

```

1  from cassandra.auth import PlainTextAuthProvider
2  from cassandra.cluster import Cluster
3
4  from config import database_config
5  from Logger import Logging
6
7  logger_obj = Logging('Advance Image Downloader') # Creating a custom based logger
8  logger_obj.initialize_logger() # Instantiating the logger object
9
10
11 class Cassandra:
12
13     def __init__(self):
14         """
15         This function will instantiate the session for the Cassandra database
16         """
17         try:
18             cloud_config = {
19                 'secure_connect_bundle': database_config.cloud_config_path
20             }
21             auth_provider = PlainTextAuthProvider(database_config.cassandra_username,
22                                                  database_config.cassandra_password)
23             cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
24             self.session = cluster.connect()
25
26         except Exception as e:
27             logger_obj.print_log('(Cassandra.py(__init__) - Something went wrong ' + str(e), '
exception')
28             raise Exception(e)
29
30     def connect_keyspace(self):
31         """
32         This function will use the given keyspace as the default method to work on.
33         """
34         try:
35             self.session.set_keyspace(database_config.keyspace_name)
36         except Exception as e:
37             logger_obj.print_log('(Cassandra.py(connect_keyspace) - Something went wrong ' +
str(e), 'exception')
38             raise Exception(e)
39
40     def create_table(self):
41         """
42         This function will create the table if it does not exists in the keyspace
43         """
44         try:
45             self.session.execute('CREATE TABLE IF NOT EXISTS { } '
'(id UUID, email text, url text, PRIMARY KEY (id, email, url));'
.format(database_config.table_name))
46         except Exception as e:
47             logger_obj.print_log('(Cassandra.py(create_table) - Something went wrong ' + str(e), '
exception')

```

```

50     raise Exception(e)
51
52     def select_query(self, req_id):
53         """
54         This function will execute and return the select query on the table
55         :param req_id: Unique request id for the request generated by the user
56         :return: Results after the selection query
57         """
58         try:
59             return self.session.execute('SELECT id, email, url FROM {} WHERE id={}'.format(
60                 database_config.table_name,
61                 req_id))
62         except Exception as e:
63             logger_obj.print_log('(Cassandra.py(select_query) - Something went wrong ' + str(e)
64             ), 'exception')
65             raise Exception(e)
66
67     def insert_url(self, uuid, email, url):
68         """
69         This function will insert the data into the table
70         :param uuid: It is a unique user id object
71         :param email: email of the user
72         :param url: url of the search query
73         """
74         try:
75             self.session.execute(
76                 "INSERT INTO " + database_config.table_name + " (id, email, url) VALUES (%s, %s
77             , %s)",
78                 (uuid, email, url))
79         except Exception as e:
80             logger_obj.print_log('(Cassandra.py(insert_url) - Something went wrong ' + str(e), '
81             exception')
82             raise Exception(e)
83
84     def shutdown(self):
85         """
86         This function will close the cassandra session
87         """
88         try:
89             self.session.shutdown()
90         except Exception as e:
91             logger_obj.print_log('(Cassandra.py(shutdown) - Something went wrong ' + str(e), '
92             exception')
93             raise Exception(e)
94
95     def delete_url(self, req_id):
96         """
97         This function will delete the url for given request ID
98         """
99         try:
100             self.session.execute('DELETE FROM {} WHERE id={}'.format(database_config.

```

```
96 table_name, req_id))
97     except Exception as e:
98         logger_obj.print_log('(Cassandra.py(shutdown) - Something went wrong ' + str(e), '
exception')
99         raise Exception(e)
100
101 def drop_table(self):
102     """
103     This function will drop the given table from the keyspace
104     """
105     try:
106         self.session.execute('DROP TABLE IF EXISTS {}'.format(database_config.table_name))
107     except Exception as e:
108         logger_obj.print_log('(Cassandra.py(drop_table) - Something went wrong ' + str(e), '
exception')
109         raise Exception(e)
110
```