Competitor NRIC No : _____

Work Station No : _____

WORLD SKILLS SINGAPORE 2020

IT SOFTWARE SOLUTIONS FOR BUSINESS

DAY 1 SESSION 1

3 HOURS

THURSDAY 1 OCTOBER 2020

INSTRUCTIONS

1. **You are required to answer ALL the questions.**
2. **Do not open this question paper until you are told to do so.**
3. **Write your NRIC Number and Work Station Number clearly at the top right-hand corner on the front page of this question paper.**
4. **Return this question paper at the end of the test.**
5. **Please read all instructions before you begin.**

Opening Narrative

ASEAN Skills Competition (ASC) brings together competitors from 10 ASEAN member countries to compete in 22 skill areas. ASC 2020, the 13th installment of this competition, was originally scheduled to be held at the Suntec Singapore Exhibition and Convention Centre from 25 to 30 July 2020, and has now been moved to 6 to 11 April 2021.

Your consulting agency CF Consultants has been engaged to build a Flight and Greeting Scheduling Systems (FGS) that will automatically schedule flights for competitors and country officials (collectively known as “delegates”) based on a series of rules. In addition the ASC Organizing Committee has a group of volunteer “greeters” who will greet delegates at the airport and see them safely to their respective hotels.

Since many delegates extend their stay in Singapore past the end of the competition, you are required to schedule only their incoming flights from their home countries to Singapore, together with the greeters who will meet them at the airport. You are not required to schedule their flights home, and neither do you need to schedule greeters to send them off.

In the following questions, you will design and implement the FGS, which is a web-based application that will take in information about delegates, flights and greeters, and generate a flight schedule for the delegates and a greeting schedule for the greeters.

Note that the FGS is a WEB-BASED application, and thus you are required to produce the screens using HTML and CSS or equivalent, and any back-end logic, and RESTful end-points that you may require to implement the system.

FOR THE PURPOSE OF THIS COMPETITION, WE WILL BE USING THE ASC’S ORIGINAL DATES FROM 25 TO 30 JULY 2020. HENCE ALL DATA FILES AND QUESTIONS WILL CONTAIN DATES IN THIS RANGE. THIS IS NOT AN ERROR.

Preliminaries (IMPORTANT MUST READ)

In some tasks you will be asked to cut and paste results into a submission document. This is a WORD file called <name>-sess1.docx. Replace <name> with your own name, with all spaces removed.

For example if you are named "Teo Ah Kow", then your submission document should be called "teoahkow-sess1.docx". This file must be placed in your submission folder, together with all source codes, database files, etc.

Question 1

In this question you are to design the database backend for our FGS. The FGS will receive its data from CSV files through an upload interface that you will build later.

There are **TWO TASKS** in this question. There is some level of duplication between Tasks 1.1 and 1.2, and you might be tempted to do them together. However the objective of Task 1.1 is to get you to properly plan your database schema, so you are advised to do this separately from Task 1.2

Task 1.1 Designing the Database Schema and Data Dictionary

The tables below show the data columns in the three CSV files that you have been given.

The first file contains information about delegates. A delegate may be either a competitor, or an official. A competitor is a student who actually takes part in the competition, while an official may be an expert, a coach, or some other representative.

Filename: delegates.csv	
Column Name	**Description**
Passport	Delegate's passport number
Name	Delegate's name.
Nationality	Delegate's nationality.
Trade	Delegate's trade area.
Official	A single character Y if the delegate is an official, N otherwise.

The second file contains information about flights. All flights are from the respective departure city to Singapore. Competitors travel in economy class while officials travel in business class.

Filename: flights.csv	
Column Name	**Description**
City	Departure city
Country	Departure country
Flight	Flight number
Dep. Date	Departure date
Dep. Time	Departure time
Arr. Time	Arrival time in Singapore. All flights arrive on the same date that they depart.
Class	A single character E for economy class seats, B for business class seats.
Seats	Number of seats available in the respective class.
Price(SGD)	Price in Singapore Dollars (SGD)

The third file contains information about the greeters. Although the file is called "helpers.csv", every entry in this file is a greeter.

Filename: helpers.csv	
Column Name	**Description**
Helper ID	A unique four digit code identifying a helper
Helper Name	Helper's Name
Available From	Helper is available to greet delegates on or after this time.
Available To	Helper is available to greet delegates on or before this time.

Your task is to create a database schema and data dictionary to store all this information. In additional to having suitably designed tables to store the information in the three files given, you also need a table to store flight allocations to delegates (i.e. which delegate flying on which flight), as well as a table of helpers allocated to flights.

Your database schema should be properly normalized. I.e. you should make proper use of relationships and foreign keys to avoid unnecessary duplication of data.

You can use any suitable software to draw your database schema and the data dictionary.

CUT AND PASTE your answers to your submission document (<name>-sess1.docx). As you progress through this paper, you may find yourself amending and changing your database schema and/or data dictionary. If this happens, you may update your answers to this question to reflect the changes you made.

Task 1.2 Building the Database

Use your database software to create all the tables and relationships. However DO NOT store the data in the CSV files into the database yet. You will write an upload interface for this later on.

Question 2

In this question you will build ALL of the following screens. **IMPORTANT:** The screens as shown do not comply with the WSS 2020 Style Guide. Please ensure that your own screens comply with the Style Guide, or you will be penalized.

There are **FOUR TASKS** in this question.

Task 2.1 Building the Main Menu

The figure below shows the Main Menu of the FGS:

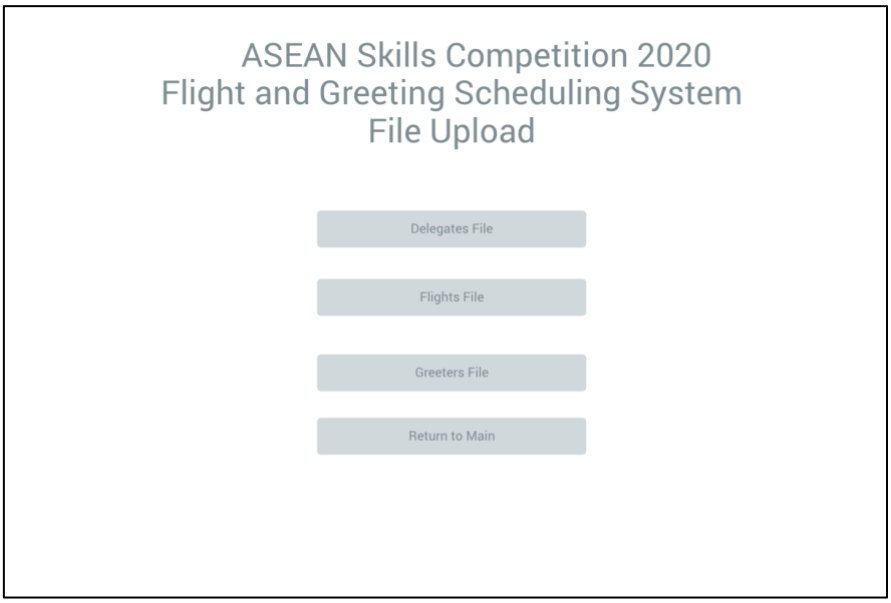
Again you must note that this screen does not comply with WSS 2020 Style Guide, and in particular the logo is missing. When you build your actual screens all these elements must be present and in the correct positions, with the correct padding. In case you are unable to see the labels on the buttons, the labels are:

Top button: Upload Data
Middle button: Schedule
Bottom button: Query

When the user clicks on "Upload Data", it will bring him to the Upload Data Screen in Task 2.2. Clicking on "Schedule" will bring him to the Schedule Flights and Greeters Screen in Task 2.3. Clicking on "Query" will bring him to the Query Screen in Task 2.4.

Task 2.2 Building the Upload Data Screen

When the user clicks on the "Upload Data" button in the Main Menu in Task 2.1, he will be brought to this screen shown below. Build this screen (and all other screens) following the WSS2020 Style Guide.



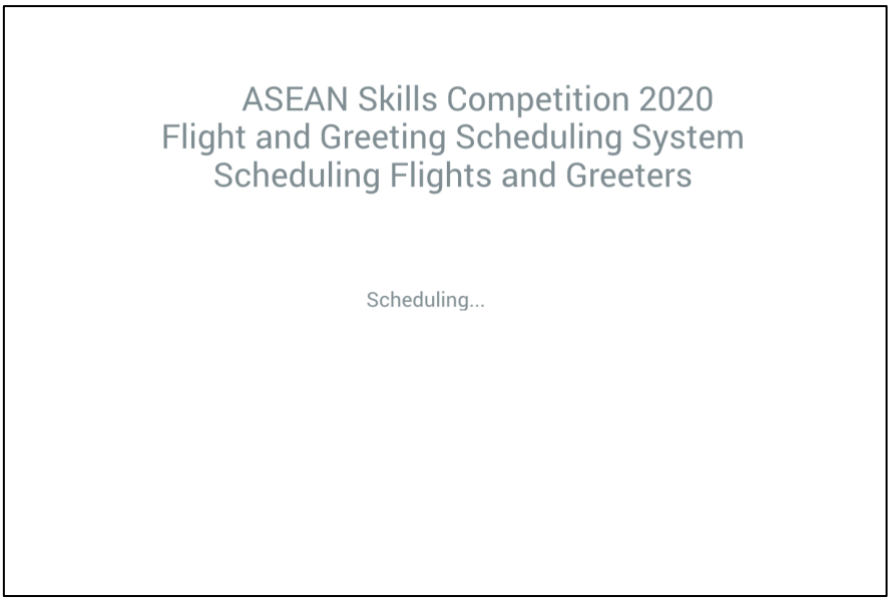
In case you cannot see the labels on the buttons, the labels are:

Top button: Delegates File
Second button: Flights File
Third button: Greeters File
Bottom button: Return to Main

The "Back to Main" button brings the user back to the Main Menu (Task 2.1). You will implement the behavior of the remaining three buttons in Question 3.

Task 2.3 Building the Schedule Screen

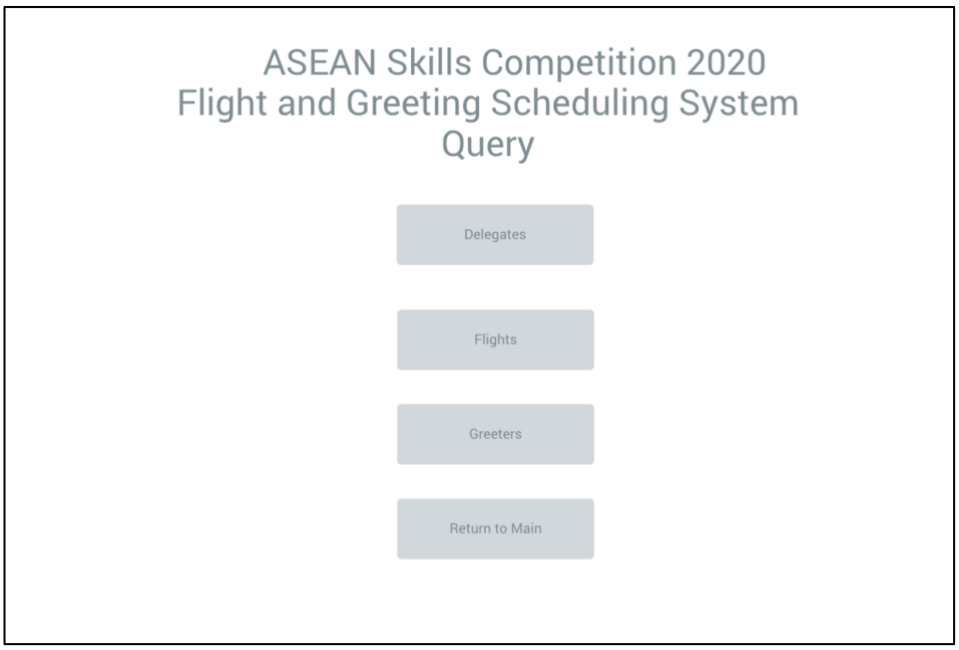
When the user clicks "Schedule" in the Main Menu, he is brought to this screen. Build this screen.



The remainder of this screen's behavior will be implemented in Question 5.

Task 2.4 Building the Query Screens

When the user clicks on "Query" in the Main Menu (Task 2.1), he will first be brought to this screen where he can query delegate, flight or greeter information:



In case you cannot see the labels on the buttons, the labels are:

a. Top button: Delegates
b. Second button: Flights
c. Third button: Greeters
d. Bottom button: Return to Main

Clicking on the "Return to Main" button will bring the user back to the Main Menu in Task 2.1. The remaining buttons are described in Tasks 2.4.1 to 2.4.3 below.

Task 2.4.1 Delegates Query

Clicking on Delegates will bring up this screen:

In case you are unable to see the labels clearly:

a. The top two text entry boxes read “Passport Number” and “Name”
b. The bottom two text entry boxes read “Nationality” and “Trade”
c. The left button at the bottom reads “Search”, and the right button “Cancel”/

This screen allows the user to search by Passport Number, Name, Nationality or Trade. The Cancel button will bring him back to the Query Screen in Task 2.4. You will implement the behaviour of the Search button in Question 4.

Task 2.4.2 Flights Query

Clicking on the Flights button in the Query screen (Task 2.4) brings up the following screen:

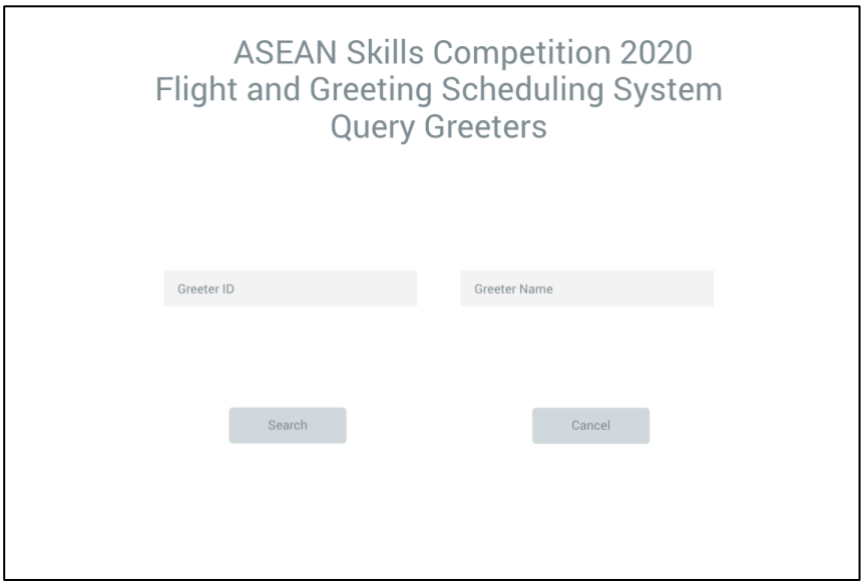
In case you are unable to see the labels clearly:

a. The two text entry boxes read “Flight” and “Departure City”.
b. The buttons at the bottom read “Search” and “Cancel”.

The user can search by flight (e.g. SQ254) or departure city (e.g. Manila). Clicking on Cancel will bring him back to the Query menu (Task 2.4). The behaviour for the Search button will be implemented in Question 4.

Task 2.4.3 Greeters Query

Clicking on the “Greeters” button in the Query screen (Task 2.4) brings up the following screen:



In case you are unable to see the labels clearly:

a. The two text boxes read “Greeter ID” and “Greeter Name”.
b. The buttons at the bottom read “Search” and “Cancel”.

Here the user can search for a Greeter by Greeter ID or by name. Clicking on Cancel will bring the user back to the Query screen in Task 2.4. The behaviour of the Search button will be implemented in Question 4.

Question 3 Uploading Data

In this question we will build the data upload capability of our FGS, allowing the users to upload information on delegates, flights and greeters. There are **THREE TASKS** in this question.

Task 3.1 Building the Front-End Behaviour

Clicking on the "Delegate File", "Flights File" or "Greeters File" button on the File Upload Screen in Task 2.2 should achieve the following:

- Give the user some way to select the file to upload, e.g. with a file-picker.
- When the user has selected which file he wants to upload, your front-end should interact with the appropriate API end-points in Task 3.2.
- Your front-end should display an appropriate results page for a successful upload or an error.

Task 3.2 Building the Back-End Behaviour

In this task you should do the following:

- When the front end sends a file, check the first row to ensure that it is the correct file for the particular upload (e.g. the delegates file should have a passport column, a name column, etc. The flights file will have a city column, a country column, a flight column, etc.). Return an appropriate error if the columns are incorrect. Note that the provided files (delegates.csv, flights.csv and helpers.csv) do not contain any errors or dirty data.
- Insert the data into the appropriate database tables.

Task 3.3 Upload the Data

Now upload the data from delegates.csv, flights.csv and helpers.csv using the interface you have just built in Tasks 3.1 to 3.2.

Question 4 Handling Queries

We will now implement the behaviour of the search buttons in Tasks 2.4.1 to 2.4.3. In every case the individual search fields are optional. Not specifying anything in any of the search fields (i.e. leaving them blank) will return every record. If the user specifies more than one search field, your system should return records that meet every field that is specified. There are **THREE TASKS** in this section.

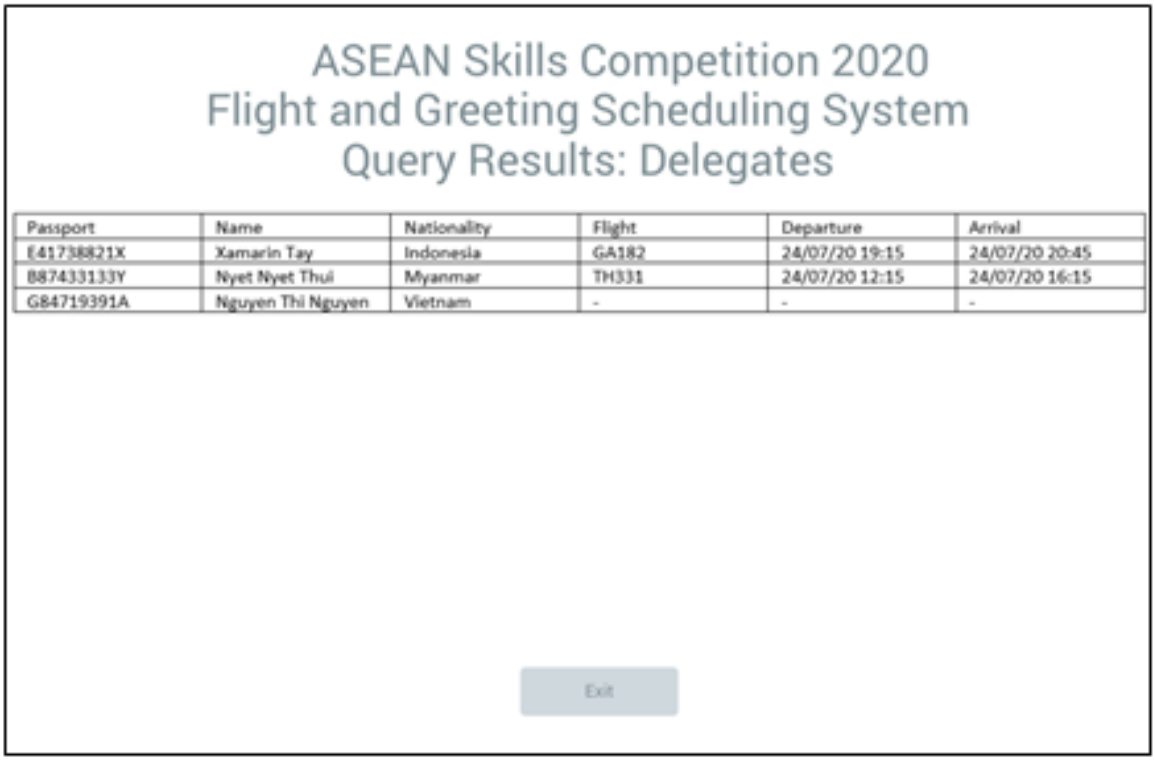
Task 4.1 Delegates Query

When the user clicks search in the Delegates Query screen (Task 2.4.1), the backend should have the following behaviour:

- If none of the search fields are filled, the back-end returns the data on all delegates.
- If one or more search fields are filled, the back-end returns the data on delegates that match each search field.
- The back-end should do a "contains" search for all filled text fields. For example if the user specifies a name of "Adam" and a Passport Number of "123", then the back-end should return all results where the name contains "Adam", and a Passport Number that contains

"123". E.g. "Name: John Adambert, Passport Number: E06123898X". All searches are not case sensitive.

Your front-end should return the following (values shown are only examples):



That is your front-end should display the delegates' passport numbers, names, nationalities, flights, departure and arrival times. If flight information is not available, display a dash ("-"). Results should be returned sorted by nationality, then name, in ascending order.

Clicking Exit will bring the user back to the Delegates Query screen (Task 2.4.1).

GENERATING RESULTS

Now use your delegates query system to generate results for the following searches:

a. Passport: N47320955P
b. Passport: B33291583W
c. Passport: Z12345678Y
d. Name: Shante

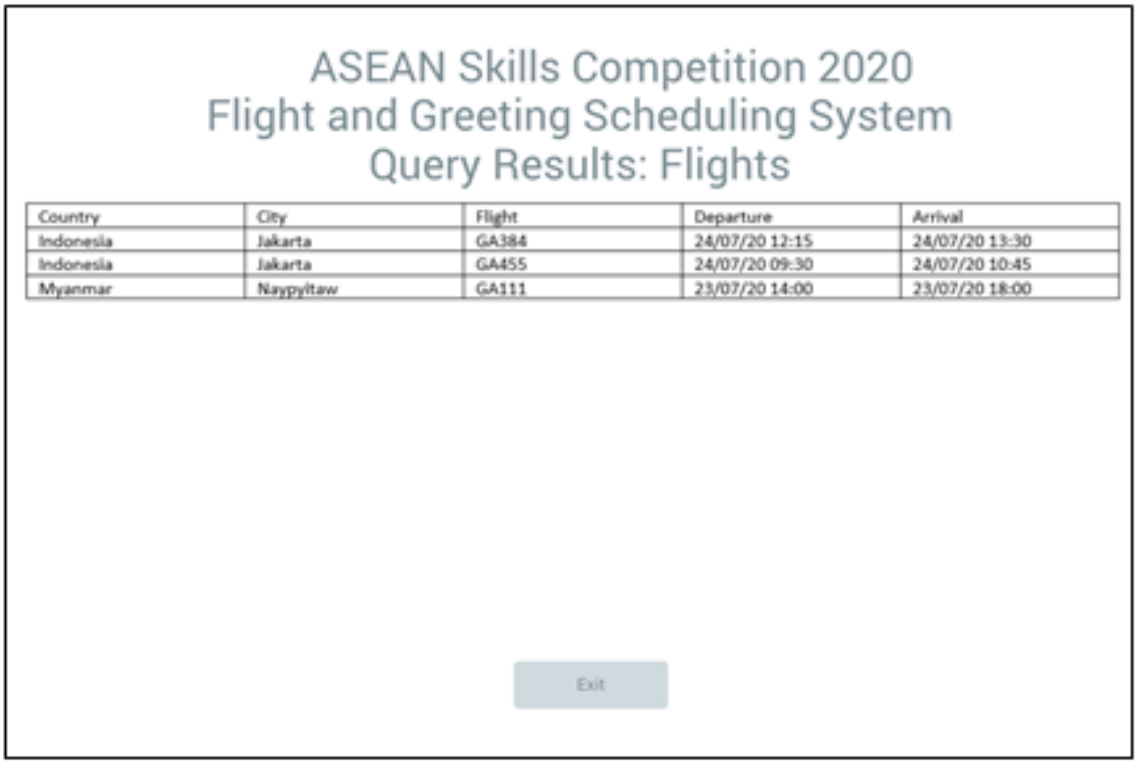
Cut and paste each result in your submission document <name>-sess1.docx.

Task 4.2 Flights Query

When the user clicks on the Search button in the Flights Query (Task 2.4.2) screen, your back end should do the following:

- If none of the search fields are filled, the back-end returns the data on all flights
- If one or more search fields are filled, the back-end returns the data on flights that match each search field.
- The backend should return all flights that contain the information in the filled query fields. For example if the user enters “363” in the Flight field, flight data on SQ363, TH363, etc. will be returned.

Your front-end should return the following (values shown are only examples):



Your system should return the departure country, departure city, flight number, departure date and time, arrival date and time, sorted by departure country and flight number, in ascending order. Clicking Exit will bring the user back to the Flights Query (Task 2.4.2) screen.

GENERATING RESULTS

Now use your flights query system to generate results for the following searches:

a. Country: USA
b. Country: Laos
c. Flight: GA800
d. Flight: 462

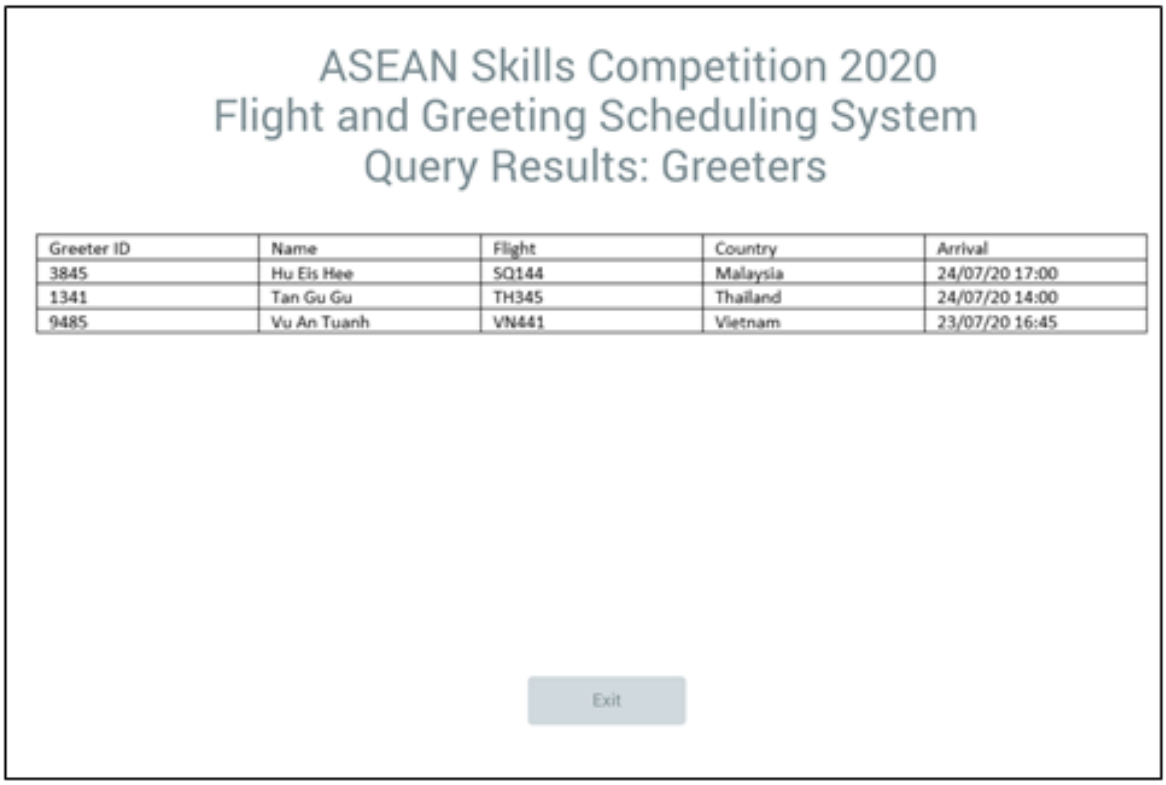
Cut and paste each result in your submission document <name>-sess1.docx.

Task 4.3 Greeters Query

When the user clicks on the Search button in the Greeters Query (Task 2.4.3) screen, your back end should do the following:

- If none of the search fields are filled, the back-end returns the data on all greeters.
- If one or more search fields are filled, the back-end returns the data on greeters that match each search field.
- The backend should return all greeters that contain the information in the filled query fields. For example if the user enters "363" in the Greeter ID field, then greeters with IDs of 7363, 3632, 4363, etc. are returned.

Your front-end should return the following (values shown are only examples):



That is your system should return the Greeter ID, Greeter Name, Flight that the Greeter is allocated to, the Flight's departure country, and arrival date and time. The list should be sorted by Greeter Name in ascending order. Clicking Exit will return the user to the Greeters Query (Task 2.4.3) screen.

GENERATING RESULTS

Now use your greeters query system to generate results for the following searches:

a. Greeter ID: 8961
b. Greeter ID: 0000
c. Name: Vilma
d. Greeter ID: 2676

Cut and paste each result in your submission document <name>-sess1.docx.

Question 5 Scheduling

We will now schedule the delegate's flights and the greeters. This function is triggered when the user clicks on the Schedule button in the Main Menu in Task 2.1. The front end will display the Schedule Screen in Task 2.3 as scheduling is in progress. This question has **TWO TASKS**:

Task 5.1 Building the Scheduler

The scheduler will schedule using the following rules:

1. For each country IN THIS ORDER: Indonesia, Singapore, Thailand, Malaysia, Philippines, Vietnam, Laos, Cambodia, Brunei, Myanmar -
    a. Schedule the "non-officials" delegates first, sorted by trade and then by name. Then allocate for "officials" delegates, again sorted by trade and then by name. Non-officials will be allocated Economy Class seats, while officials will be allocated Business Class seats.
    b. Schedule delegates to flights based on the following rules:
        i. Must arrive in Singapore before midnight of 25 July 2020 (i.e. by 23:59 of 24 July 2020)
        ii. Must originate in the country being scheduled.

iii. Flights should be sorted and allocated to delegates by ARRIVAL date and time in descending order (e.g. 24/7/2020 11.00 pm, then 24/7/2020 9.45 pm, then 23/7/2020 10.45 pm etc.), followed by cost of ECONOMY class seats for non-officials, and cost of BUSINESS class seats for officials in ascending order. Table below shows an example sorting for Indonesia:

Flight	Arrival Time	Economy Class Price	# of seats
SQ145	24/07/20 23:00	S$315	7
TH451	24/07/20 23:00	S$345	6
MH114	24/07/20 22:30	S$250	9
SQ175	24/07/20 22:00	S$245	8

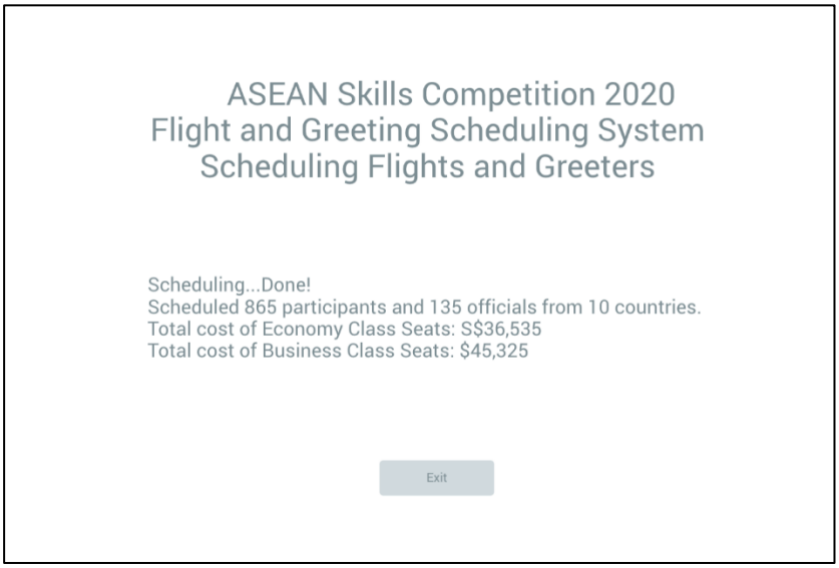
Flights will be allocated to delegates in the order shown in the table. That is, the first 7 delegates will be allocated to SQ145, the next 6 to TH451, the next 9 to MH114, etc. Again the delegates are sorted alphabetically by trade area then by name, in ascending order.

iv. Obviously you should not allocate more people to a flight than the number of allocated seats in a particular class.

c. Schedule Greeters to each flight according to the following rules:
   i. The flight's arrival time should be between the greeter's start and end times inclusive.
   ii. Greeters are ordered and assigned to flights by Greeter ID.

2. You must maintain find the total cost of economy class seats, and the total cost of business class seats, for EACH country.

3. When your system finishes the scheduling, your front-end should show:



That it should show how many participants and officials were scheduled across how many countries, and the total cost of Economy and Business class seats across all countries (numbers shown are only examples).

Task 5.2 Generating Results

1. Click on Schedule in the Main Menu to generate the flight and greeter allocation.

2. Now use your delegates query system in Task 4.1 to generate results for the following delegates:
    a. N47320955P
    b. B33291583W
    c. Z12345678Y
    d. Shante

    Cut and paste each result in your submission document <name>-sess1.docx.

3. Use your greeters query system in Task 4.3 to generate results for the following greeters:

    a. 8961
    b. 0000
    c. Vilma
    d. 2676

    Again cut and paste each result in your submission document <name>-sess1.docx.

SUBMISSION

Remember to ensure that you submit your submission document (e.g. tanahkow-sess1.docx), all source codes needed to compile and run your system, and all database files. Incomplete submissions may not be graded.

-END OF PAPER