CPRG211 Final Project
Group 3

Exceptions

Most exception have been caught through careful coding. However, there are still possible null values that can occur and cause the program to crash. Here are some of the means used to catch and resolve the errors.

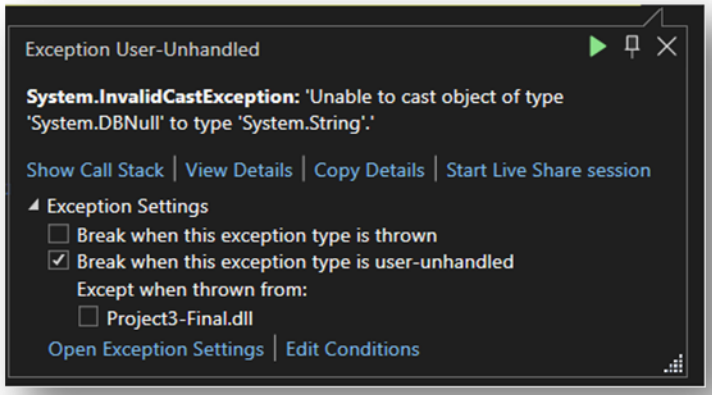
An error occurs when the datatype in the MySQL database cannot be cast to the object is assigned.

For example, if the MySQL data contains a null value and the application tries to cast the empty data cell as a variable for an object, this can result in an error.

gymID	street	city	prov	postalCode	gymStatus
1	123 Mainstreet NW	Calgary	Alberta	T3Z1B2	1
2	1301 16 Ave NW	Calgary	Alberta	T2M0L4	1
3	5150 North	Regina	Saskatchewan	S0K0Y	0
4	101 55st	Dinosaur City	Newfoundland and Labrador	B1B1X1	1
5	53 University Drive NW	NULL	Alberta	T3Z1B2	1
6	123 Road	Maple	Alberta	t3d4bs	1

1. Example of null value in database

When the null value is cast to a string it results as in an **InvalidCastException**



2.Example of error resulting from null value in database.

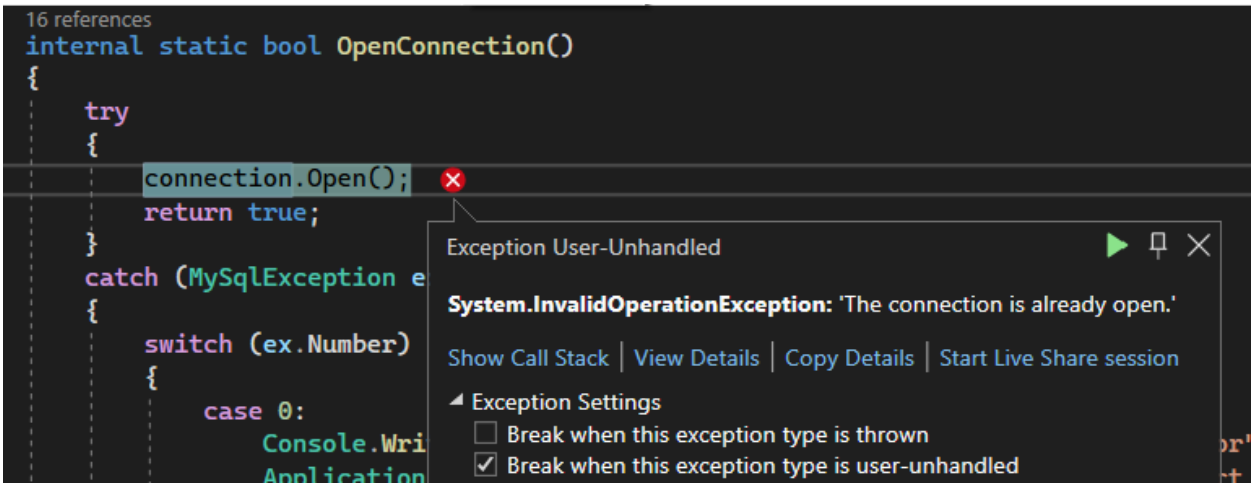
The solution to this is to have the if/else statements used to read from the database wrapped in a try-catch-finally block.

```
//Try Catch block to incase of invalid cast exceptions
//ie table field is null
try
{


    if (OpenConnection() == true)...
    else...
}
catch (InvalidCastException ex)
{
    Debug.WriteLine("Error casting value from database. Check database data type and if null\n" + ex.Message);
}
catch (Exception)
{
    Debug.WriteLine("Something even worse happened loading from database!");
}
finally //Close connection to prevent InvalidOperationException
{
    CloseConnection();
```

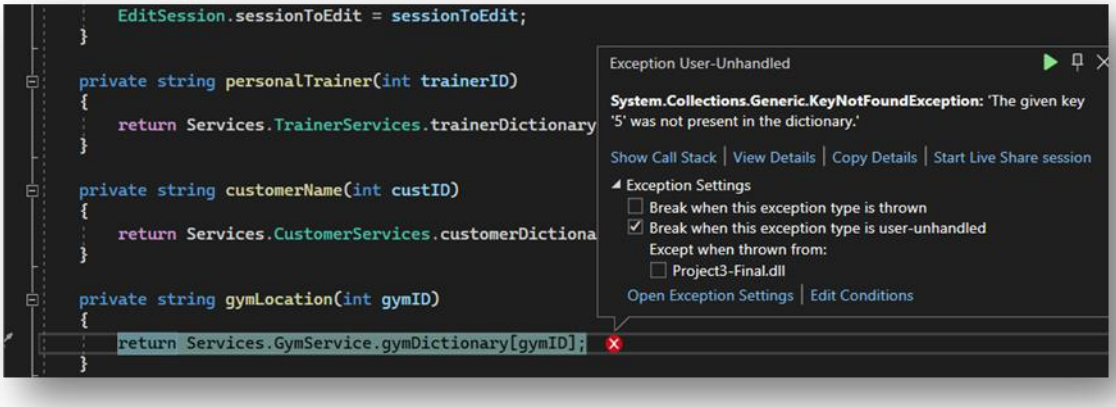
3. Try-Catch-Finally block used to catch exemptions

Try will contain the code to read the records from the database. The first catch is an **InvalidCastException**, while the second is a more general **Exeception**. Lastly, there is a finally statement saying to close the connection to the database. This is to prevent ab **InvalidOperationExeption** should another method try to open a connection.



4. Example of InvalidOperationExmption caused by application trying to open connection without closing existing connection.

Another error which can occur is a **KeyNotFoundExecption** when the various dictionaries try to return a value which does not exist in the dictionary.

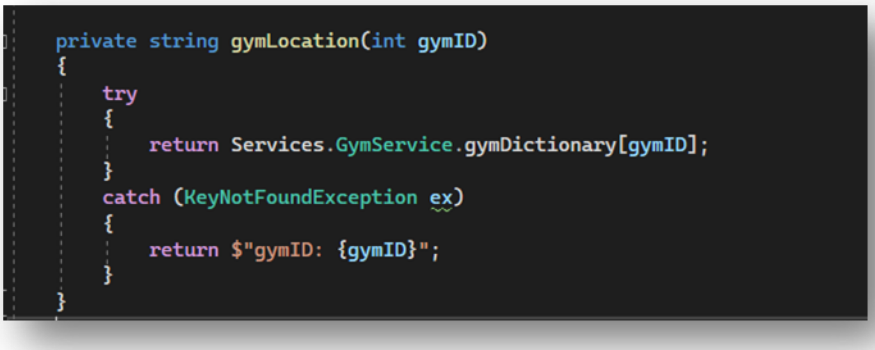


5.KeyNotFoundExeception as a result of missing dictionary value for key 'K'

The above error occurred because there is an session object requesting a dictionary value for a GymID which does not exist in the database. Normally whenever records are pulled from the MySQL database dictionary keys and values are automatically created for all gym records. However, in this example the *null* value in the city field of the resulted in only a partial loading of the records for Gym objects and an incomplete dictionary.

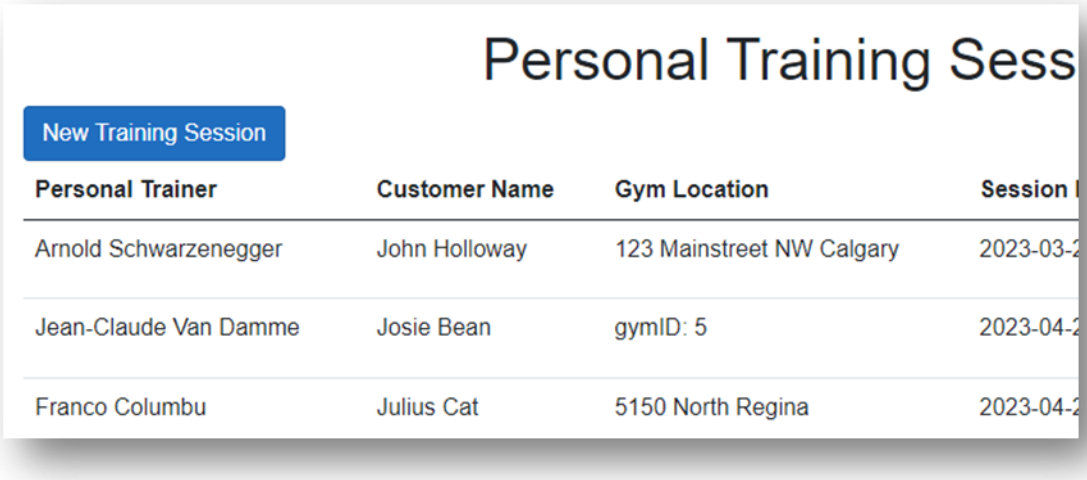
This error can also occur if one encounters internet connection issues while using the application. If there is a connection error and a specific table is not loaded from the database, but internet connectivity is re-established afterwards, an incomplete dictionary can exist resulting in an occurrence of this error.

The solution to this exception is a try-catch block when invoking the dictionary objects



6. Try-catch block for dictionary

The try catch block will catch the error and return an alternative string value for the method.



7. Alternate string returned to application displayed in GUI