UTM

UNIVERSITI TEKNOLOGI MALAYSIA

SECJ3553 ARTIFICIAL INTELLIGENCE

SEMESTER 1 2024/2025

PROJECT DOCUMENTATION

(ROUTESMART - AI-DRIVEN ROAD NAVIGATION APPLICATION)

SECTION: 02

LECTURER: ASSOCIATE PROF. DR. MOHD ADHAM ISA

GROUP NAME: LogiCode

NO.	NAME	MATRIC NO.
1.	KOH LI HUI	A22EC0059
2.	KOH SU XUAN	A22EC0060
3.	MAISARAH BINTI RIZAL (LEADER)	A22EC0192
4.	TANG YAN QING	A22EC0109

TABLE OF CONTENT

1.0 WORKING PROTOTYPE - BOTPRESS

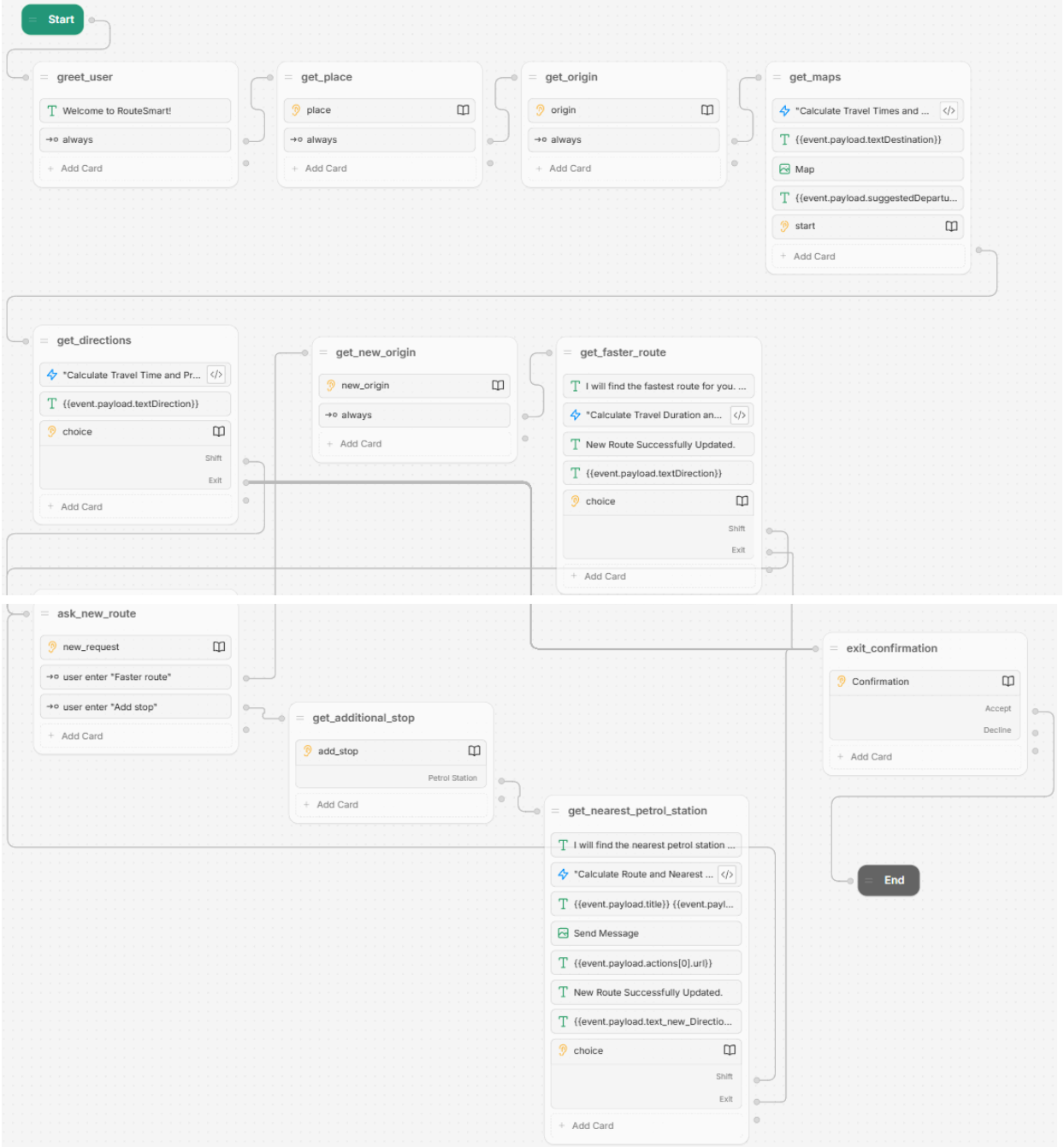


Figure 1.1 Overview of the Botpress Flow

Botpress Link:

https://cdn.botpress.cloud/webchat/v2.2/shareable.html?configUrl=https://files.bpcontent.cloud/2025/01/15/11/20250115110759-Q476XGLP.json

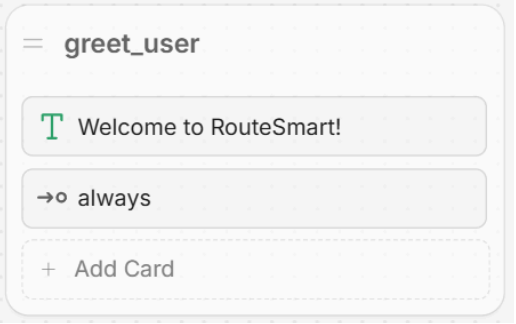


Figure 1.2 Greeting User Node

Figure 1.2 represents the "greet_user" node, which is designed to send a welcome message to the user when they start interacting with the chatbot.

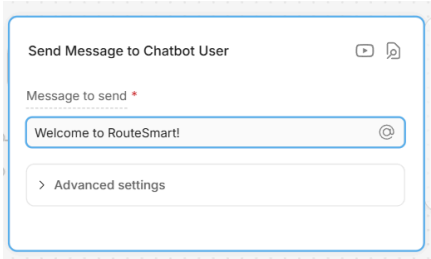


Figure 1.2.1 Send Welcome Message to Chatbot User

Figure 1.2.1 shows the configuration of the "Send Message" action within the "greet_user" node. The action sends the message "Welcome to RouteSmart!" to the user.

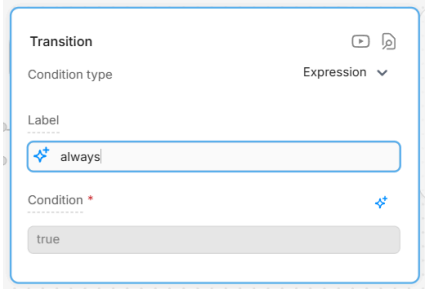


Figure 1.2.2 Transition Logic

Figure 1.2.2 highlights the transition logic configuration for the "greet_user" node. The transition is labeled "always" and the condition is set to "true", which means that the transition will always be executed regardless of the context or user input.

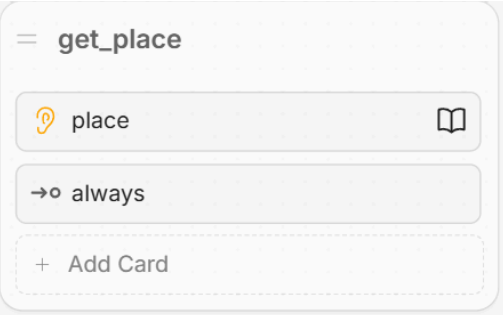


Figure 1.3 Get Place Node

Figure 1.3 represents the "get_place" node, which is designed to ask the user for their destination.

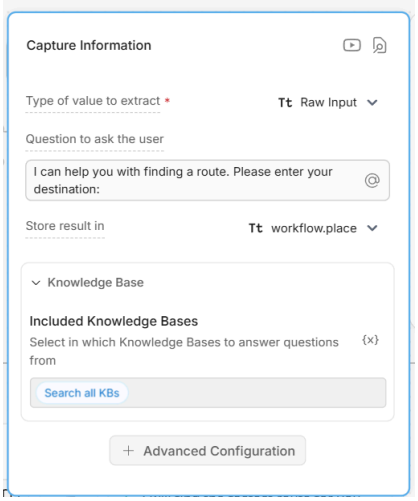


Figure 1.3.1 Capture Destination Information

Figure 1.3.1 illustrates the configuration details for capturing user input - destination. The type of value to extract is set to "Raw Input", which means the chatbot will store the user's exact response without additional processing. The prompt displayed to the user is: "I can help you with finding a route. Please enter your destination." The response is stored in the variable "workflow.place" and will be used for further processing in finding routes.

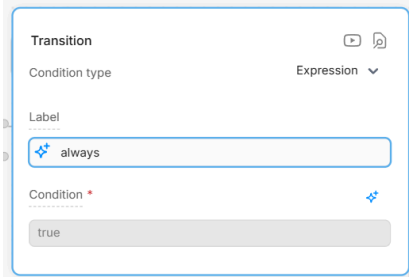


Figure 1.3.2 Transition Logic

Figure 1.3.2 shows the setup of a transition for the "get_place" node within the chatbot workflow. The label is set to "always", and the condition is set to "true". This means the transition is unconditional and will always be executed.

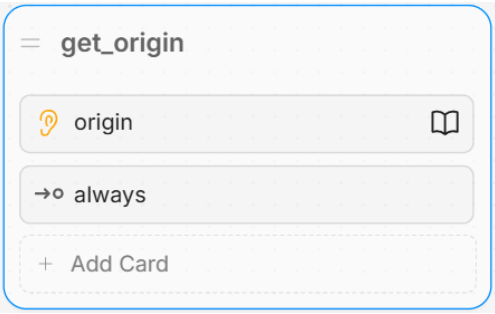


Figure 1.4 Get Origin Node

Figure 1.4 represents the "get_origin" node, which is designed to capture the user's origin location.

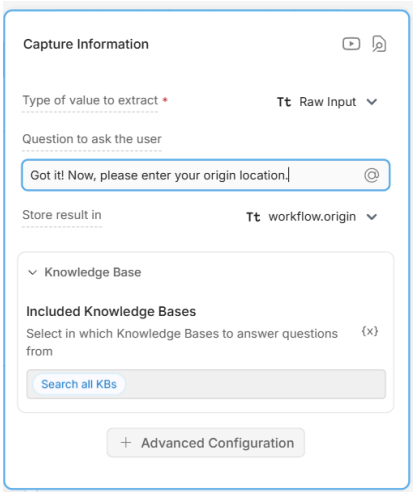


Figure 1.4.1 Capture Origin Location Information

Figure 1.4.1 highlights the configuration for capturing the user's origin location. The type of value to extract is set to "Raw Input", ensuring the chatbot stores the user's unprocessed response. The question displayed to the user is: "Got it! Now, please enter your origin location." The captured input is saved in the variable "workflow.origin" for further processing, such as computing the routes, calculating the distance and time.

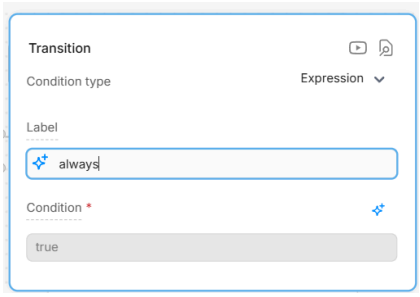


Figure 1.4.2 Transition Logic

Figure 1.4.2 shows the transition setup for the "get_origin" node. The label is set to "always", and the condition is defined as "true". This means the transition will always be triggered.

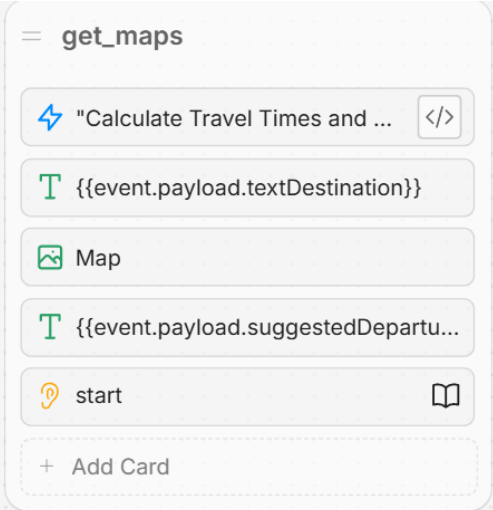


Figure 1.5 Get Maps Node

Figure 1.5 represents the "get_maps" node, designed to generate essential travel details, including the destination's static map, suggested departure time, and estimated arrival time. It retrieves input data, such as the origin and destination, and processes it through code to produce the necessary outputs, which are then used to display the information to the user.

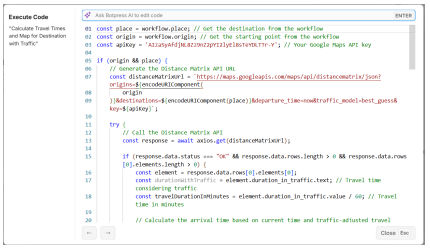


Figure 1.5.1 Travel Time and Map Generation

Figure 1.5.1 illustrates how the Execute Code card calculates travel details such as estimated arrival and suggested departure times using Google Maps APIs. It retrieves the origin and destination from the workflow, constructs a Distance Matrix API request, and processes the response to obtain travel duration, factoring in traffic. The card calculates arrival time by adding the travel duration to the current time and determines departure time by subtracting the travel duration from the desired arrival time, displaying "Now" if close to the current time. It also generates a static map URL showing the destination marker. All details, including the map, suggested departure time, and arrival time, are stored in the event payload, with fallback messages for errors or invalid data.

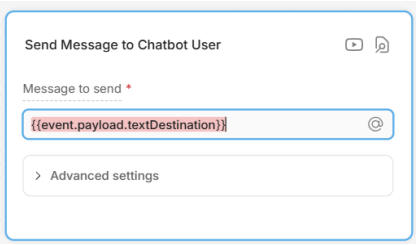


Figure 1.5.2 Send Text Message to Chatbot User

Figure 1.5.2 illustrates that the chatbot sends a message to the user with the text "Here is the map for {destination}:". The "{{event.payload.textDestination}}" placeholder retrieves this predefined message, which includes the destination name. This ensures that the chatbot provides a clear introduction before displaying the map.

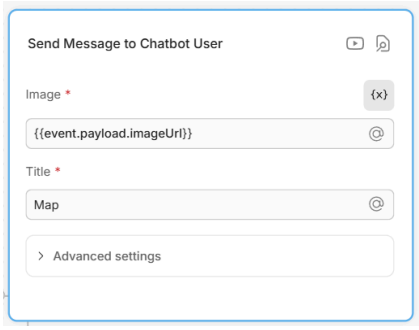


Figure 1.5.3 Send Map to Chatbot User

Figure 1.5.3 shows that the chatbot sends a static map image of the destination to the user. The "{{event.payload.imageUrl}}" placeholder retrieves the URL of the static map generated through the code in the previous card. The message includes a title "Map" to indicate the content's purpose.

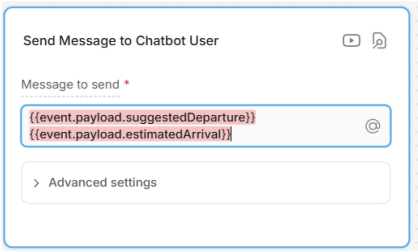


Figure 1.5.4 Send Time Information to Chatbot User

Figure 1.5.4 shows that the chatbot sends a message with the suggested departure time and the estimated arrival time to the user. The placeholders "{{event.payload.suggestedDeparture}}" and "{{event.payload.estimatedArrival}}" display the message with relevant travel information.

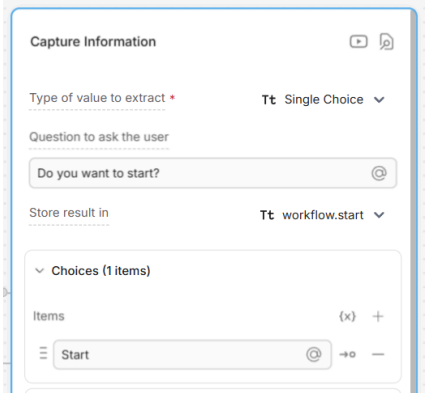


Figure 1.5.5 Capture User Confirmation to Start Heading to the Destination

Figure 1.5.5 shows that the chatbot asks the user whether they want to start heading to the destination. Using a single-choice input, the chatbot prompts the user with "Do you want to start?" and stores the response in "workflow.start". Once the user selects "Start," the workflow proceeds to the next node to send directions and related travel information.

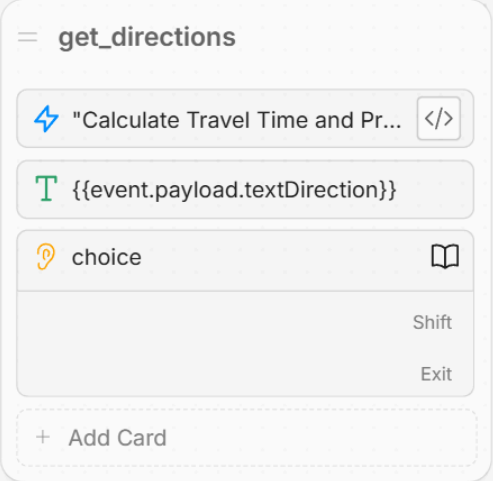


Figure 1.6 Get Directions Node

The get_directions node integrates custom logic to calculate distance, travel time and provide navigation directions to the user. The results of the calculation and directions are dynamically displayed to the user.

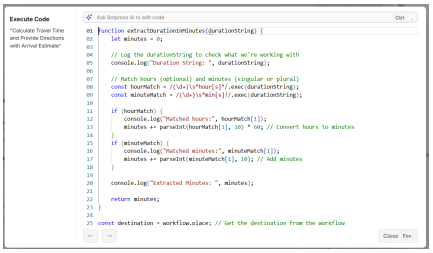


Figure 1.6.1 Real-Time Directions and Travel Time Calculation

Figure 1.6.1 demonstrates how this code retrieves and processes route information using the Google Directions API. It calculates the travel duration, distance, and estimated arrival time while considering real-time traffic conditions. The code extracts the origin and destination from the workflow, queries the API, and processes the response to provide clear, step-by-step directions for the journey. The estimated arrival time is calculated and formatted for user-friendly display. The final output, including distance, travel duration, arrival time, and navigation steps, is stored in the event payload for seamless integration into the bot's response.

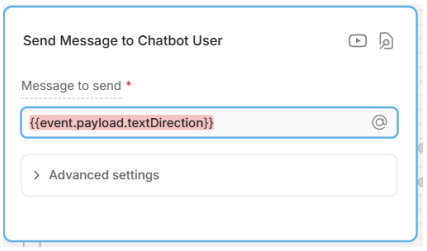


Figure 1.6.2 Send Directions Message to Chatbot User

Figure 1.6.2 illustrates how the chatbot sends a message to the user. The message to send field contains the placeholder "{{event.payload.textDirection}}", which displays distance, duration, estimated arrival time and directions to the user.

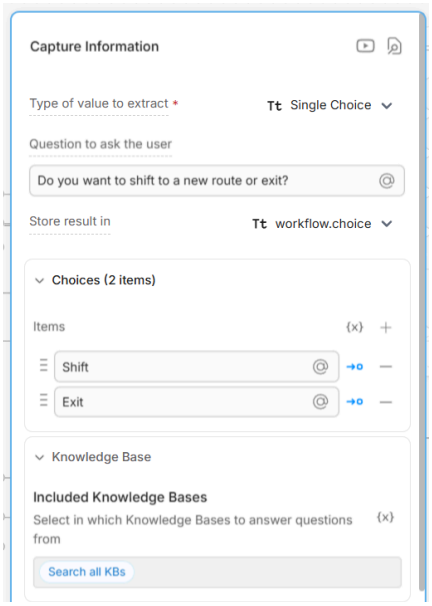


Figure 1.6.3 Capture User Choice Information

Figure 1.6.3 displays a configuration that captures user input as a "Single Choice". The question presented to the user is: "Do you want to shift to a new route or exit?" The two available options are labeled as "Shift" and "Exit". The user's response is stored in the variable "workflow.choice". This step allows the chatbot to collect user preferences for the next action.

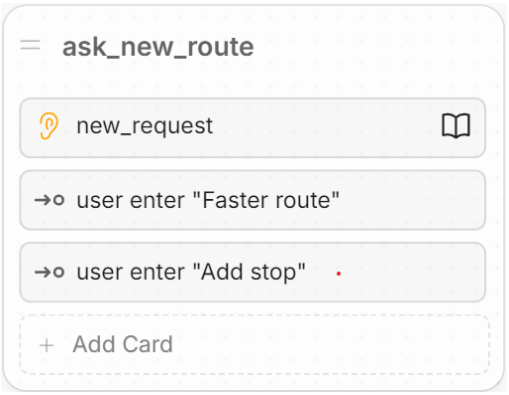
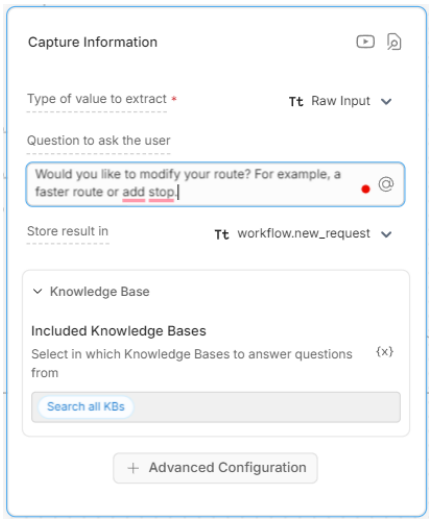
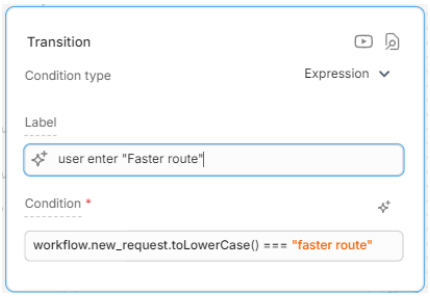


Figure 1.7 Ask New Route Node

The ask_new_route flow handles user inputs related to modifying a route. Users can request a "Faster route" to optimize travel time or "Add stop" to include an additional location in the navigation.

 ***Figure 1.7.1 Capture User Choice Information***	Figure 1.7.1 is designed to prompt the user with the question, "Would you like to modify your route? For example, a faster route or add stop." It captures the user's raw input and stores it in the variable workflow.new_request for later evaluation. It serves as the starting point for collecting user responses that drive the transitions defined in the next two cards.
 ***Figure 1.7.2 Transition Logic for "Faster route"***	Figure 1.7.2 handles a transition that is triggered when a user requests a "Faster route." The condition checks whether the user's input, stored in workflow.new_request and converted to lowercase, matches the string "faster route." If this condition is met, the transition is triggered, allowing the workflow to proceed with this specific request.

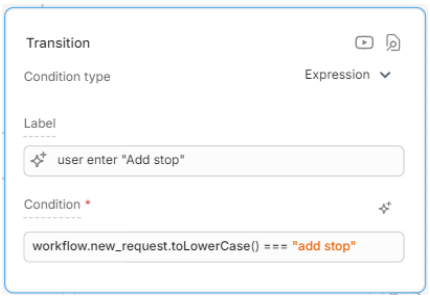


Figure 1.7.3 Transition Logic for "Add Stop"

Figure 1.7.3 defines a transition in the workflow triggered when a user requests to "Add stop." It includes a condition that evaluates whether the user's input, stored in the variable workflow.new_request, matches the exact string "add stop" after converting it to lowercase. If the condition is true, this transition is activated, guiding the workflow to respond accordingly.

Figure 1.8 Get New Origin Node

The get_new_origin node prompts the user for their origin location and transitions to the next step.

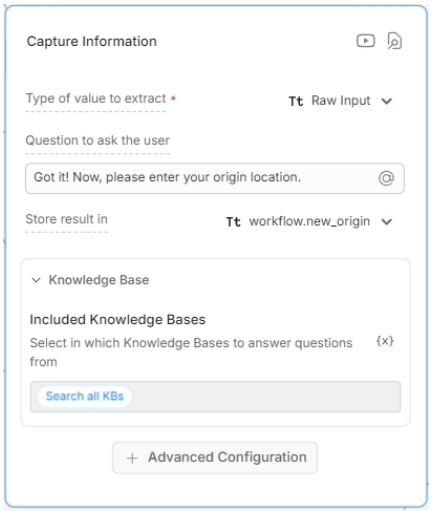


Figure 1.8.1 Capture User Choice Information

Figure 1.8.1 prompts the user to enter their origin location with the question, "Got it! Now, please enter your origin location." It captures the user's raw input and stores it in the variable workflow.new_origin.

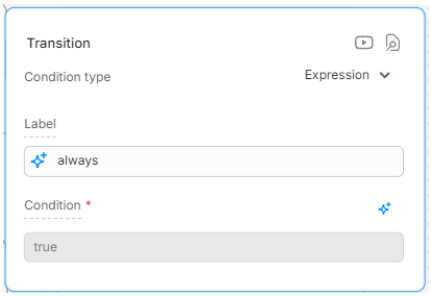


Figure 1.8.2 Transition Logic

Figure 1.8.2 defines a transition that is always active. Its condition is set to true, meaning it will trigger without any specific input or criteria.

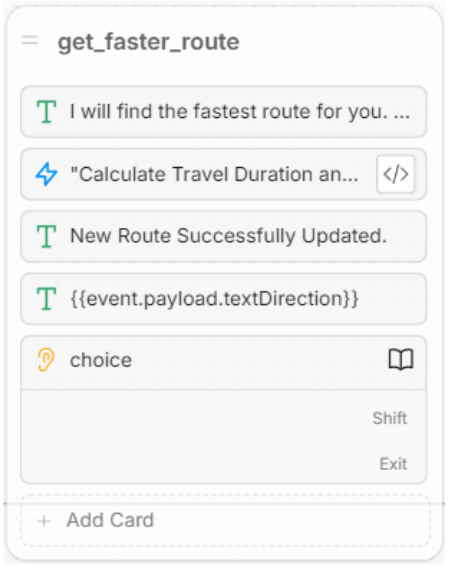


Figure 1.9 Get Faster Route Node

The get_faster_route node calculates and updates the route to provide the fastest option, confirms the update to the user, and handles the next steps based on their choice.

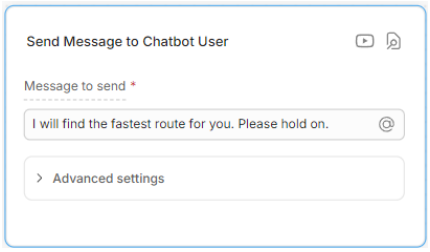


Figure 1.9.1 Send Notification Message to User

Figure 1.9.1 sends a message to the user stating, "I will find the fastest route for you. Please hold on." It acts as a notification to inform the user that the system is processing their request for a faster route.

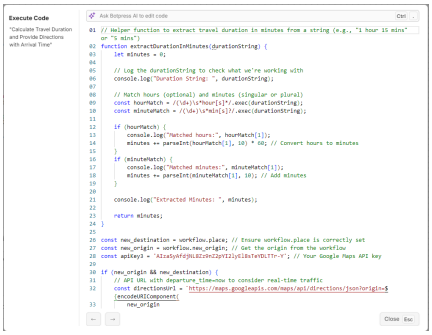


Figure 1.9.2 Calculate Travel Duration and Provide Directions with Arrival Time

Figure 1.9.2 is to calculate travel duration and provide updated route directions. It extracts travel times in minutes from a string, processes user inputs, and makes an API call to Google Maps to retrieve the fastest route considering real-time traffic. This logic ensures the updated route is accurate and time-efficient.

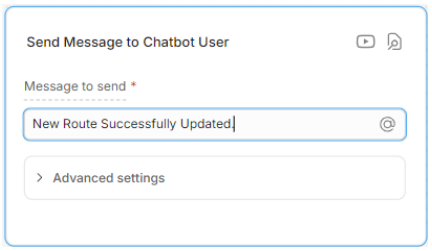


Figure 1.9.3 Send Confirmation Message to User

Figure 1.9.3 sends a confirmation message to the user, "New Route Successfully Updated," after the system processes the request and calculates the updated route. It assures the user that their request has been completed successfully.

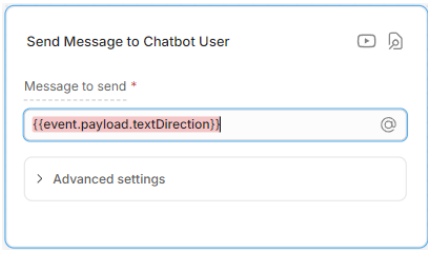


Figure 1.9.4 Display Updated Route Directions

Figure 1.9.4 displays the updated route directions to the user using the placeholder {{event.payload.textDirection}}. It ensures the user receives the detailed step-by-step directions for their updated route.

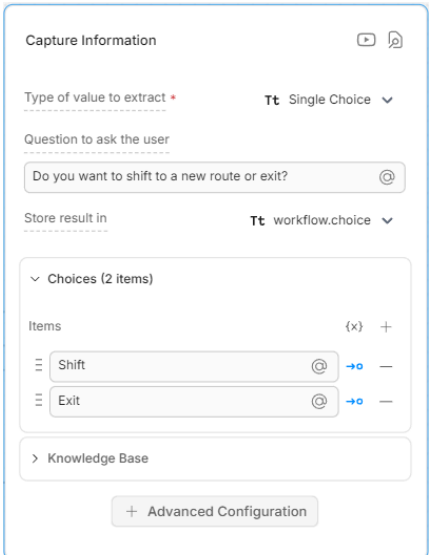


Figure 1.9.5 Capture User Response for Shift or Exit

Figure 1.9.5 captures the user's response to a multiple-choice question: "Do you want to shift to a new route or exit?" The options provided are "Shift" and "Exit," which are stored in the variable workflow.choice to determine the next action in the workflow.

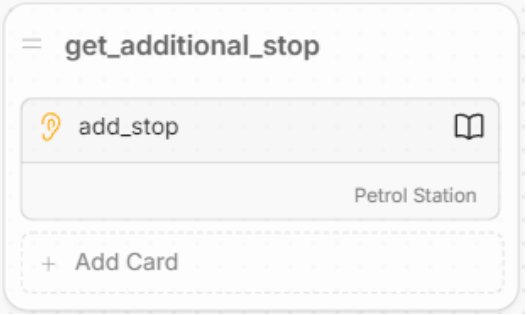


Figure 1.10 Get Additional Stop Node

The get_additional_stop node handles adding an extra stop to the route, such as a petrol station, by capturing the user's input and updating the workflow with the specified stop.

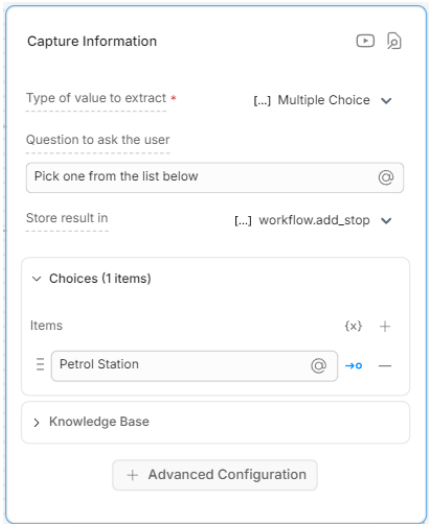


Figure 1.10.1 Capture User Input for Additional Stop

Figure 1.10.1 captures the user's input to add an additional stop to their route. It presents a multiple-choice prompt with a question, "Pick one from the list below," and currently provides the option to select a "Petrol Station." The user's selection is stored in the variable workflow.add_stop, which can later be used to update the route or provide specific directions to the chosen stop.

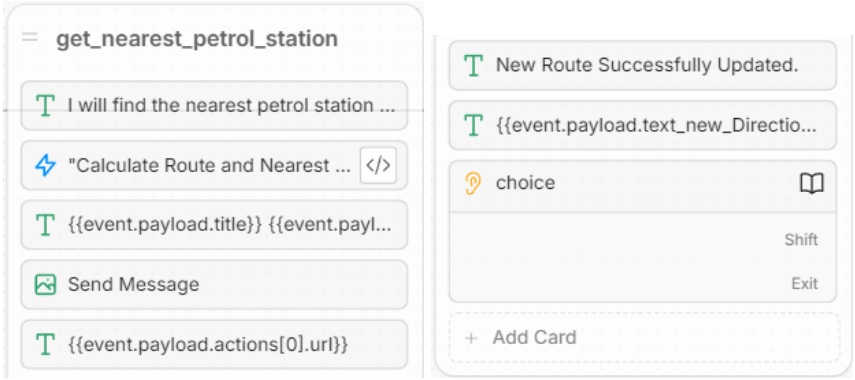
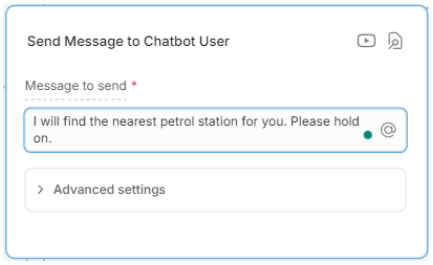
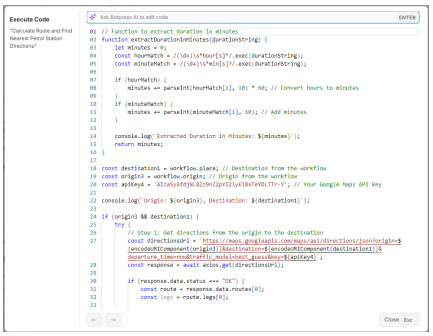


Figure 1.11 Get Nearest Petrol Station Node

The get_nearest_petrol_station node identifies and provides directions to the nearest petrol station. It calculates a new route, updates the user with the necessary information, and allows them to decide whether to proceed with the updated route or exit.

 ***Figure 1.11.1 Send Message to User***	Figure 1.11.1 is designed to inform the user that the system is working on their request. It sets the expectation that the chatbot is actively searching for the nearest petrol station. This helps enhance the user experience by acknowledging their query while providing a clear and polite communication of the next steps.
 ***Figure 1.11.2 Calculate Route and Find Nearest Petrol Station Directions***	Figure 1.11.2 is to calculate the route and retrieve the nearest petrol station. The code extracts travel duration in minutes, uses workflow variables to identify the origin and destination, and integrates with the Google Maps API. It fetches directions, calculates optimal routes, and processes the API response to extract details like distance, duration, and navigation steps, which can then be passed on to the user in a structured manner.

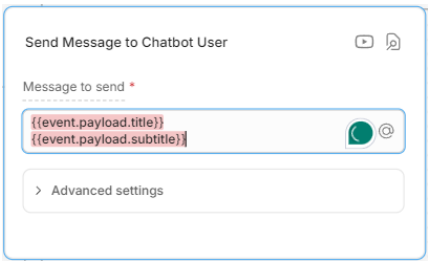
<table>
<tr>
<td> 

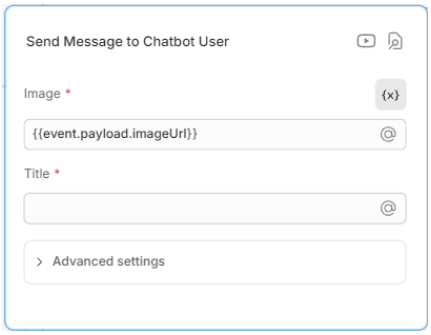
Figure 1.11.3 Display Address of Nearest Petrol Station</td>
<td>Figure 1.11.3 is configured to send a message to the user containing dynamic content pulled from the payload, specifically the route's title and subtitle. These values are placeholders populated from the API response, allowing the chatbot to provide route-specific details in a personalized and professional format.</td>
</tr>
<tr>
<td> 

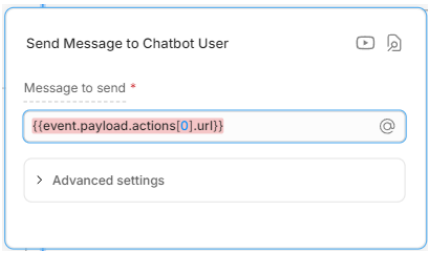
Figure 1.11.4 Display Map of Nearest Petrol Station</td>
<td>Figure 1.11.4 sends a message to the user containing an image of the route. The image URL is dynamically fetched from the event payload. This visualization of the route improves user experience by providing a clear, graphical representation of the navigation details.</td>
</tr>
<tr>
<td> 

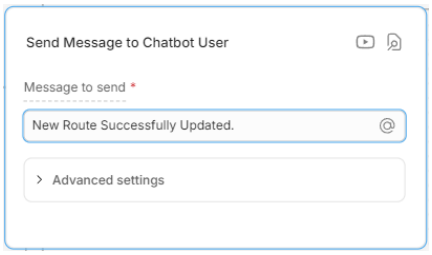
Figure 1.11.5 Display URL of the Nearest Petrol Station</td>
<td>Figure 1.11.5 sends the URL of the calculated route to the user, allowing them to open and view the route in a web-based map interface. The URL is fetched dynamically from the payload, enabling the user to directly access navigation or route details for further action or review.</td>
</tr>
<tr>
<td> 

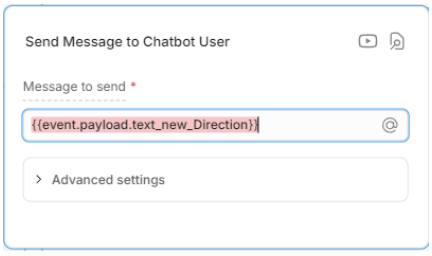
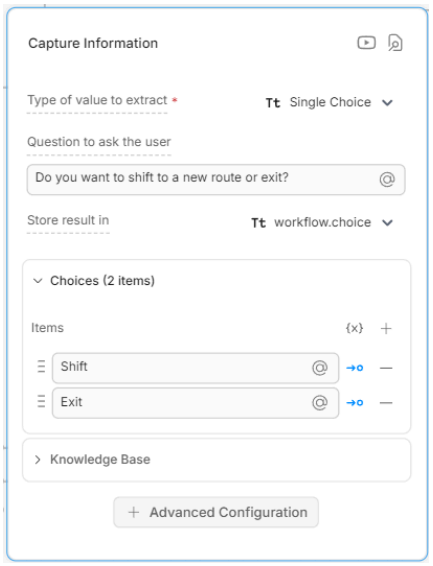
Figure 1.11.6 Send Confirmation Message to User</td>
<td>Figure 1.11.6 serves as a confirmation message to notify the user that the route has been successfully updated. It is the final acknowledgement in the process, ensuring closure for the user and confirming that their request to calculate a new route was completed successfully.</td>
</tr>
<tr>
<td> 

Figure 1.11.7 Display Updated</td>
<td>Figure 1.11.7 communicates updated navigation directions to the user in a text format. The message uses a variable, text_new_Direction, which is dynamically populated with the latest instructions or steps for navigating to the petrol station. This ensures the user</td>
</tr>
</table>

Route with The Additional Stop	receives accurate and actionable route guidance in real-time.
 ***Figure 1.11.8 Capture User Choice Information***	Figure 1.11.8 captures the user's response to a multiple-choice question: "Do you want to shift to a new route or exit?" The options provided are "Shift" and "Exit," which are stored in the variable workflow.choice to determine the next action in the workflow. Advanced configurations allow further refinement of user input handling.

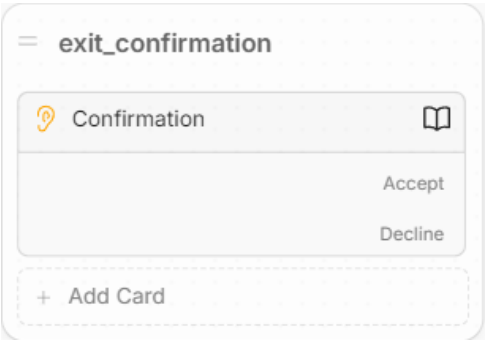


Figure 1.12 Exit Confirmation Node

The exit_confirmation node presents the user with a confirmation prompt when they are about to exit. It offers two clear options: "Accept" to confirm the exit and proceed, or "Decline" to cancel the action and remain in the current flow.

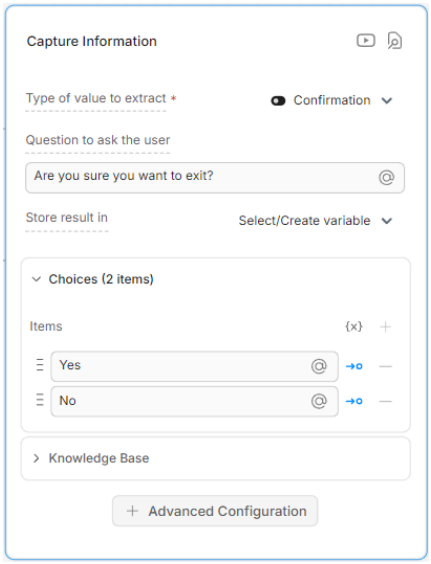


Figure 1.12.1 Capture User Input Confirmation

Figure 1.12.1 prompts the user with the question, "Are you sure you want to exit?", and provides two response options: Yes and No. The user's selection is then stored in a specified variable for further use in the workflow. This configuration ensures that the user's intent to exit is explicitly confirmed or declined, enabling the bot to proceed accordingly.

2.0 WORKING USER INTERFACE (UI) - FIGMA

Figure 2.1 Landing Page

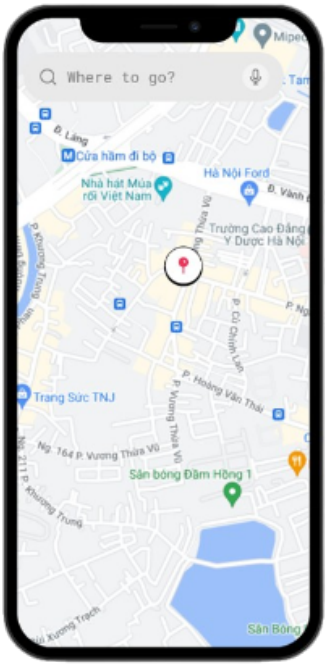


Figure 2.2 Navigation Home Page

Figure 2.3 Voice Command Page

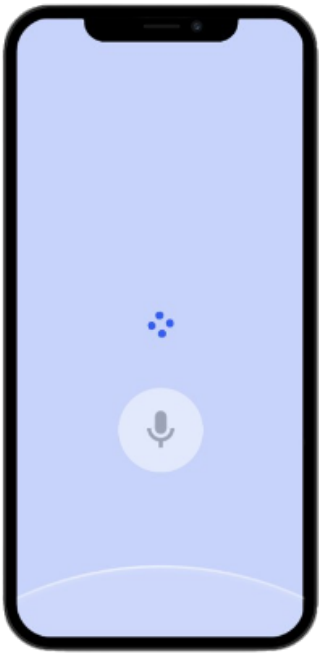


Figure 2.4 Listening Page

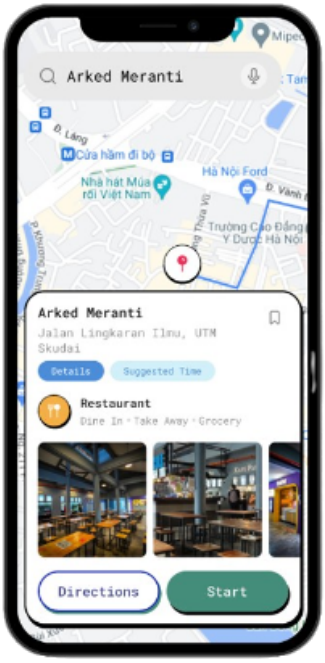


Figure 2.5 Location Overview Page

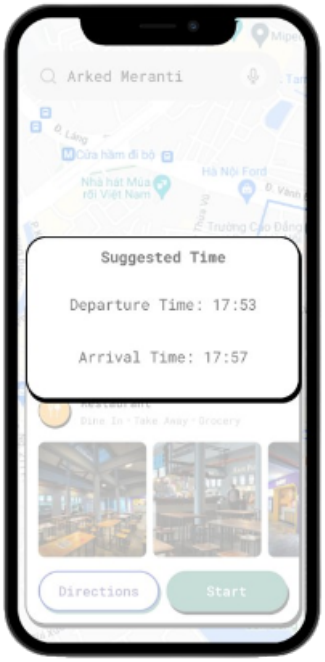


Figure 2.6 Suggested Time Page

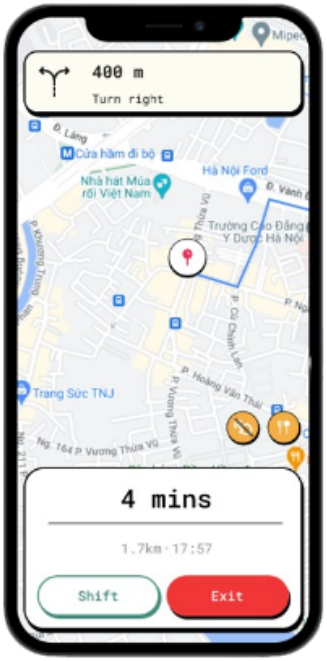


Figure 2.7 Route Ongoing Page

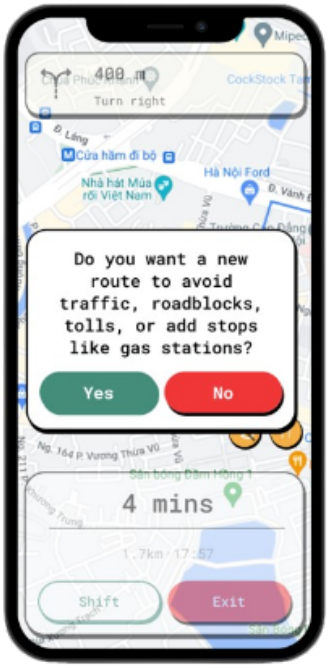


Figure 2.8 New Route Confirmation Page

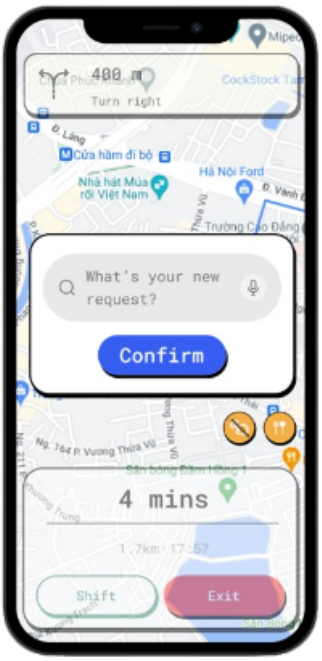


Figure 2.9 New Route Request Page

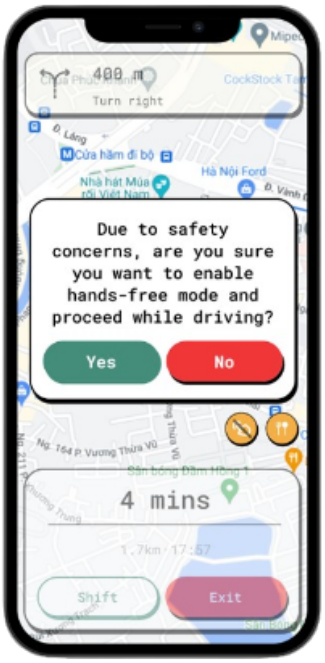


Figure 2.10 Hands-Free Mode Warning Page

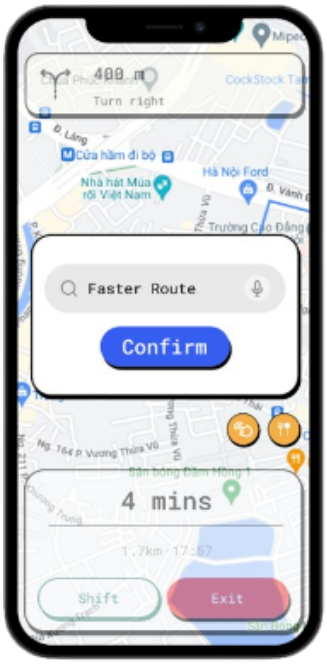


Figure 2.11 New Request Page

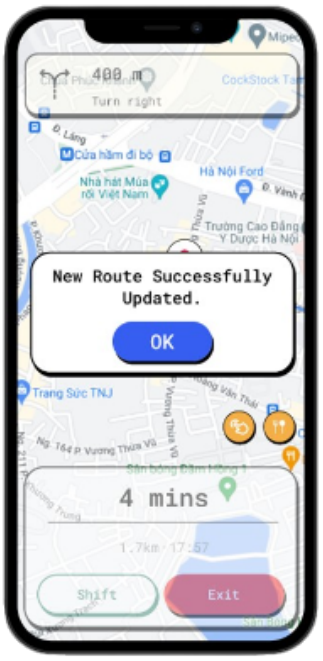


Figure 2.12 Route Updated Notification Page

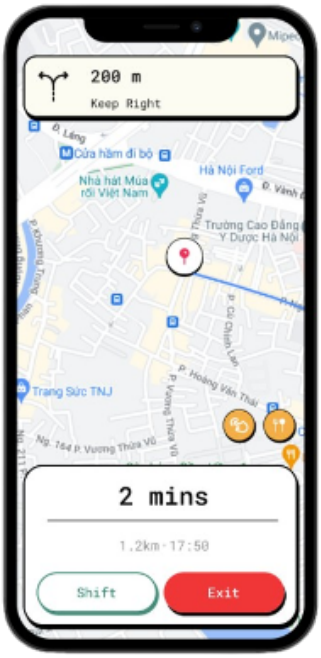


Figure 2.13 Live Trip Progress Page

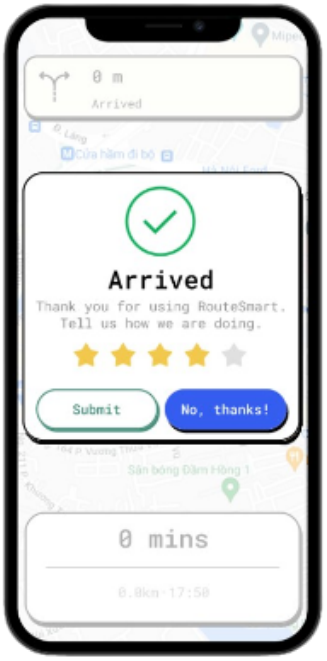


Figure 2.14 Route Completion and Rating Page

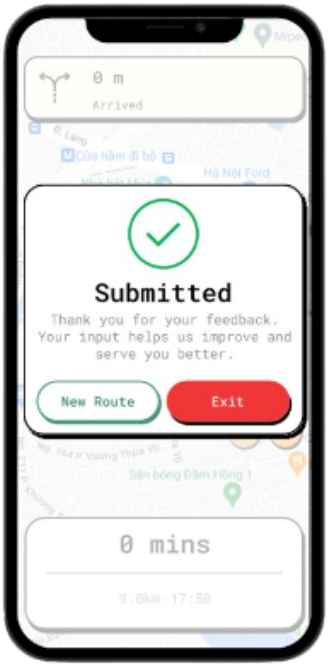


Figure 2.15 Rating Completion Page

Figma Link:
https://www.figma.com/design/ixZuodgAK575XlljDfPcKg/RouteSmart?node-id=0-1&t=HKZQHFpDhsIFRmtI-1

Explanation of User Flow in the Prototype Interface

The user flow of the RouteSmart prototype interface begins on the Landing Page (Figure 2.1), where users are introduced to the application and prompted to tap "Get started" to initiate their navigation experience. This action leads them to the Navigation Home Page (Figure 2.2), where they can input their destination using the search bar or the voice input. If the user chooses to have voice input, it will proceed to the Voice Command Page (Figure 2.3), allowing them to give commands verbally, with the system actively listening on the Listening Page (Figure 2.4).

Once a destination is selected, the Location Overview Page (Figure 2.5) provides details about the location, suggested time and options to start navigation. By tapping the “Suggested Time” button, the Suggested Time Page (Figure 2.6) is then presented to show optimal departure and arrival times based on current traffic conditions. As navigation begins, the Route Ongoing Page (Figure 2.7) displays real-time instructions, estimated time taken, distance and estimated arrival times.

If route adjustments are necessary due to traffic or other factors, the user can select the “Shift” button, leading to the New Route Confirmation Page (Figure 2.8) to alert the user. The user is then led to the New Route Request Page (Figure 2.9) to request new routes. Similar to the Navigation Home Page (Figure 2.2), the user can input their new request by using the search bar or the voice input. However, this time, for safety purposes, if the user wants to use voice input, the Hands-Free Mode Warning Page (Figure 2.10) is prompted to enable the user to enable hands-free mode while driving. Upon inputting, as shown in New request Page (Figure 2.11), the input field will be filled in with the request for the user to check on the request as if it is the desired one. Once the new route request is checked and confirmed, the Route Updated Notification Page (Figure 2.12) assures users of successful updates. Throughout the journey, the Live Trip Progress Page (Figure 2.13) provides ongoing trip details.

Upon reaching the destination, the Route Completion and Rating Page (Figure 2.14) invites users to rate their experience, with the Rating Completion Page (Figure 2.15) confirming feedback submission. This flow ensures a seamless, user-friendly navigation experience, prioritizing real-time adaptability, safety, and user preferences.