**BlueJack Station**

**BlueJack Station** is the biggest train station in Jakarta. Many people come here to board on trains to various destinations. Because the train station grows bigger each year, the station manager needs a program to keep track of all the train routes from **BlueJack Station** to other cities. Unfortunately, he can’t code. You as a good programmer, have been asked to **create a program** that can **list all the train routes** starting from **BlueJack Station**, based on the following criteria using **Java Programming Language:**

At the start of the application, it will show a **menu**. The menu consists of the following specifications :

1. **Insert Route**
2. **View Route**
3. **Delete Route**
4. **Exit**

Background pattern

Description automatically generated with low confidence

**Figure 1. home menu**

* If the user chooses **menu 1 (Insert Route):**
  + Ask the user to input:
    - **Destination**, which must be either **‘Jakarta’, ‘Bogor’, ‘Depok’, ‘Tangerang’, or ‘Bekasi’ (case sensitive)**
    - **Distance,** which must be a **number** between **1 and 1000**
  + After the user **successfully** inputted all the data, **generate** an **ID** for the route based on the following **formula** :

|  |
| --- |
| **ID = ‘BJXXX’ | X = random between 0-9**  **Example: BJ345** |

* + **Add** the data to **Array / Vector / ArrayList.**
  + Finally, **show success message**: **“Successfully added new route!”**

Chart

Description automatically generated with medium confidence

**Figure 2. insert route**

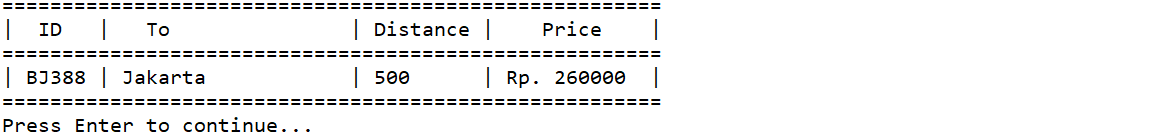
* If the user chooses **menu 2 (View Routes):**
  + If **there is no route data**, show the message **“There are no routes available.”,** then the program will return to **main menu.**



**Figure 3. no data message**

* + Otherwise, **show** the **list of all routes,** along with their **price,** which is calculated based on the following formula:

|  |
| --- |
| **Price = ((distance / 2 \* 1000) + 10000)** |



**Figure 4. view routes**

* If the user chooses **menu 3 (Update Route):**
  + If **there is no route data**, show the message **“There are no routes available.”,** then the program will return to the **main menu.**



**Figure 5. no data message**

* + Otherwise:
    - **Show** the **list of all routes,** along with their **price.**
    - Ask the user to input **Route ID**, which must be **a valid ID**.
    - If the user inputs an **invalid ID,** the program will **display error message** and return to the **main menu.**

Diagram

Description automatically generated with medium confidence

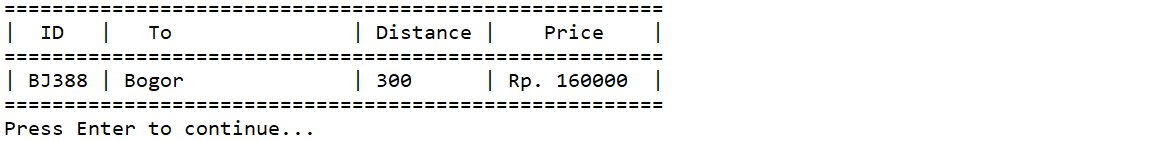
**Figure 6. invalid ID error message**

* + - Otherwise, the program will **ask** the user to **input the route details:**
      * **Destination**, which must be either **‘Jakarta’, ‘Bogor’, ‘Depok’, ‘Tangerang’, or ‘Bekasi’ (case sensitive)**
      * **Distance,** which must be a **number** between **1 and 1000**
    - After the user **successfully** inputted all the data, **update the route data.**
    - Finally, **show success message: “Successfully updated Route ” + the Route ID which was just updated.**

A picture containing diagram

Description automatically generated

**Figure 7. update route**



**Figure 8. view route (after update)**

* If the user chooses **menu 4 (Exit),** then the program will be closed.