Database
Management System

Report

Mina Ashraf
Hossam ElDin AbdelGhany
Pierre Maged
Ahmed Yasser

Problem Statement

It was requested to make a database management system that stores the data locally on the computer in the form of xml files and can do the following commands

create database	Drop database
create table	delete table
insert into table	update table
delete from table	select from table

with the ability to add condition to some of the previous commands

UML Diagram

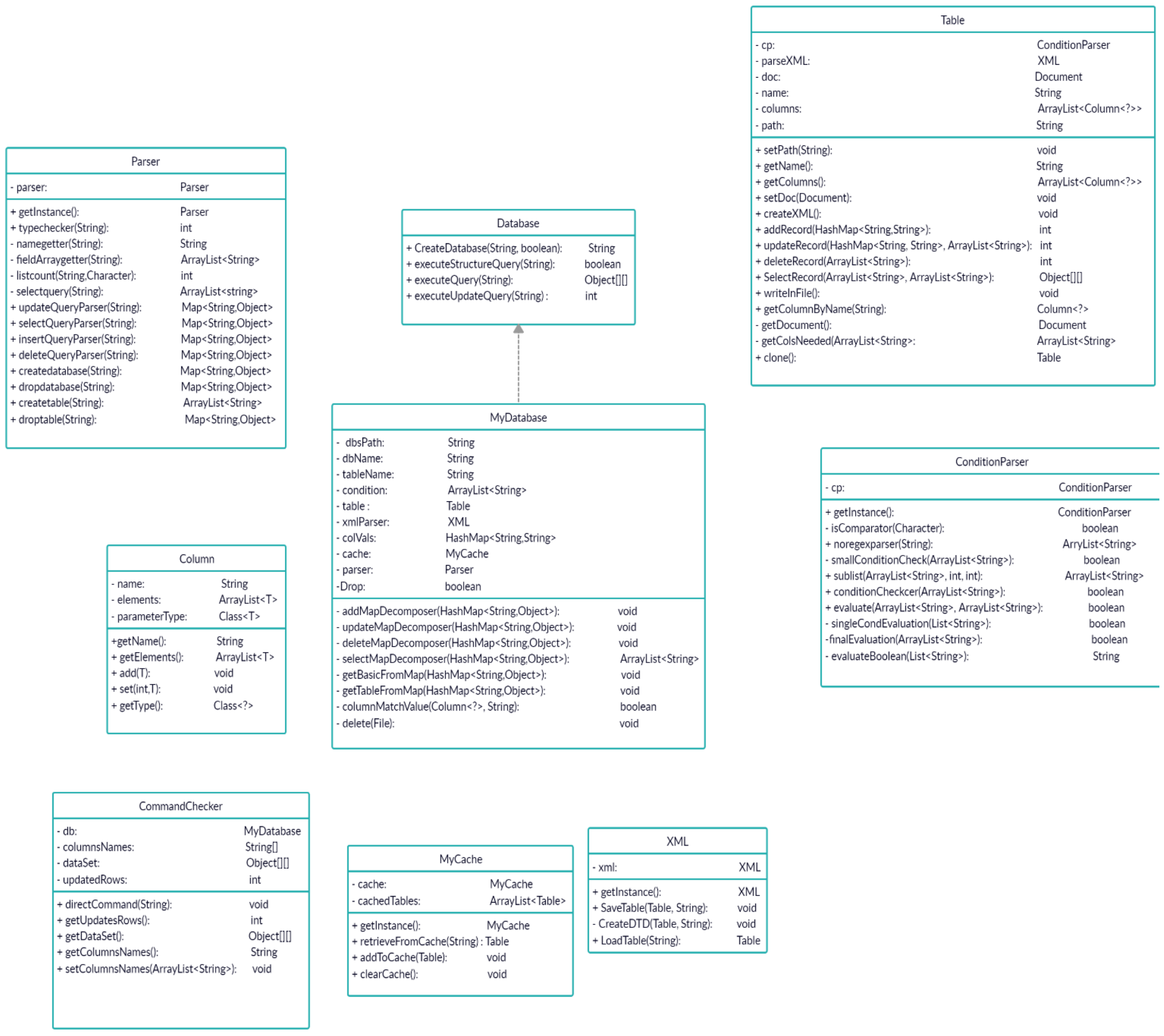
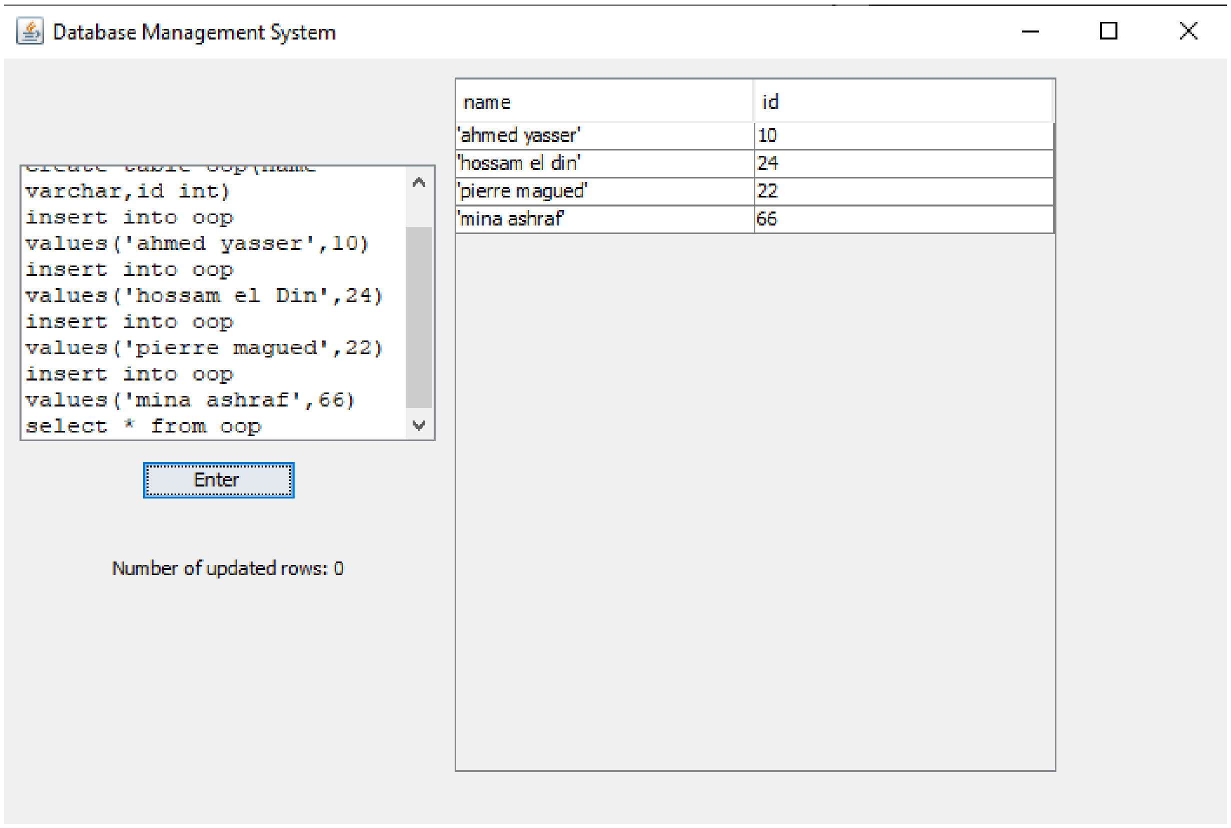


Table
- cp: ConditionParser
- parseXML: XML
- doc: Document
- name: String
- columns: ArrayList<Column<?>>
- path: String
+ setPath(String): void
+ getName(): String
+ getColumns(): ArrayList<Column<?>>
+ setDoc(Document): void
+ createXML(): void
+ addRecord(HashMap<String,String>): int
+ updateRecord(HashMap<String, String>, ArrayList<String>): int
+ deleteRecord(ArrayList<String>): int
+ SelectRecord(ArrayList<String>, ArrayList<String>): Object[][]
+ writeInFile(): void
+ getColumnByName(String): Column<?>
- getDocument(): Document
- getColsNeeded(ArrayList<String>: ArrayList<String>
+ clone(): Table
Parser
- parser: Parser
+ getInstance(): Parser
+ typechecker(String): int
- namegetter(String): String
- fieldArraygetter(String): ArrayList<String>
- listcount(String,Character): int
- selectquery(String): ArrayList<string>
+ updateQueryParser(String): Map<String,Object>
+ selectQueryParser(String): Map<String,Object>
+ insertQueryParser(String): Map<String,Object>
+ deleteQueryParser(String): Map<String,Object>
+ createdatabase(String): Map<String,Object>
+ dropdatabase(String): Map<String,Object>
+ createtable(String): ArrayList<String>
+ droptable(String): Map<String,Object>
Database
+ CreateDatabase(String, boolean): String
+ executeStructureQuery(String): boolean
+ executeQuery(String): Object[][]
+ executeUpdateQuery(String) : int
MyDatabase
- dbsPath: String
- dbName: String
- tableName: String
- condition: ArrayList<String>
- table : Table
- xmlParser: XML
- colVals: HashMap<String,String>
- cache: MyCache
- parser: Parser
-Drop: boolean
- addMapDecomposer(HashMap<String,Object>): void
- updateMapDecomposer(HashMap<String,Object>): void
- deleteMapDecomposer(HashMap<String,Object>): void
- selectMapDecomposer(HashMap<String,Object>): ArrayList<String>
- getBasicFromMap(HashMap<String,Object>): void
- getTableFromMap(HashMap<String,Object>): void
- columnMatchValue(Column<?>, String): boolean
- delete(File): void
ConditionParser
- cp: ConditionParser
+ getInstance(): ConditionParser
- isComparator(Character): boolean
+ noregexparser(String): AryList<String>
- smallConditionCheck(ArrayList<String>): boolean
+ sublist(ArrayList<String>, int, int): ArrayList<String>
+ conditionCheckcer(ArrayList<String>): boolean
+ evaluate(ArrayList<String>, ArrayList<String>): boolean
- singleCondEvaluation(List<String>): boolean
-finalEvaluation(ArrayList<String>): boolean
- evaluateBoolean(List<String>): String
Column
- name: String
- elements: ArrayList<T>
- parameterType: Class<T>
+getName(): String
+ getElements(): ArrayList<T>
+ add(T): void
+ set(int,T): void
+ getType(): Class<?>
CommandChecker
- db: MyDatabase
- columnsNames: String[]
- dataSet: Object[][]
- updatedRows: int
+ directCommand(String): void
+ getUpdatesRows(): int
+ getDataSet(): Object[][]
+ getColumnsNames(): String
+ setColumnsNames(ArrayList<String>): void
MyCache
- cache: MyCache
- cachedTables: ArrayList<Table>
+ getInstance(): MyCache
+ retrieveFromCache(String) : Table
+ addToCache(Table): void
+ clearCache(): void
XML
- xml: XML
+ getInstance(): XML
+ SaveTable(Table, String): void
- CreateDTD(Table, String): void
+ LoadTable(String): Table

Better quality available upon request

Screenshots of the app

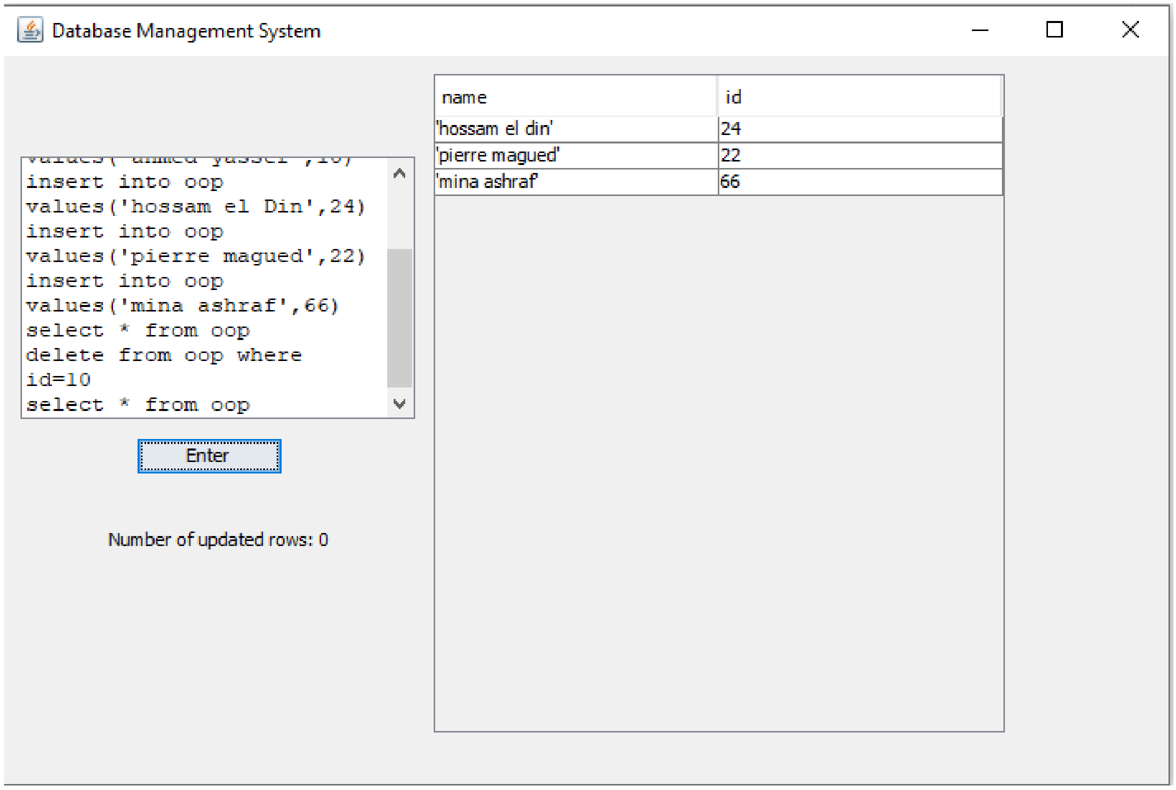


Database Management System

```
varchar,id int)
insert into oop
values('ahmed yasser',10)
insert into oop
values('hossam el Din',24)
insert into oop
values('pierre magued',22)
insert into oop
values('mina ashraf',66)
select * from oop
```

Number of updated rows: 0

name	id
'ahmed yasser'	10
'hossam el din'	24
'pierre magued'	22
'mina ashraf'	66

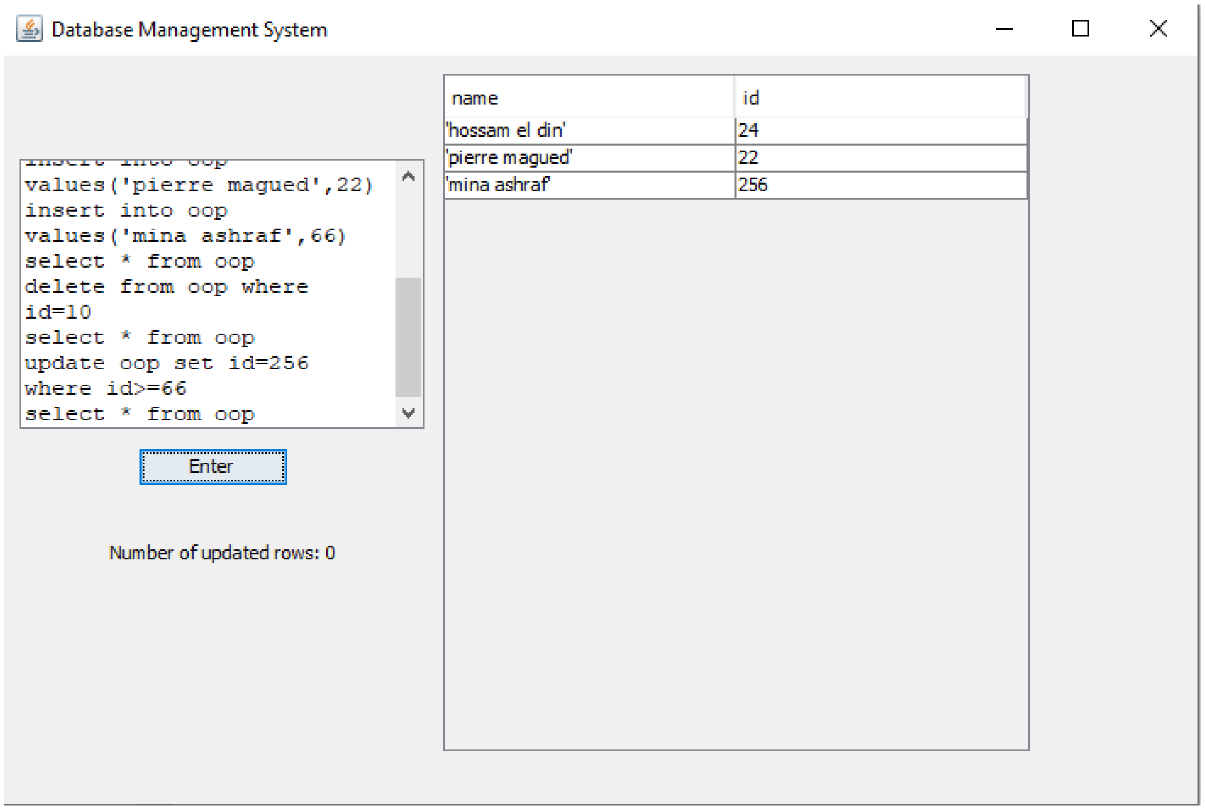


Database Management System

```
insert into oop
values('hossam el Din',24)
insert into oop
values('pierre magued',22)
insert into oop
values('mina ashraf',66)
select * from oop
delete from oop where
id=10
select * from oop
```

Number of updated rows: 0

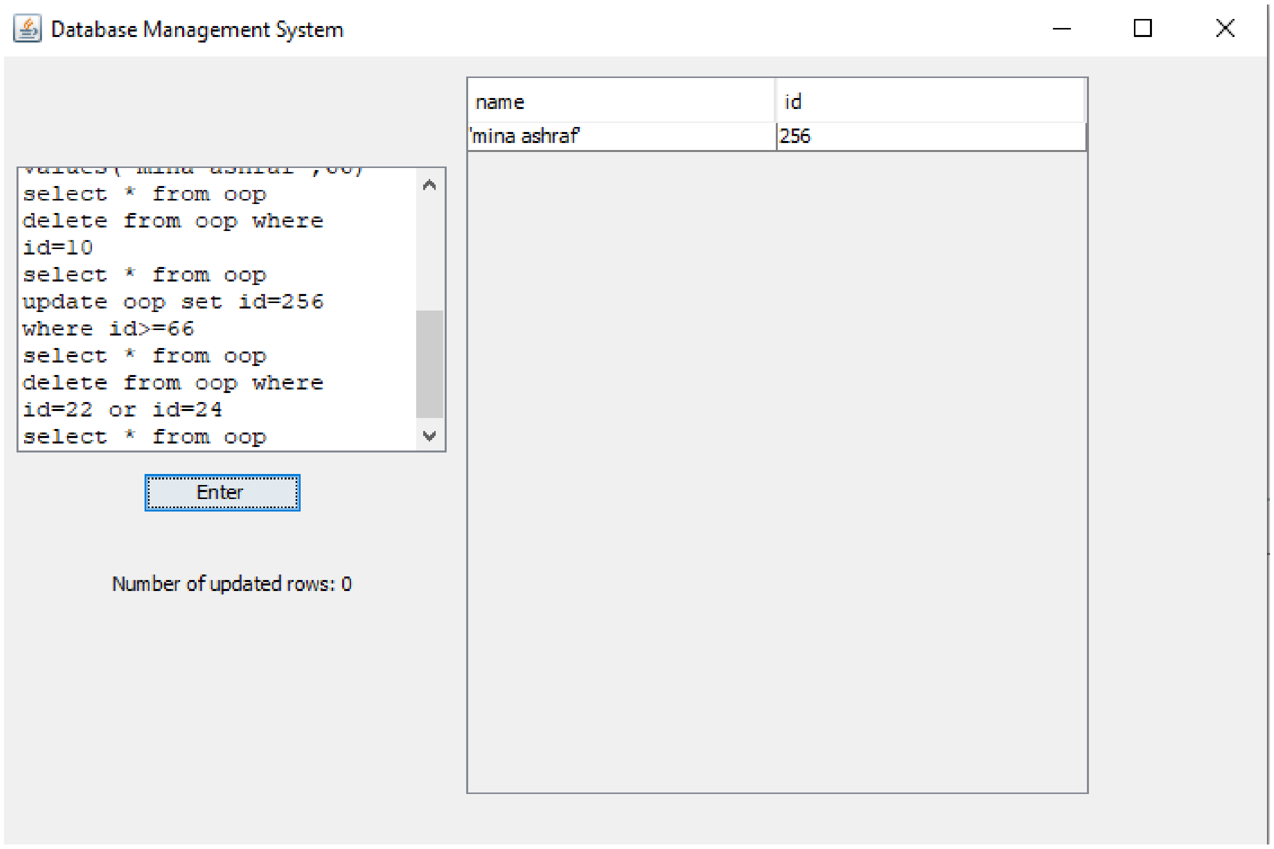
name	id
'hossam el din'	24
'pierre magued'	22
'mina ashraf'	66


Database Management System

```
values('pierre magued',22)
insert into oop
values('mina ashraf',66)
select * from oop
delete from oop where
id=10
select * from oop
update oop set id=256
where id>=66
select * from oop
```

Number of updated rows: 0

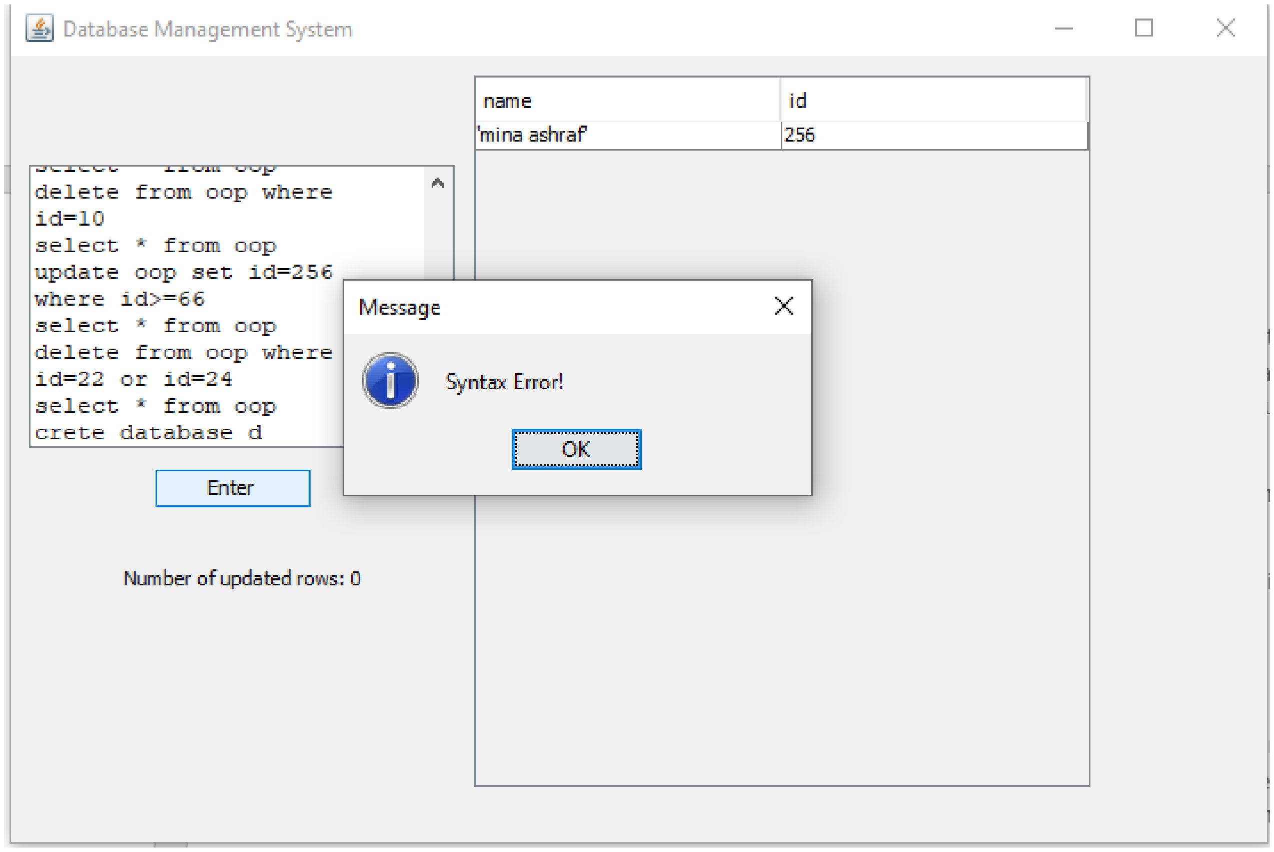
name	id
'hossam el din'	24
'pierre magued'	22
'mina ashraf'	256


Database Management System

```
select * from oop
delete from oop where
id=10
select * from oop
update oop set id=256
where id>=66
select * from oop
delete from oop where
id=22 or id=24
select * from oop
```

Number of updated rows: 0

name	id
'mina ashraf'	256



```
select * from oop
delete from oop where
id=10
select * from oop
update oop set id=256
where id>=66
select * from oop
delete from oop where
id=22 or id=24
select * from oop
crete database d
```

Number of updated rows: 0

name	id
'mina ashraf'	256

Message

Syntax Error!

Design patterns used

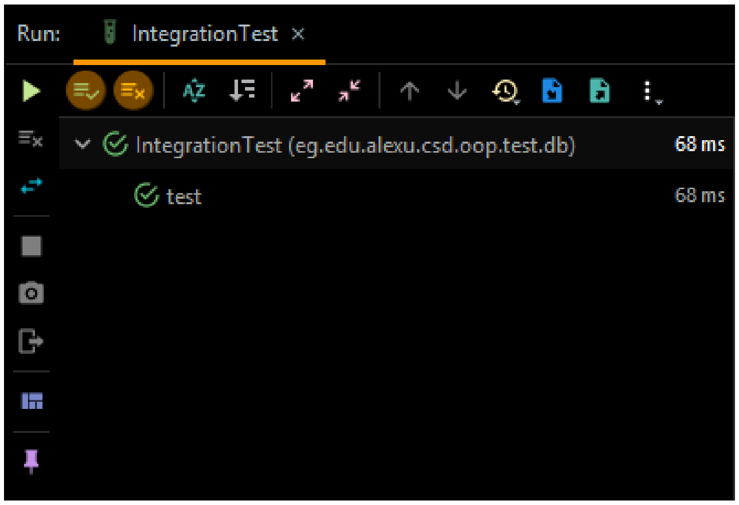
Singleton

Façade

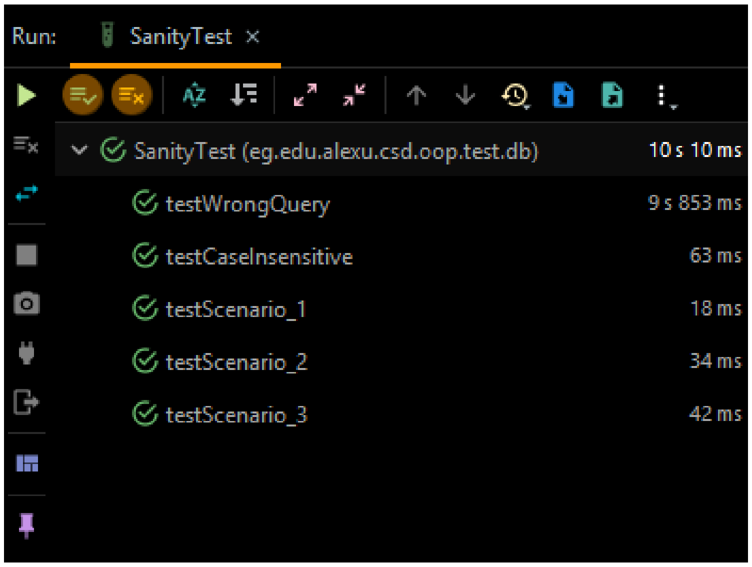
MVC

Prototype

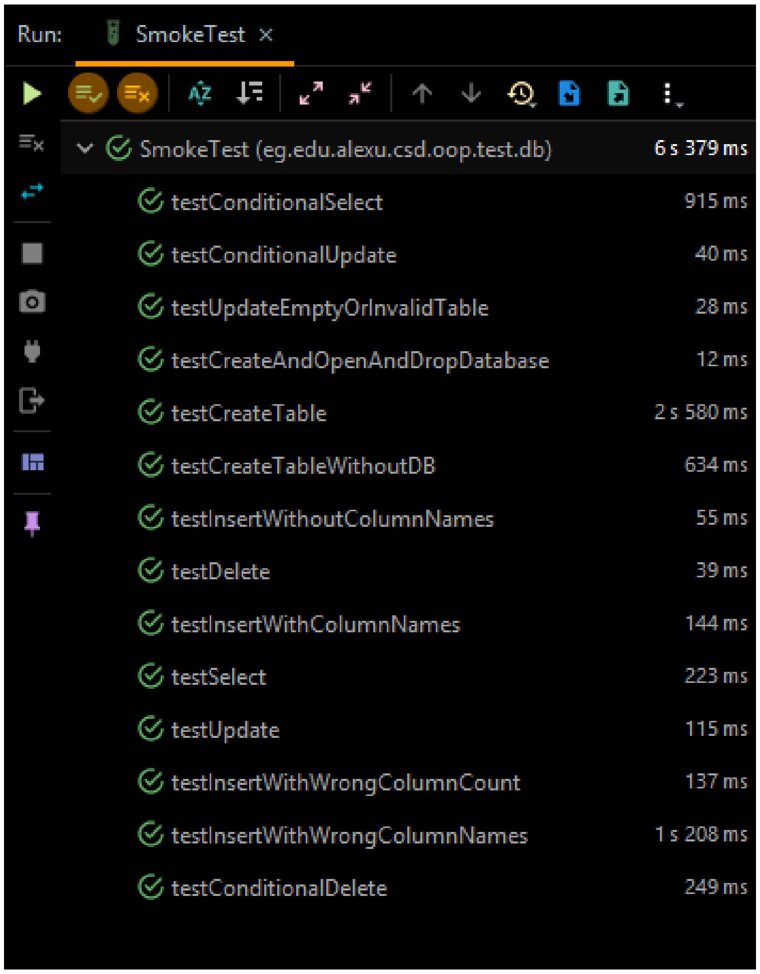
Passing the tests



```
Run:   IntegrationTest ×
IntegrationTest (eg.edu.alexu.csd.oop.test.db)   68 ms
    test                                          68 ms
```



```
Run:   SanityTest ×
SanityTest (eg.edu.alexu.csd.oop.test.db)   10 s 10 ms
    testWrongQuery                           9 s 853 ms
    testCaseInsensitive                      63 ms
    testScenario_1                           18 ms
    testScenario_2                           34 ms
    testScenario_3                           42 ms
```



```
Run:   SmokeTest ×

SmokeTest (eg.edu.alexu.csd.oop.test.db)      6 s 379 ms
    testConditionalSelect                       915 ms
    testConditionalUpdate                        40 ms
    testUpdateEmptyOrInvalidTable                28 ms
    testCreateAndOpenAndDropDatabase             12 ms
    testCreateTable                          2 s 580 ms
    testCreateTableWithoutDB                    634 ms
    testInsertWithoutColumnNames                 55 ms
    testDelete                                   39 ms
    testInsertWithColumnNames                   144 ms
    testSelect                                  223 ms
    testUpdate                                  115 ms
    testInsertWithWrongColumnCount              137 ms
    testInsertWithWrongColumnNames           1 s 208 ms
    testConditionalDelete                       249 ms
```

How it works

When the user inputs a query the command director checks the type of this query and passes it to one of the My database class's (the class that implements the database interface)according to its type, the my database method checks the subtype and then sends the query to the parser class which parses the query and returns a map containing the data needed to execute this query if this query contains a condition the condition is passed to the condition parser class which parses the condition and it is then placed into the map
if the query contains a syntax error the parses returns a null map and the My database class throws an exception and shows a dialogue box when creating a database a folder is created and when creating a table an xml and a dom file are created within this folder

The data is cached in the application (the cache class) and is saved after a number of creations and edits because the file writing is a very costly operation also the cache class writes its output to the file when the app is closed to ensure no data loss

How to use it

Type your SQL style commands into the text box and press the enter button bellow the text box, if you used the select command the output table will be shown in the table in the right part of the application if your command is invalid a pop up dialogue will appear indicating that you committed a syntax error the xml files are saved locally and can be copied and pasted to any other computer in the dbms folder and can be viewed then

Supported commands

Create database <database name >

Create table <table name> (<field 1><field 1 type int or varchar>,<field 2><field 2 type int or varchar>,....)

Drop database <database name>

Drop table <table name >

Select from <table name > <*/ fields you want to show separated by commas> where <condition>
condition is optional

Insert into <table name > (<field1>,<field 2>,...) values (<value1>,<value2>,...) or insert into <table name > values (<value1>,<value 2>,...)

Update <table name> set <field1>=<value1>, <field2>=<value2>,.... Where <condition>
if no condition is added all the able entries will be updated

Delete from <table name > where<condition>
if no condition is present al the table will be deleted