bar in order to show all the options. Also, there is no option to just play from all songs, which is unconventional as not everyone has their songs sorted into playlists. Thus, in my design, not only is the all songs option available to the user but also many of the unpopular options will not be shown unless the user clicks on the hamburger menu to bring them up, as shown below:

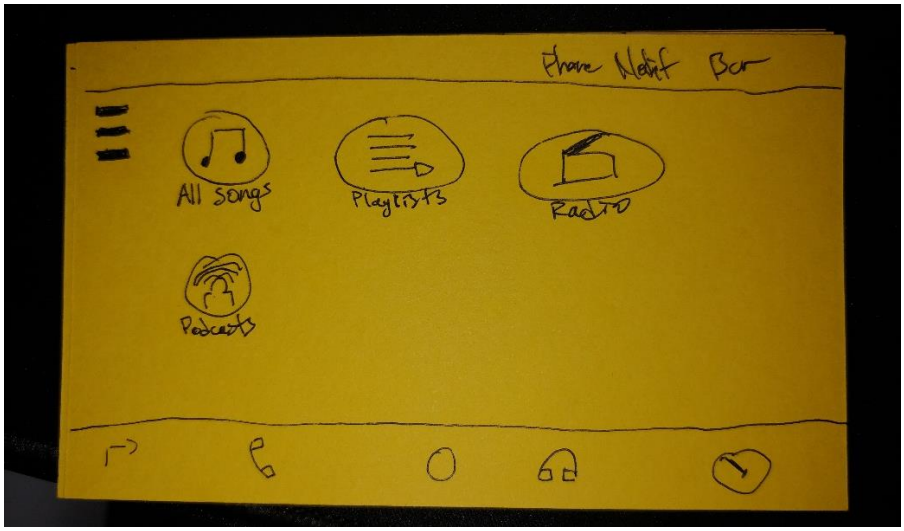

Figure 8: Sketch of new Music page

Reducing clutter and showing all the options within a single frame, users can quickly familiarize themselves with the locations of the icons. This allows users to efficiently navigate the music menu even if they are not looking at it directly, such as when driving.

Communications

Currently when the communications page is up it will list the most recent contacts called, with a small icon at the bottom that pulls up a numpad for users to dial a number. To

send a text message to someone, it is required that the user use voice commands to do this action, a major inconvenience when the voice commands do not register, or when the assistant misreads your message. Therefore, it should be allowed for passengers to type out such messages using a conventional keyboard. The solution would be having a conventional keyboard be available to the passenger on the phone's screen when connected to the car. This will allow the driver to retain concentration on navigating and driving whilst allowing the passenger to precisely type in their message as desired, reducing errors in the driving and typing components and allowing for more effective and safe way of messaging. This can be shown below in the short storyboard:

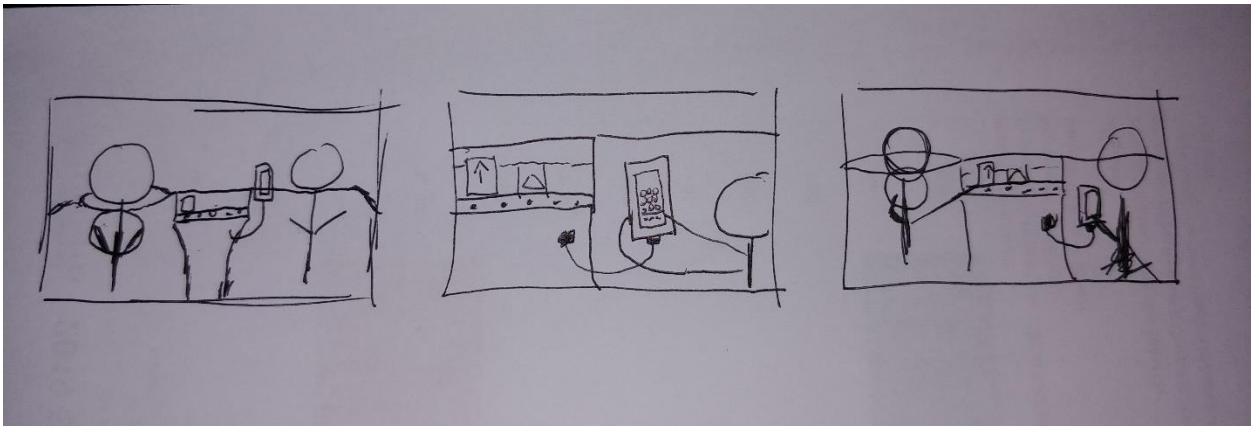

Figure 9: Storyboard of dualscreen feature/external keyboard

Splitscreen

One of the major functions that the current Android Auto lacks is splitscreen. Whenever you want to switch to another feature say the music selection page and you are currently on navigation page, the music page completely takes over the screen, meaning you lose the navigational map that is displayed for the driver (refer to figures 5 and 7 above respectively). This is a very bad design when the main aspect of Android Auto is navigation, and that is why my version of Android Auto will have such splitscreen options. Say you are in navigation page and need to bring up the music screen, you will be able to do that by pressing the music icon in the menu bar and the screen will be split in half containing your navigation screen on one side and the music screen on the other. This is illustrated in my wireframe below:

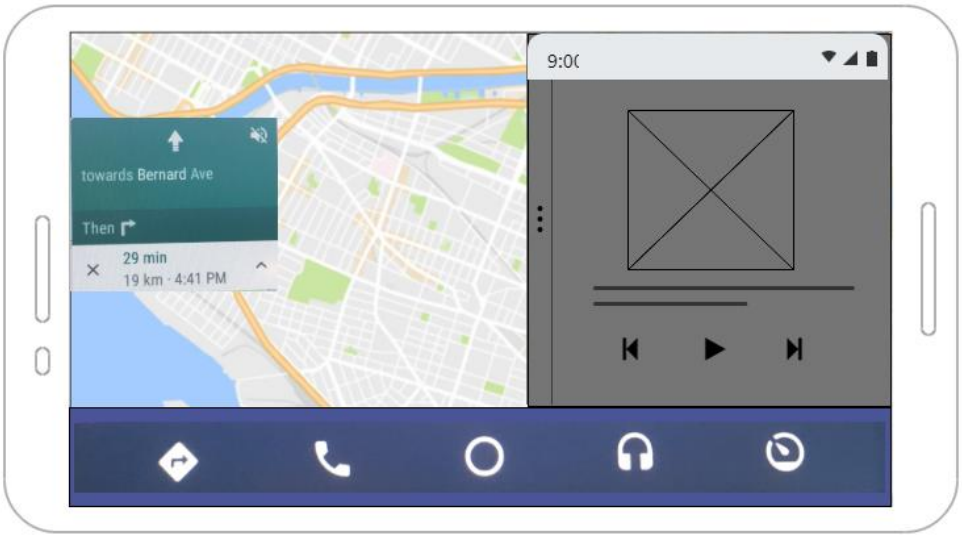

Figure 10: Wireframe of Splitscreen feature with Navigation page (left) and Music page (right)

To return the display back to navigation fullscreen the user just must tap the navigation icon in the menu bar. Likewise, if the screen is in splitscreen with navigation and music and the user wants the music page fullscreen then they must tap the music icon in the menu bar. This adheres to Nielson's usability components as toggling this function is standard between all pages of Android Auto. It is therefore easily learnable and memorizable, efficiently displaying the necessary information required by the user.

Splitscreen also has a subfunction dualscreen where passengers may choose to use another page of Android Auto on the phone without splitting the screen on the car's display if Android Auto is connected to the car and using the car's display. This allows the driver to retain full navigation authority whilst the passenger may enjoy using other functionalities of Android Auto, allowing for a safer environment for both driver and passenger. This function is even more efficient than the splitscreen function as it uses 2 screens instead of 1 and allows the driver to remain focused on the road and navigation, reducing possible errors caused by distractions. This can be illustrated with the same storyboard from figure 9 above, where the passenger is using another feature on the phone whilst the driver's navigation on the car's display remains unhindered.

Voice Control

Currently Android Auto does support voice commands with either a verbal start command like "Hey Google" or by toggling the speech button on the steering wheel when the phone is connected to the car. This is also the only way to input commands like navigate to a new location or to call someone as Android Auto locks the ability to bring up the onscreen keyboard to type when the vehicle is moving, which is good for reducing distracted driving but bad for passengers who want to use such features. Voice commands are also inefficient if it does not recognise the command and not having a keyboard to type it does not allow the user

to perform their desired action. My solution to this is allowing the option for the passenger to use the keyboard, only if the phone is connected to the car and the passenger uses the dualscreen function mentioned earlier. This way the driver stays focused on the road and the passenger can use the keyboard on the phones screen without distracting the driver thus reducing driving errors made by the driver and voice command errors by the passenger. This can be illustrated by figure 9 from above, the same storyboard as used in the communications feature.

Another design flaw that the current voice command has is its graphical interface when it is enabled. The assistance voice overlay takes a third of the screen, whilst dimming the background feature to the point where it is very difficult to see anymore.

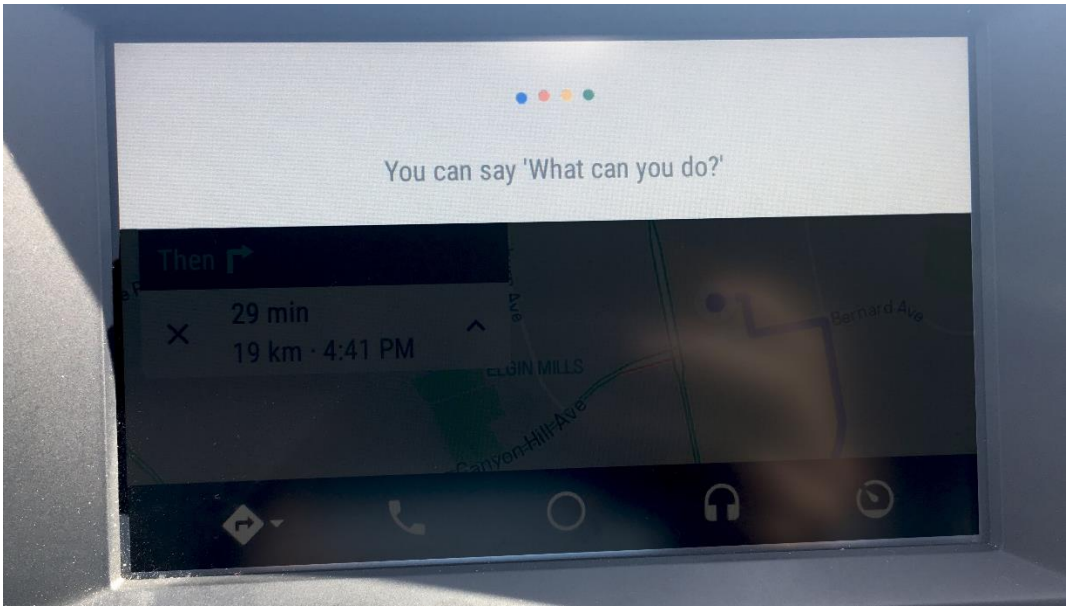

Figure 11: Current voice assistant/control overlay in Android Auto

This is a very distracting and unhelpful design where the driver relying on the navigation feature would not be able to use it for the duration voice assistant/voice control feature is actively listening for a command. An easy fix to this would be to have the voice assistant/voice control overlay not dim the background feature and reduce the graphical overlay to a small fraction of the screen that will not hinder the driver or passenger's ability to use the feature currently running on the display. As shown below in my wireframe of my version (figure 12), the graphical overlay of the voice assistant will no longer cover important information being displayed to the user and instead will only appear overtop the menu bar when in use. When the voice assistant is no longer in use, the menu bar will return to how it originally is displaying the 5 main icons as seen in figure 11.

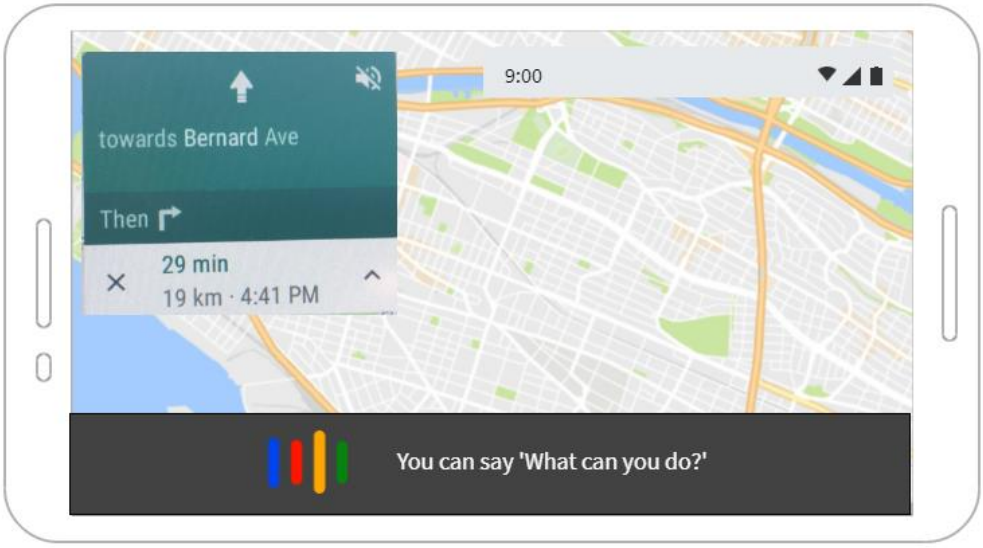

Figure 12: Wireframe of better voice assistant/control overlay

Conclusion

In conclusion, all the changes and additional functions that I made to Android Auto was made with user friendly and safety in mind. These changes will allow users to effectively and comfortably use the features in Android Auto that would promote a satisfying and safe experience.