Name: Javeria Baloch
Semeter: BCaS semeter 1
Project Name: My Travel App

Outline

Personas:

Izaa Salar

Age: 28

Gender: female

Pression: student

Bio:

Izaa is done with her master of economics and before going for a PhD she want to make a foreign tour

GOALS AND NEEDS:

1. Want to enjoy her life know.
2. Looking for a suitable place to enjoy.
3. Love adventures.

Izaa MOTIVATIONS:

1. Want to try new things.
2. Want to enjoy every day for her life.

Izaa VEXATION:

1. Tired of long academic journey.

Izaa QUOTES:

1. “Live for once, let’s enjoy every day”

Aslam saleem

Bio:

Aslam spent his whole life in army service and fighting wars. Now he wants to enjoy sometime with his family and planning for a foreign tour with them

GOALS AND NEEDS:

1. Want to enjoy his retirement.
2. Looking for a suitable place to enjoy with family.
3. Want to live with his family.

Aslam saleem MOTIVATIONS:

1. 1. Want to enjoy every day for his l ife.

Aslam saleem QUOTES:

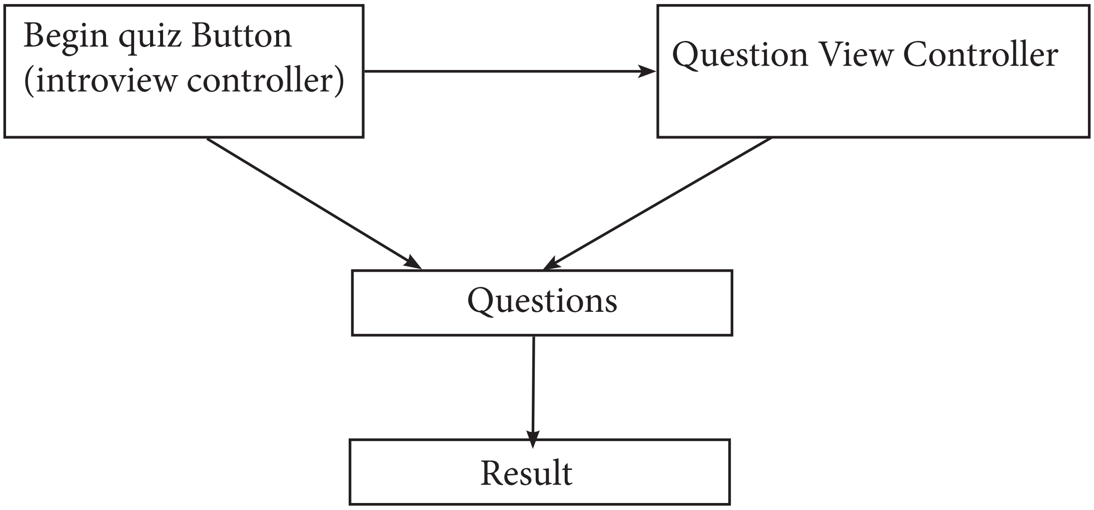
1. Tired of long army service journey.
2. Don’t want to go away from his family.

Age: 76

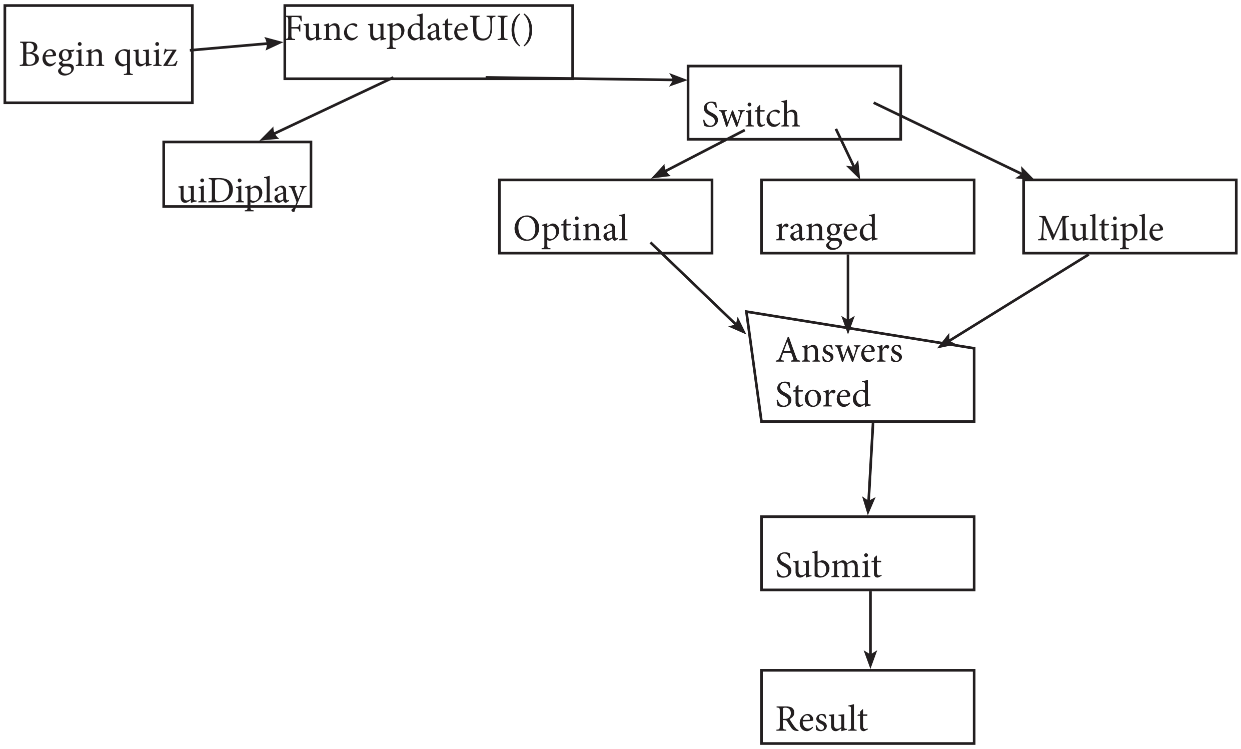
Gender: male

Pression: Retired military officer

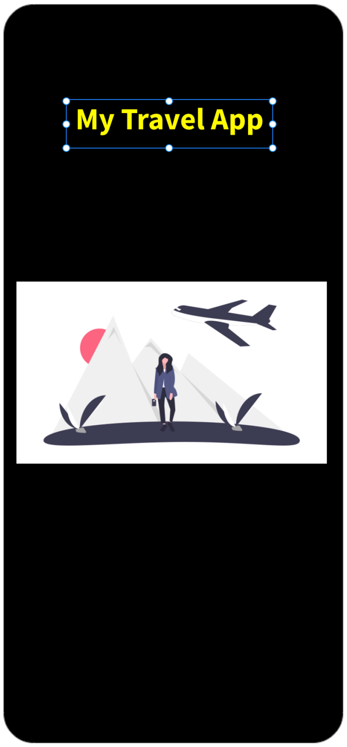
Flow of program:



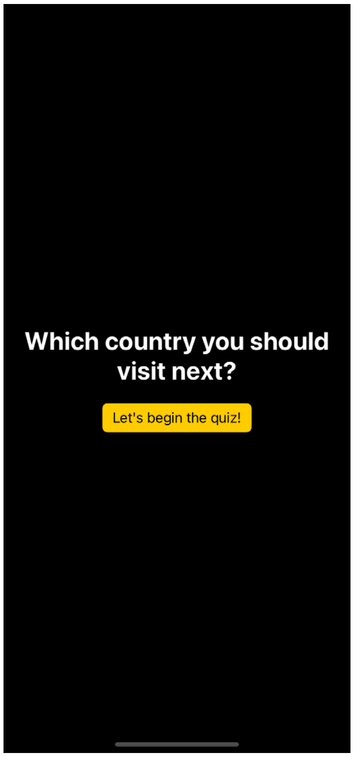
Functionalty of program:



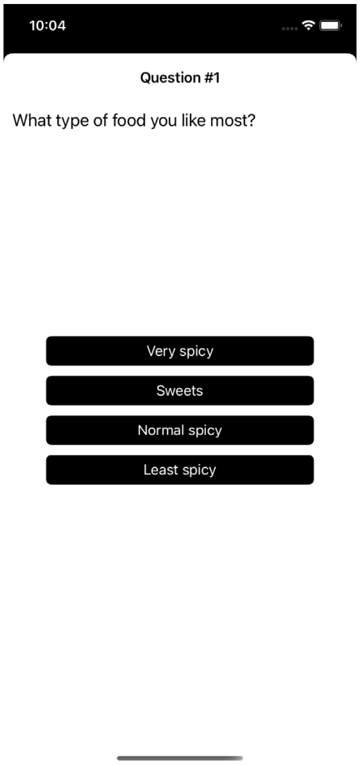
wireframes and mockups



launch screen


Which country you should visit next?

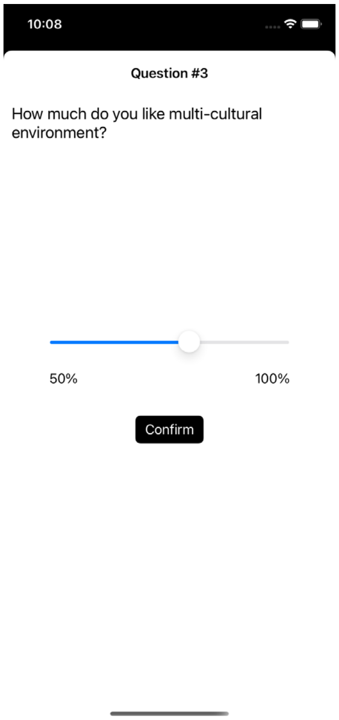
Intro screen



Question #1

What type of food you like most?

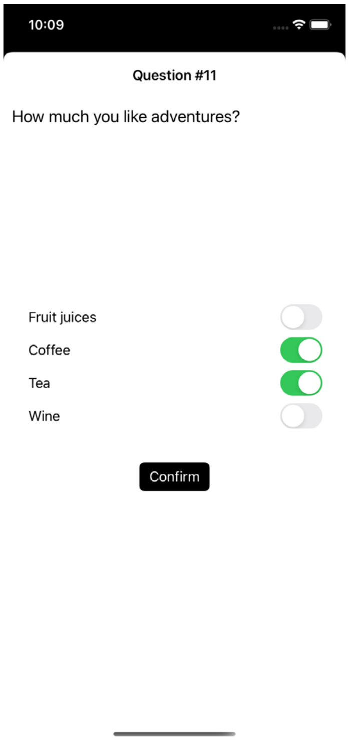
optional question screen



Question #3

How much do you like multi-cultural environment?

ranged question screen



Question #11

How much you like adventures?

Fruit juices	○
Coffee	●
Tea	●
Wine	○

multiple choice question screen

Selected Countries

1. Iceland
2. Netherlands
3. India
4. Mexico
5. Brazil
6. Uruguay
7. Iran
8. Finland
9. Oman
10. Nepal
11. Maldives
12. Egypt
13. Indonesia
14. Turkey
15. New Zealand
16. Canada

Questions:

1. What type of food you like most?(optional)

a. Highly spicy (Mexico)
b. Sweets (Netherlands)
c. Normal spicy (India)
d. Least spicy

2. Your favorite whether?(optional)

a. Extreme Cold (Island)
b. Cold (Canada)
c. Normal (Turkey)
d. Warm (Oman)
e. Windy (New Zealand)

3. How much do you like multi-cultural environment?(ranged)

a. 25(Uruguay)
b. 50 (Oman)
c. 75 (Turkey)
d. 100 (Canada)

4. How much do you like multi-religious environment? (ranged)

a. 25 (Oman)
b. 50 (Turkey)
c. 75(Canada)
d. 100 (India)

5. How much do you like multi-lingual environment? (ranged)
a. 25 (Brazil)
b. 50 (Mexico)
c. 75(India)
d. 100(Indonesia)

6. What activity you like most? (multiple)
a. Walking (Netherlands)
b. Sleeping (New Zealand)
c. Talking (Indonesia)
d. Studying (India)

7. Do you like to live in an area mixture of Asian and European culture? (optional)
a. love them (Turkey)
b. Love Asian most (India)
c. Love European (Island)
d. None (Mexico)

8. Do you like snow? (optional)
a. Love it (Island)
b. I am okay with it (Canada)
c. Don't like it (India)
d. I Don't care (Nepal)

9. How much you like adventures? (ranged)
a. 100 (Uruguay)
b. 75(Nepal)
c. 50(Egypt)
d. 25(Maldives)
10. Are you good in high altitudes? (ranged)
a. 100 (Nepal)
b. 75(Mexico)
c. 50(India)
d. 25(Iran)

11. What do you like to drink? (multiple)
a. Fruit juices (Canada)
b. Coffee (Finland)
c. Tea (India)
d. Wine (Uruguay)

12. What type of land you like most? (multiple)
a. Deserted (Oman)
b. High Buildings (Canada)
c. Muddy Houses (Iran)
d. All (India)

13. Which season you like most? (optional)
a. Winter (Canada)
b. Summer (Maldives)
c. All (Iran)
d. None (Turkey)

14. What you like to eat most? (optional)
a. Meat (Canada)
b. Vegetable (India)
c. Mixture (Oman)
d. Only Spicy (Mexico)

15. What you hate most? (optional)
a. Extreme whether (India)
b. Spicy food (Netherlands)
c. High building (Iran)
d. High temperature (Turkey)

design thoughts:

Because this app is for the post-COVID-19 circumstances, it needs to be simple and calm, so I simply used black, orange, and white colours. I wanted to show it something different at first, thus the entire screen is dark. A button of orange hue is presented in the centre at the start, with a question phrase displayed above it. As soon as the user clicks the button, the question is tech views will be called according to the question type, and the entire screen will remain white throughout the questioning process. Once the user has completed all of the questions, a large black button with white text on it will appear on the screen for submission on the top of the screen, the computer will display the name of the country it has chosen based on the answers provided by the user. In the centre of the screen, a picture of the map of the selected country will appear. At the bottom of the screen, you'll see the COVID entry limits.

Process

We have three Swift files and two Concoda Touch files in this application. One Swift file will be used for assessing responses (called "Answers"), another for storing questions (named "Questions"), and a file for travel information (named "Covid-travel-info") will be used to store information about COVID entry limits in various countries. The first Concoda file, "IntroViewController," is used for the intro screen, while the second file, "QuestionsViewController," is used for the question view controller.

Let's begin with Question.swift file it is having a struct name Question, 2 enums QuestionType and CountryList and an array variable named allQuestions.

Question struct which is having four variables, question is the first variable having a string data type and will be utilized in the question view controller for displaying the current question, second variable is the question type which is used to update the stack view according to the current question type in the question view controller, third variable is the possibleAnswers having the type of string array which really store all the answers for the current question the one last array variable is the answersReflection having the data type of CountryList enum.

Let's discuss the two enums, first one is the QuestionType, it is having three cases optional, multiple and ranged. second in enum is the CountryList, It is having the names and corresponding text as the cases of the selected countries inside it and a variable description data type of string which will return the corresponding text with the country name.

Last variable is the all questions array, it is having all the questions as the object of Question struct inside.

now discuss the "Answers.swift" file it is having three variables, and one function inside it. first variable is an array named answers having the data type of CountryList (this variable will be utilized to store the name of the countries corresponding with the user provided answers all the answers), second variable is the answerMap having a data type of is string(this variable will be utilized by question view controller to display the resulted country's map on the screen), third variable is the countryName having a data type of string(this variable will be utilized by the question view controller for displaying the country name and is also used inside the mostMatch function which we are going to discuss next)

The last method to examine in the "Answers.swift" file is mostMatch. In this function, we built a variable named nameCountDictionary, which has good keys as text and values as integer. The country names in the answers array provided by the user will become the keys of the nameCountDictionary, and every time the same key (country name) comes when will be added to the value after dictionary element, we will use a for in loop to convert the answers are in doodle name count dictionary. The second variable, mostPreferredCountry, with a data type of string, comes into play here, and we have another for loop to figure out which key represents which country. Which key indicates which country has the highest frequency in the nameCountDictionary, as determined in the previous for in loop which means that most answered value will be provided to the mostPreferredCountry variable, and the mostPreferredCountry variable will be passed to the countryName after we know which value has a greater number, indicating which country is repeated the most in the replies array.

We have variables for each selected country's covidRestrictionsInside the "covid-travel-info" swift file, and the second variable is the covidEntryRequirements, which is currently empty but will have a string to display all the information about the entry requirements to those countries, and these variables will be used to display the information on the questions view controller in the results stack view. The third element is the CountryMatch enum, which contains the strings for the switch statement in the covidStatement function (which we are going to discuss next). The covidStatement function is the last element in this file, and it sets the covidEntryRequirements according to the user-selected country and the image of the answerMap variable. To achieve this functionality, we'll use a switch statement with the current country name as the switch condition, and countryMatch enums for this case statement.

We have a button on the intro view controller, and a message such as "Which country should you visit?". We will travel to the navigation and then to the question view controller by clicking this button.

Who created you functions and variables in QuestionViewController, so it can change the view of the screen. list of these variables and functions is provided below:

- questionIndex (which is initially set 0),
- totalQuestions (which is equal to allQuestions' arrays number of elements in question swift file)
- UI Components variables
- updateUI (function)
- uiDisplay (function)
- optionsDisplayForMultipleQuestions(function)
- optionsDisplayForOptionalQuestions(function)
- sliderPressed(UI component function)
- rangedConfirmPressed(UI component function)
- limitQuestionIndex(function)
- buttonOnePressed(UI component function)
- buttonTwoPressed(UI component function)
- buttonThreePressed(UI component function)
- buttonFourPressed(UI component function)
- AnswerForMultipleChoose (UI component function)
- seeResult(UI component function)

Let's start with the updateUI function, which is used to show the question number, question, and stack view based on the question type. If the questionIndex is less than the totalQuestions variable, it will call another function called uiDisplay to hide all of these stack views in the beginning every time a question is displayed, after which the question index will be passed to the allQuestion variable array to get the currentQuestion variable value, and then using the currentQuestion variable in a switch statement to get the question type and display the stack view according to the question type.

The uiDisplay function is used to hide all of the stack views. This function is important at the beginning and also when the question is modified, because the stack is updated as well. However, all stack views should be hidden at first, and updateUI has a switch statement to deal with the question changing the step and the display of the stack views, as discussed above.

The function optionsDisplayForMultipleQuestions is used to update the options label's text based on the question. To accomplish this, we'll use the currentQuestion constant object, which will get the current question object from the allQuestions array, and then we'll use the label UI variables we discussed earlier. Because currentQuestion is an object that also contains the possibleAnswers array, which has four elements, we'll pass the index number to the possible answers array.

Single option questions are handled by the function optionsDisplayForOptionalQuestions, and we've showed that these questions will have buttons to answer them. We've generated UI button variables and linked them to the UI in the beginning of this file. All the buttons are called in the function and their text is set using the currentQuestion object's variable possibleAnswers arrays. we'll use the index number of this array to give the data to the button text.

The rangedConfirmPressed function is tied to the confirm button on the ranging questions stack view, and because the user will enter a percentage between 0 and 100 to answer the question, we have recorded the percentage values on the back end. So, to deal with that, we have a constant currentQuestion, which is a question object similar to the components covered earlier, in this question object we are also having the answerReflection array, so we'll get a country's name as a result of the user-supplied percentage and this name will added to answers array. Because we need to move on to the next question after hitting this button, the limitQuestionIndex and updateUI functions are called in the code after each case.

sliderPressed is a user interface function that is used to answer part of the range questions. As the user moves the slider, the first label text with the minimum slider value should also change with the movement of the thumb of the slider. To accomplish this, we would like this function to have a switch statement that converts the sliders input into an integer and evaluates it to show a percentage in the first label text to display the current value of the slider.

The limitQuestionIndex function is used to handle the circumstance where the user has answered all of the questions and there are no more questions to be answered. When the user has completed all of the questions, the question label text should be left blank, and a submit button should appear on the screen, allowing the user to submit the quiz and view the results.

In the single option question part we are having the buttons to deal with. So there are four buttons in the stack UI and users have to select one of them to answer, to deal with that we have created one UI functions for each button separately, and same as earlier we created inside these functions are constant variable for a question object named "currentQuestion", this object is carrying the answerReflection array and an answer will be added to answers array according to the pressed button and the value will be passed to the answer array.

The UI function AnswerForMultipleChoose is associated to the confirmed button of multiple choice questions. We have a constant variable currentQuestion that is identical to the current question object, and we have "if" expressions that add the answerReflection elements based on the switches on the answers array, same like previously.

By clicking this button, the result UI will be displayed, the submit button will be hidden, the countryNameVariable will be passed to the countryName label text, a COVID-19 precaution or entry requirements function will be called, and text will be passed to the covid19UpdateLabel text, and the image of the result will be displayed. The country selected by the programme based on the user's replies will be updated in the CountryMap UI component, and the top title of the navigation will be changed to "Result."