Truc Huynh

Purdue University

Client Sever Java Implementation

I. Overview:

- Project Admin: Truc Huynh
- Project using Java and build **without any framework**.
- Using Apache Tomcat as the servlet.
- Client and Server are built separately (Please refer to the UML)

II. Test Plan:

a. Run Server and test if Server is function properly

b. Simultaneously create three clients and test if they are function properly

c. Upload the data to Server (#1)

d. Send Request to Server (#2)

e. Received Request from Server (#2) (#3)

III. Implementation and Result

a. Run Server and test if Server is function properly:

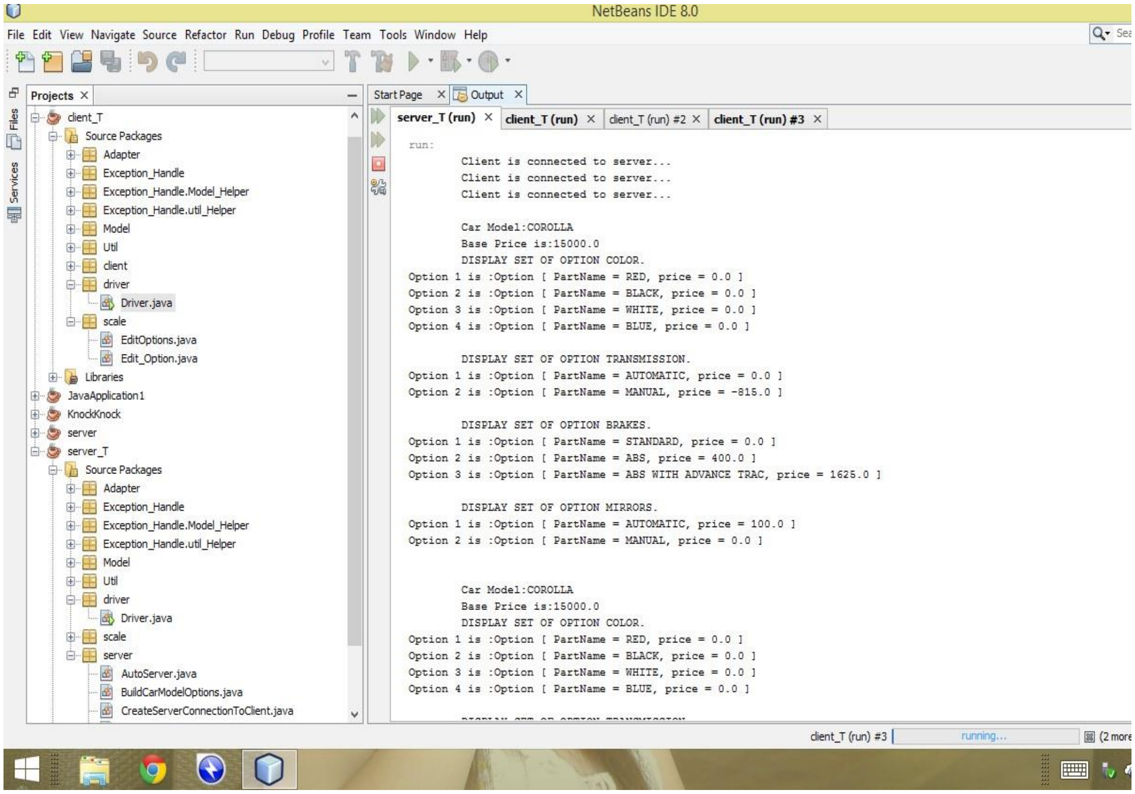
Tested complete successfully.

b. Simultaneously create three clients and test if they are function properly:

Tested complete successfully.

c. Upload the data to Server (#1):

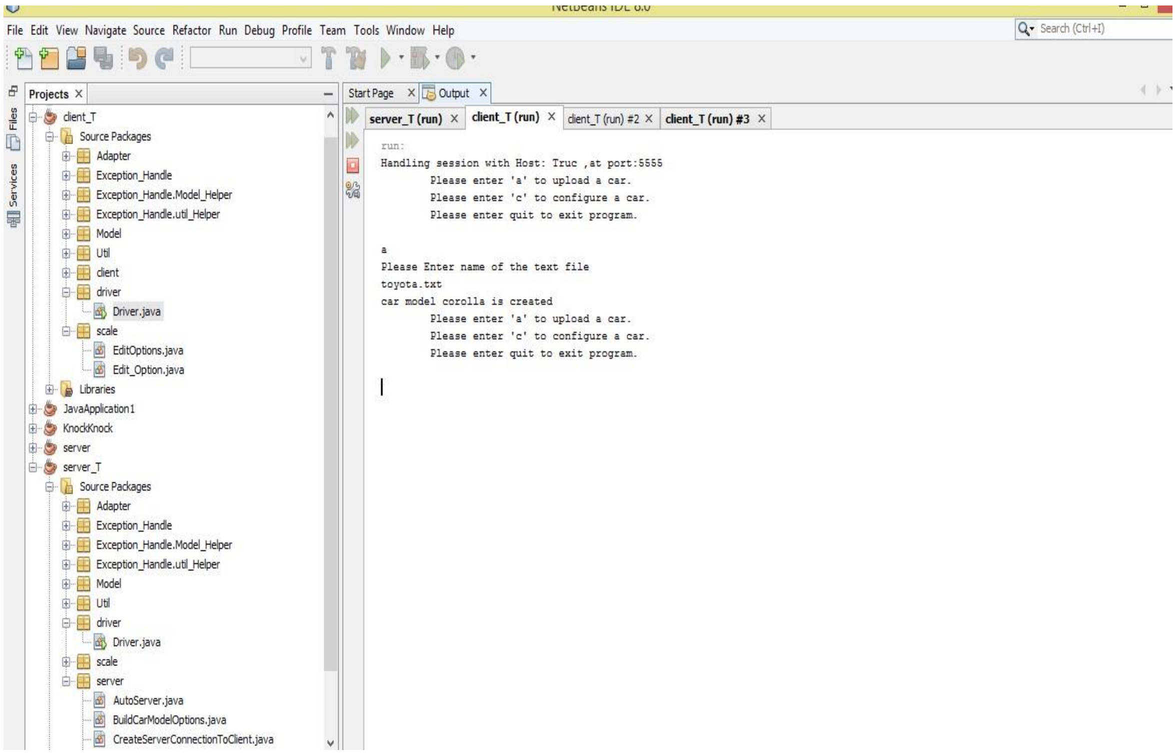
Create multithreading Server that can handle multiple clients. In Image 1, I run one server and three clients. The Server will receive a properties file (data) which upload by the client (#1), parse it, and store it on the server. (Validate data is successfully upload)



```
run:
        Client is connected to server...
        Client is connected to server...
        Client is connected to server...

        Car Model:COROLLA
        Base Price is:15000.0
        DISPLAY SET OF OPTION COLOR.
Option 1 is :Option [ PartName = RED, price = 0.0 ]
Option 2 is :Option [ PartName = BLACK, price = 0.0 ]
Option 3 is :Option [ PartName = WHITE, price = 0.0 ]
Option 4 is :Option [ PartName = BLUE, price = 0.0 ]

        DISPLAY SET OF OPTION TRANSMISSION.
Option 1 is :Option [ PartName = AUTOMATIC, price = 0.0 ]
Option 2 is :Option [ PartName = MANUAL, price = -815.0 ]

        DISPLAY SET OF OPTION BRAKES.
Option 1 is :Option [ PartName = STANDARD, price = 0.0 ]
Option 2 is :Option [ PartName = ABS, price = 400.0 ]
Option 3 is :Option [ PartName = ABS WITH ADVANCE TRAC, price = 1625.0 ]

        DISPLAY SET OF OPTION MIRRORS.
Option 1 is :Option [ PartName = AUTOMATIC, price = 100.0 ]
Option 2 is :Option [ PartName = MANUAL, price = 0.0 ]


        Car Model:COROLLA
        Base Price is:15000.0
        DISPLAY SET OF OPTION COLOR.
Option 1 is :Option [ PartName = RED, price = 0.0 ]
Option 2 is :Option [ PartName = BLACK, price = 0.0 ]
Option 3 is :Option [ PartName = WHITE, price = 0.0 ]
Option 4 is :Option [ PartName = BLUE, price = 0.0 ]
```

Image 1(Image by Truc Huynh, 2014)

After client_T (#1) post (send) the text file (properties file) to the server. Server receive the properties file and parsed the text file to create an instance of Automobile and store it in the Link Hash Map (which is the application database) (Refer to Image 2).

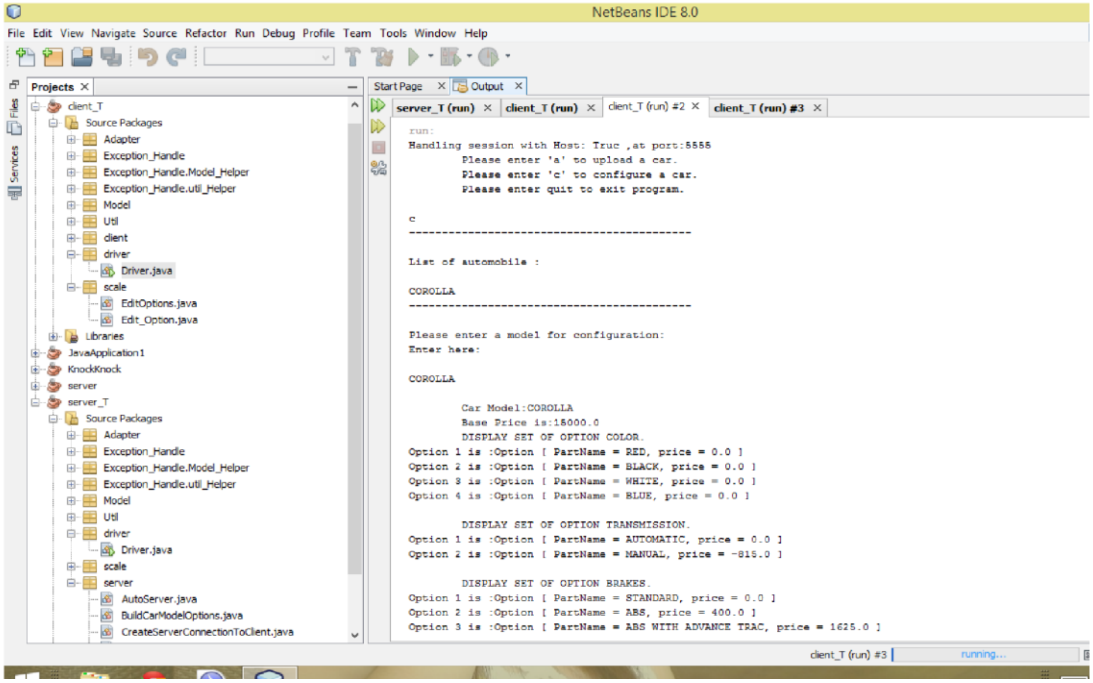


```
run:
Handling session with Host: Truc ,at port:5555
        Please enter 'a' to upload a car.
        Please enter 'c' to configure a car.
        Please enter quit to exit program.


a
Please Enter name of the text file
toyota.txt
car model corolla is created
        Please enter 'a' to upload a car.
        Please enter 'c' to configure a car.
        Please enter quit to exit program.
```

Image 2 (Image by Truc Huynh, 2014)

d. Send Request to Server (#2)

Client-T (#2) send a request to the Server for cars configuration. Then Server will send the respond back (instance of Automobile). (Image 3)

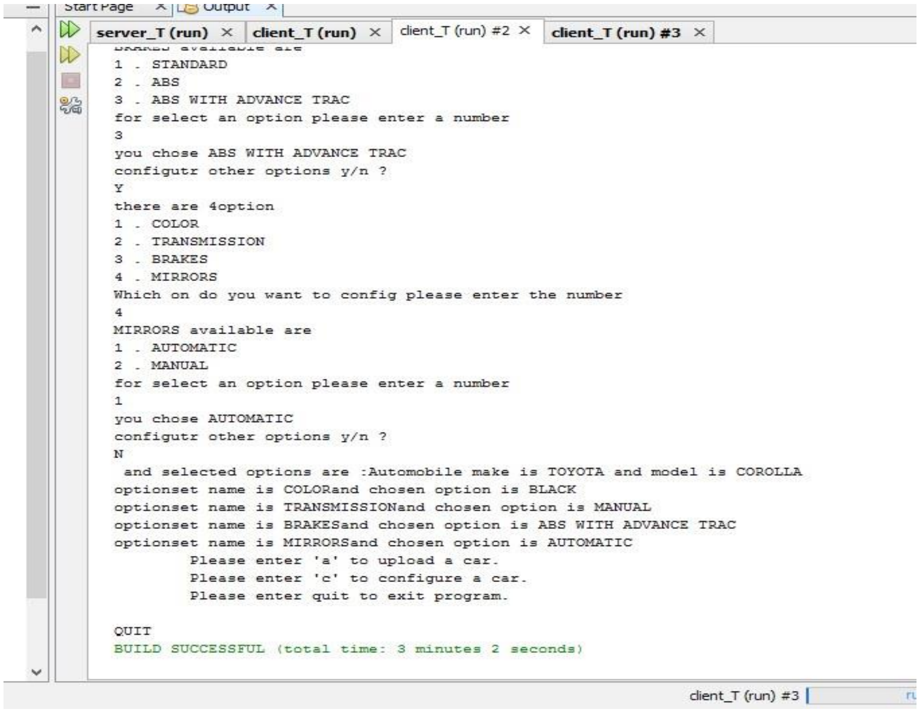


```
run:
Handling session with Host: Truc ,at port:5555
        Please enter 'a' to upload a car.
        Please enter 'c' to configure a car.
        Please enter quit to exit program.

c
-------------------------------------------

List of automobile :

COROLLA
-------------------------------------------

Please enter a model for configuration:
Enter here:

COROLLA

        Car Model:COROLLA
        Base Price is:15000.0
        DISPLAY SET OF OPTION COLOR.
Option 1 is :Option [ PartName = RED, price = 0.0 ]
Option 2 is :Option [ PartName = BLACK, price = 0.0 ]
Option 3 is :Option [ PartName = WHITE, price = 0.0 ]
Option 4 is :Option [ PartName = BLUE, price = 0.0 ]

        DISPLAY SET OF OPTION TRANSMISSION.
Option 1 is :Option [ PartName = AUTOMATIC, price = 0.0 ]
Option 2 is :Option [ PartName = MANUAL, price = -815.0 ]

        DISPLAY SET OF OPTION BRAKES.
Option 1 is :Option [ PartName = STANDARD, price = 0.0 ]
Option 2 is :Option [ PartName = ABS, price = 400.0 ]
Option 3 is :Option [ PartName = ABS WITH ADVANCE TRAC, price = 1625.0 ]
```

Image 3 (Image by Truc Huynh, 2014)

e. Received Request from Server (#2) (#3)

Client-T (#2) will get the choice from this instance of Automobile and store it within this Automobile. Picture below show successfully configurate the Car (Upload the data to server successfully) (Image 4)

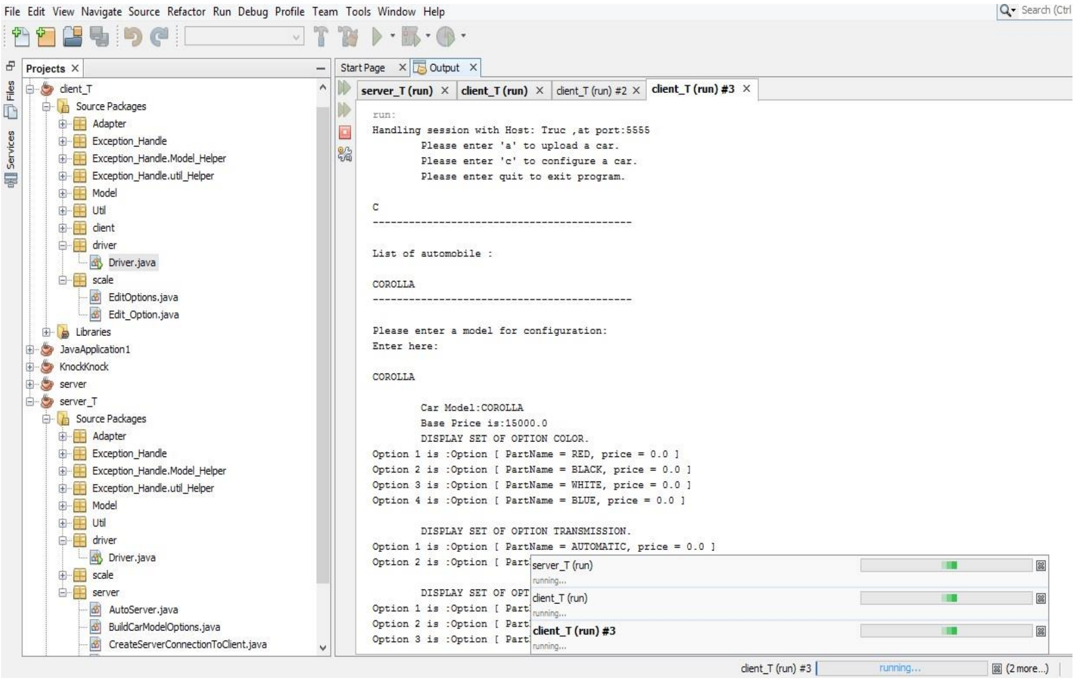


```
1 . STANDARD
2 . ABS
3 . ABS WITH ADVANCE TRAC
for select an option please enter a number
3
you chose ABS WITH ADVANCE TRAC
configutr other options y/n ?
Y
there are 4option
1 . COLOR
2 . TRANSMISSION
3 . BRAKES
4 . MIRRORS
Which on do you want to config please enter the number
4
MIRRORS available are
1 . AUTOMATIC
2 . MANUAL
for select an option please enter a number
1
you chose AUTOMATIC
configutr other options y/n ?
N
 and selected options are :Automobile make is TOYOTA and model is COROLLA
optionset name is COLORand chosen option is BLACK
optionset name is TRANSMISSIONand chosen option is MANUAL
optionset name is BRAKESand chosen option is ABS WITH ADVANCE TRAC
optionset name is MIRRORSand chosen option is AUTOMATIC
        Please enter 'a' to upload a car.
        Please enter 'c' to configure a car.
        Please enter quit to exit program.

QUIT
BUILD SUCCESSFUL (total time: 3 minutes 2 seconds)
```

Image 4 (Image by Truc Huynh, 2014)

f. Send and Received Request from Server (#3)

Call Client-T (#3) to configurate a Car after terminate Client-T (#2) to verify if the data is successful upload to the Server. (Image 5)



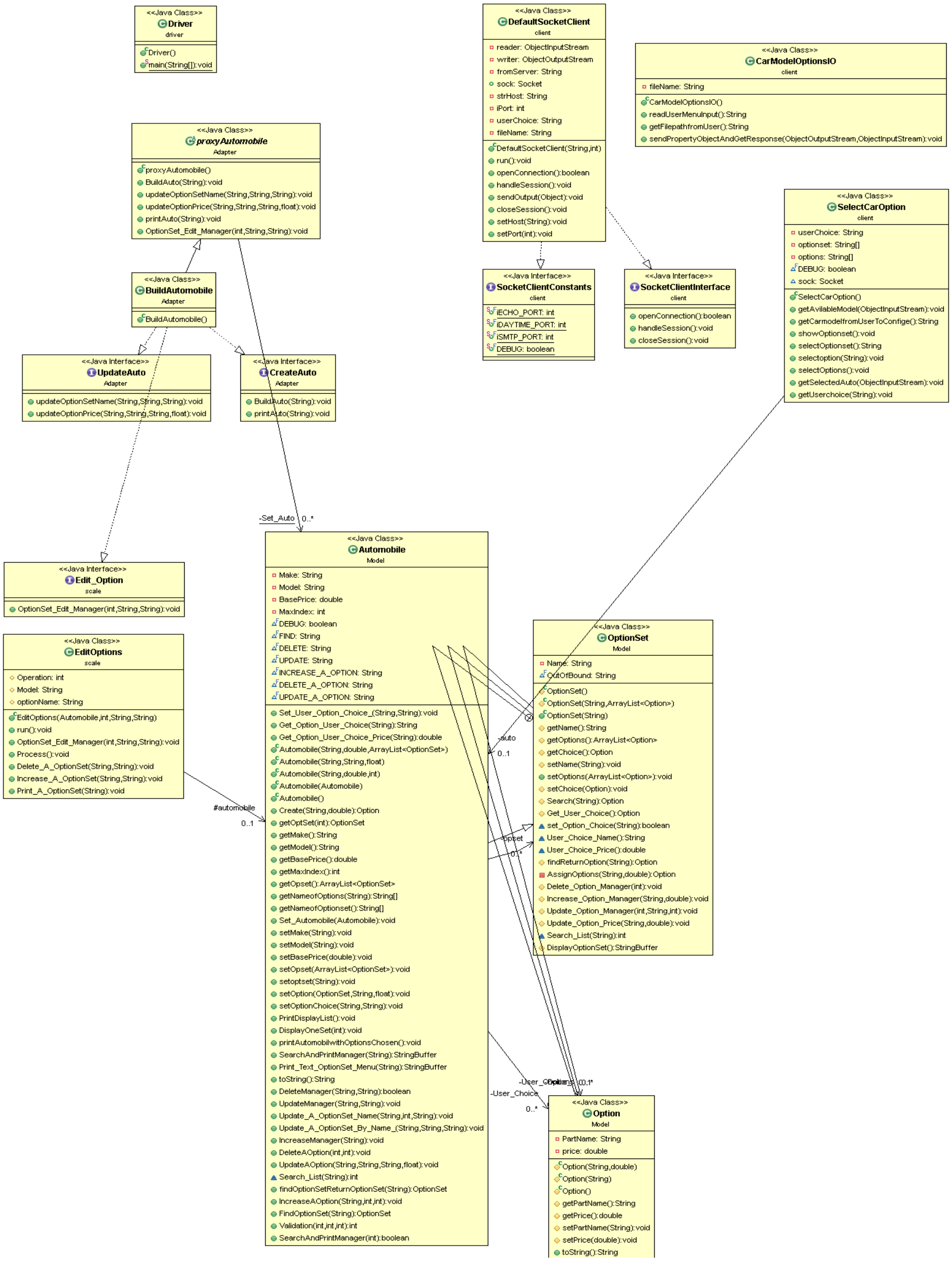
```
run:
Handling session with Host: Truc ,at port:5555
        Please enter 'a' to upload a car.
        Please enter 'c' to configure a car.
        Please enter quit to exit program.

C
------------------------------------------

List of automobile :

COROLLA
------------------------------------------

Please enter a model for configuration:
Enter here:

COROLLA

        Car Model:COROLLA
        Base Price is:15000.0
        DISPLAY SET OF OPTION COLOR.
Option 1 is :Option [ PartName = RED, price = 0.0 ]
Option 2 is :Option [ PartName = BLACK, price = 0.0 ]
Option 3 is :Option [ PartName = WHITE, price = 0.0 ]
Option 4 is :Option [ PartName = BLUE, price = 0.0 ]

        DISPLAY SET OF OPTION TRANSMISSION.
Option 1 is :Option [ PartName = AUTOMATIC, price = 0.0 ]
Option 2 is :Option [ Part

        DISPLAY SET OF OPT
Option 1 is :Option [ Part
Option 2 is :Option [ Part
Option 3 is :Option [ Part
```

Image 5 (Image by Truc Huynh, 2014)

IV. Conclusion:

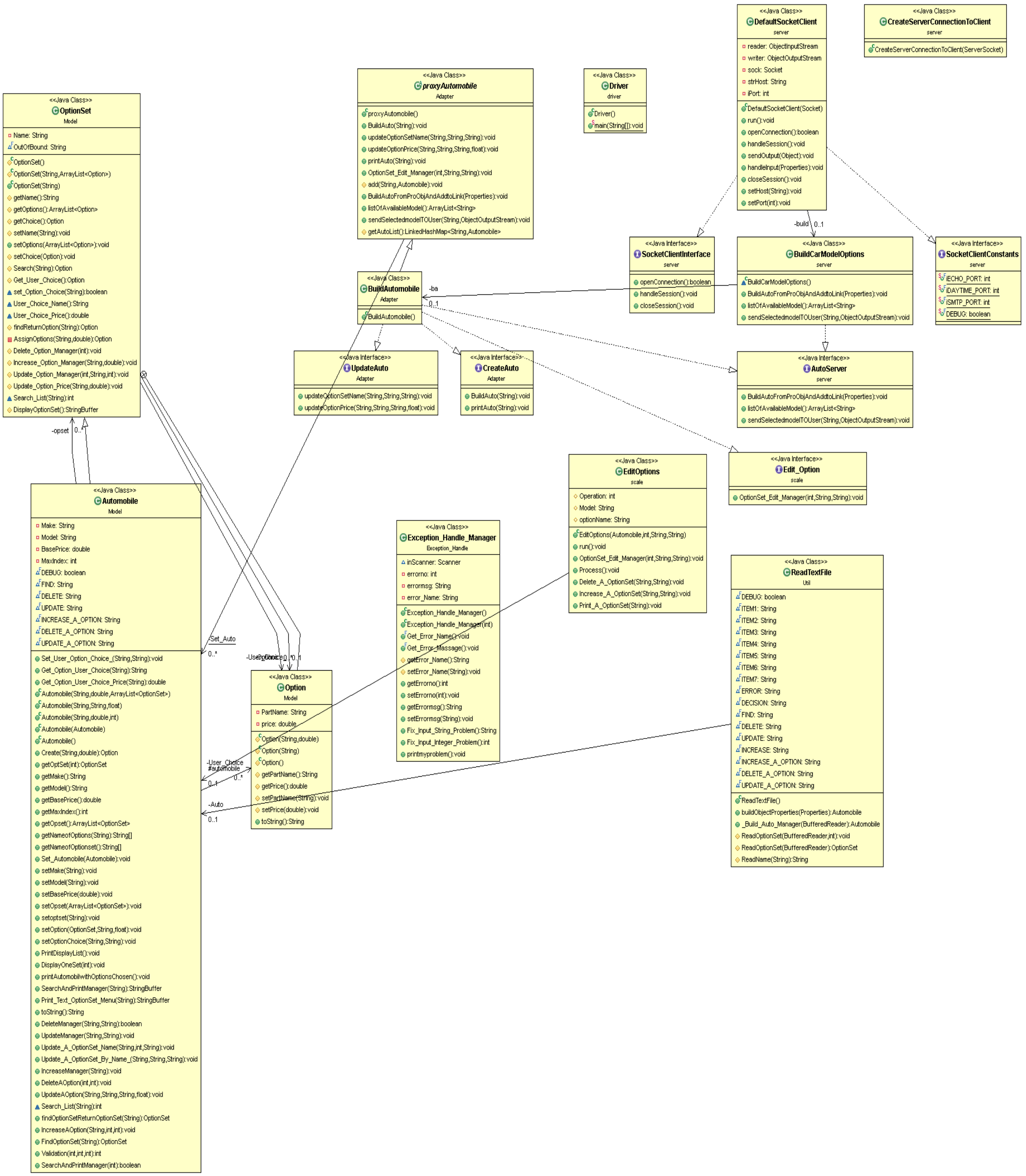
The Server and Clients satisfy basic communication: send and receive request. Server receive upload data and store in its database (instance of Link Hash Map). The clients have ability to insert, update and delete an Automobile. No crashing (Implement Exception Handling)

V. Architecture UML:



<<Java Class>>
Driver
driver
Driver()
main(String[]):void
<<Java Class>>
DefaultSocketClient
client
reader: ObjectInputStream
writer: ObjectOutputStream
fromServer: String
sock: Socket
strHost: String
iPort: int
userChoice: String
fileName: String
DefaultSocketClient(String,int)
run():void
openConnection():boolean
handleSession():void
sendOutput(Object):void
closeSession():void
setHost(String):void
setPort(int):void
<<Java Class>>
CarModelOptionsIO
client
fileName: String
CarModelOptionsIO()
readUserMenuInput():String
getFilepathfromUser():String
sendPropertyObjectAndGetResponse(ObjectOutputStream,ObjectInputStream):void
<<Java Class>>
proxyAutomobile
Adapter
proxyAutomobile()
BuildAuto(String):void
updateOptionSetName(String,String,String):void
updateOptionPrice(String,String,String,float):void
printAuto(String):void
OptionSet_Edit_Manager(int,String,String):void
<<Java Class>>
SelectCarOption
client
userChoice: String
optionset: String[]
options: String[]
DEBUG: boolean
sock: Socket
SelectCarOption()
getAvilableModel(ObjectInputStream):void
getCarmodelfromUserToConfige():String
showOptionset():void
selectOptionset():String
selectoption(String):void
selectOptions():void
getSelectedAuto(ObjectInputStream):void
getUserchoice(String):void
<<Java Class>>
BuildAutomobile
Adapter
BuildAutomobile()
<<Java Interface>>
SocketClientConstants
client
iECHO_PORT: int
iDAYTIME_PORT: int
iSMTP_PORT: int
DEBUG: boolean
<<Java Interface>>
SocketClientInterface
client
openConnection():boolean
handleSession():void
closeSession():void
<<Java Interface>>
UpdateAuto
Adapter
updateOptionSetName(String,String,String):void
updateOptionPrice(String,String,String,float):void
<<Java Interface>>
CreateAuto
Adapter
BuildAuto(String):void
printAuto(String):void
-Set_Auto
0..*
<<Java Class>>
Automobile
Model
Make: String
Model: String
BasePrice: double
MaxIndex: int
DEBUG: boolean
FIND: String
DELETE: String
UPDATE: String
INCREASE_A_OPTION: String
DELETE_A_OPTION: String
UPDATE_A_OPTION: String
Set_User_Option_Choice_(String,String):void
Get_Option_User_Choice(String):String
Get_Option_User_Choice_Price(String):double
Automobile(String,double,ArrayList<OptionSet>)
Automobile(String,String,float)
Automobile(String,double,int)
Automobile(Automobile)
Automobile()
Create(String,double):Option
getOptSet(int):OptionSet
getMake():String
getModel():String
getBasePrice():double
getMaxIndex():int
getOpset():ArrayList<OptionSet>
getNameofOptions(String):String[]
getNameofOptionset():String[]
Set_Automobile(Automobile):void
setMake(String):void
setModel(String):void
setBasePrice(double):void
setOpset(ArrayList<OptionSet>):void
setoptset(String):void
setOption(OptionSet,String,float):void
setOptionChoice(String,String):void
PrintDisplayList():void
DisplayOneSet(int):void
printAutomobilwithOptionsChosen():void
SearchAndPrintManager(String):StringBuffer
Print_Text_OptionSet_Menu(String):StringBuffer
toString():String
DeleteManager(String,String):boolean
UpdateManager(String,String):void
Update_A_OptionSet_Name(String,int,String):void
Update_A_OptionSet_By_Name_(String,String,String):void
IncreaseManager(String):void
DeleteAOption(int,int):void
UpdateAOption(String,String,String,float):void
Search_List(String):int
findOptionSetReturnOptionSet(String):OptionSet
IncreaseAOption(String,int,int):void
FindOptionSet(String):OptionSet
Validation(int,int,int):int
SearchAndPrintManager(int):boolean
<<Java Interface>>
Edit_Option
scale
OptionSet_Edit_Manager(int,String,String):void
<<Java Class>>
EditOptions
scale
Operation: int
Model: String
optionName: String
EditOptions(Automobile,int,String,String)
run():void
OptionSet_Edit_Manager(int,String,String):void
Process():void
Delete_A_OptionSet(String,String):void
Increase_A_OptionSet(String,String):void
Print_A_OptionSet(String):void
#automobile
0..1
<<Java Class>>
OptionSet
Model
Name: String
OutOfBound: String
OptionSet()
OptionSet(String,ArrayList<Option>)
OptionSet(String)
getName():String
getOptions():ArrayList<Option>
getChoice():Option
setName(String):void
setOptions(ArrayList<Option>):void
setChoice(Option):void
Search(String):Option
Get_User_Choice():Option
set_Option_Choice(String):boolean
User_Choice_Name():String
User_Choice_Price():double
findReturnOption(String):Option
AssignOptions(String,double):Option
Delete_Option_Manager(int):void
Increase_Option_Manager(String,double):void
Update_Option_Manager(int,String,int):void
Update_Option_Price(String,double):void
Search_List(String):int
DisplayOptionSet():StringBuffer
-auto
0..1
-opset
0..*
-User_Choice
-options
0..1
0..*
-User_Choice
0..*
<<Java Class>>
Option
Model
PartName: String
price: double
Option(String,double)
Option(String)
Option()
getPartName():String
getPrice():double
setPartName(String):void
setPrice(double):void
toString():String

Client UML, Designed by Truc Huynh


<<Java Class>>
DefaultSocketClient
server
reader: ObjectInputStream
writer: ObjectOutputStream
sock: Socket
strHost: String
iPort: int
DefaultSocketClient(Socket)
run():void
openConnection():boolean
handleSession():void
sendOutput(Object):void
handleInput(Properties):void
closeSession():void
setHost(String):void
setPort(int):void
<<Java Class>>
CreateServerConnectionToClient
server
CreateServerConnectionToClient(ServerSocket)
<<Java Class>>
proxyAutomobile
Adapter
proxyAutomobile()
BuildAuto(String):void
updateOptionSetName(String,String,String):void
updateOptionPrice(String,String,String,float):void
printAuto(String):void
OptionSet_Edit_Manager(int,String,String):void
add(String,Automobile):void
BuildAutoFromProObjAndAddtoLink(Properties):void
listOfAvailableModel():ArrayList<String>
sendSelectedmodelTOUser(String,ObjectOutputStream):void
getAutoList():LinkedHashMap<String,Automobile>
<<Java Class>>
Driver
driver
Driver()
main(String[]):void
<<Java Class>>
OptionSet
Model
Name: String
OutOfBound: String
OptionSet()
OptionSet(String,ArrayList<Option>)
OptionSet(String)
getName():String
getOptions():ArrayList<Option>
getChoice():Option
setName(String):void
setOptions(ArrayList<Option>):void
setChoice(Option):void
Search(String):Option
Get_User_Choice():Option
set_Option_Choice(String):boolean
User_Choice_Name():String
User_Choice_Price():double
findReturnOption(String):Option
AssignOptions(String,double):Option
Delete_Option_Manager(int):void
Increase_Option_Manager(String,double):void
Update_Option_Manager(int,String,int):void
Update_Option_Price(String,double):void
Search_List(String):int
DisplayOptionSet():StringBuffer
-build
0..1
<<Java Interface>>
SocketClientInterface
server
openConnection():boolean
handleSession():void
closeSession():void
<<Java Class>>
BuildCarModelOptions
server
BuildCarModelOptions()
BuildAutoFromProObjAndAddtoLink(Properties):void
listOfAvailableModel():ArrayList<String>
sendSelectedmodelTOUser(String,ObjectOutputStream):void
<<Java Interface>>
SocketClientConstants
server
iECHO_PORT: int
iDAYTIME_PORT: int
iSMTP_PORT: int
DEBUG: boolean
<<Java Class>>
BuildAutomobile
Adapter
BuildAutomobile()
-ba
0..1
<<Java Interface>>
UpdateAuto
Adapter
updateOptionSetName(String,String,String):void
updateOptionPrice(String,String,String,float):void
<<Java Interface>>
CreateAuto
Adapter
BuildAuto(String):void
printAuto(String):void
<<Java Interface>>
AutoServer
server
BuildAutoFromProObjAndAddtoLink(Properties):void
listOfAvailableModel():ArrayList<String>
sendSelectedmodelTOUser(String,ObjectOutputStream):void
-opset
0..*
<<Java Class>>
EditOptions
scale
Operation: int
Model: String
optionName: String
EditOptions(Automobile,int,String,String)
run():void
OptionSet_Edit_Manager(int,String,String):void
Process():void
Delete_A_OptionSet(String,String):void
Increase_A_OptionSet(String,String):void
Print_A_OptionSet(String):void
<<Java Interface>>
Edit_Option
scale
OptionSet_Edit_Manager(int,String,String):void
<<Java Class>>
Automobile
Model
Make: String
Model: String
BasePrice: double
MaxIndex: int
DEBUG: boolean
FIND: String
DELETE: String
UPDATE: String
INCREASE_A_OPTION: String
DELETE_A_OPTION: String
UPDATE_A_OPTION: String
Set_User_Option_Choice_(String,String):void
Get_Option_User_Choice(String):String
Get_Option_User_Choice_Price(String):double
Automobile(String,double,ArrayList<OptionSet>)
Automobile(String,String,float)
Automobile(String,double,int)
Automobile(Automobile)
Automobile()
Create(String,double):Option
getOptSet(int):OptionSet
getMake():String
getModel():String
getBasePrice():double
getMaxIndex():int
getOpset():ArrayList<OptionSet>
getNameofOptions(String):String[]
getNameofOptionset():String[]
Set_Automobile(Automobile):void
setMake(String):void
setModel(String):void
setBasePrice(double):void
setOpset(ArrayList<OptionSet>):void
setoptset(String):void
setOption(OptionSet,String,float):void
setOptionChoice(String,String):void
PrintDisplayList():void
DisplayOneSet(int):void
printAutomobilwithOptionsChosen():void
SearchAndPrintManager(String):StringBuffer
Print_Text_OptionSet_Menu(String):StringBuffer
toString():String
DeleteManager(String,String):boolean
UpdateManager(String,String):void
Update_A_OptionSet_Name(String,int,String):void
Update_A_OptionSet_By_Name_(String,String,String):void
IncreaseManager(String):void
DeleteAOption(int,int):void
UpdateAOption(String,String,String,float):void
Search_List(String):int
findOptionSetReturnOptionSet(String):OptionSet
IncreaseAOption(String,int,int):void
FindOptionSet(String):OptionSet
Validation(int,int,int):int
SearchAndPrintManager(int):boolean
<<Java Class>>
Exception_Handle_Manager
Exception_Handle
inScanner: Scanner
errorno: int
errormsg: String
error_Name: String
Exception_Handle_Manager()
Exception_Handle_Manager(int)
Get_Error_Name():void
Get_Error_Massage():void
getError_Name():String
setError_Name(String):void
getErrorno():int
setErrorno(int):void
getErrormsg():String
setErrormsg(String):void
Fix_Input_String_Problem():String
Fix_Input_Integer_Problem():int
printmyproblem():void
<<Java Class>>
ReadTextFile
Util
DEBUG: boolean
ITEM1: String
ITEM2: String
ITEM3: String
ITEM4: String
ITEM5: String
ITEM6: String
ITEM7: String
ERROR: String
DECISION: String
FIND: String
DELETE: String
UPDATE: String
INCREASE: String
INCREASE_A_OPTION: String
DELETE_A_OPTION: String
UPDATE_A_OPTION: String
ReadTextFile()
buildObjectProperties(Properties):Automobile
_Build_Auto_Manager(BufferedReader):Automobile
ReadOptionSet(BufferedReader,int):void
ReadOptionSet(BufferedReader):OptionSet
ReadName(String):String
-Set_Auto
0..*
-User_Choice
-Options
0..*
0..1
<<Java Class>>
Option
Model
PartName: String
price: double
Option(String,double)
Option(String)
Option()
getPartName():String
getPrice():double
setPartName(String):void
setPrice(double):void
toString():String
-User_Choice
#automobile
0..1
0..*
-Auto
0..1

Server UML, Designed by Truc Huynh