

Classic ToolShop Desktop App with API

This is a joint, cross-course project with ENSF 607/608. It is a classic Tool Shop desktop application written in Java.

Contributors

Ziad Chemali 10109966

Stanley Shizheng Chen 10084134

Quick Start

Client and remote server

1. fulfill dependencies in `pom.xml` if builds are missing packages
2. making sure remote backend server is up and running. (See note below)
3. to connect to the remote client, compile and run `src/client/main.java`

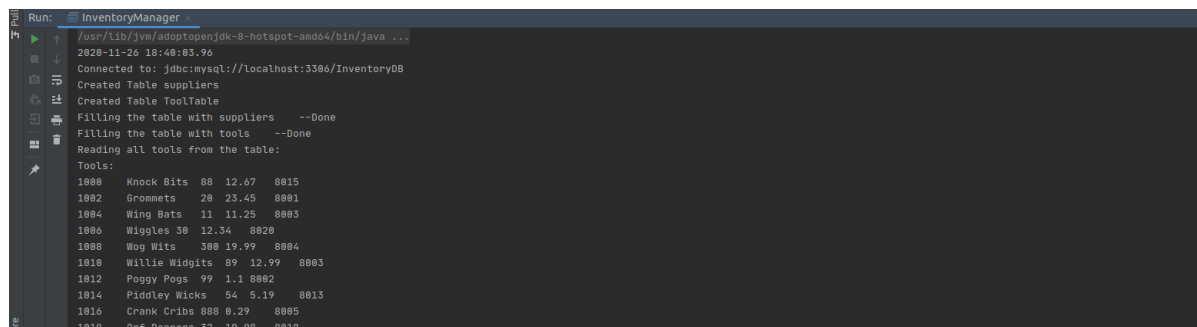
Client and local server

1. fulfill dependencies in `pom.xml` if builds are missing packages
2. to start the server, compile and run `src/server/startServerLocal.java` (See note below)
3. to start the local client, compile and run `src/client/mainLocal.java`

Pre-project Demo

to run the pre-project demo, please compile and run `src/preproject/InventoryManager.java`.

An running mysql is required to be hosted at localhost:3306



```
Run: InventoryManager
2020-11-26 18:40:03.96
Connected to: jdbc:mysql://localhost:3306/InventoryDB
Created Table suppliers
Created Table ToolTable
Filling the table with suppliers --Done
Filling the table with tools --Done
Reading all tools from the table:
Tools:
1000 Knock Bits 88 12.67 8015
1002 Grommets 20 23.45 8001
1004 Wing Bats 11 11.25 8003
1006 Wiggles 30 12.34 8020
1008 Wog Wits 300 19.99 8004
1010 Willie Widdits 89 12.99 8003
1012 Pogygy Pogs 99 1.1 8002
1014 Piddle Wicks 54 5.19 8013
1016 Crank Cribs 888 0.29 8005
1018 Dnf Danners 32 19.98 8018
```

After running the InventoryManager.java, the program should generate log similar to the following:

```
2020-11-26 18:40:03.96
Connected to: jdbc:mysql://localhost:3306/InventoryDB
Created Table suppliers
Created Table ToolTable
Filling the table with suppliers --Done
```

Filling the table with tools --Done

Reading all tools from the table:

Tools:

1000	Knock Bits	88	12.67	8015
1002	Grommets	20	23.45	8001
1004	Wing Bats	11	11.25	8003
1006	Wiggles	30	12.34	8020
1008	Wog Wits	300	19.99	8004
1010	Willie Widgits	89	12.99	8003
1012	Poggy Pogs	99	1.1	8002
1014	Piddley Wicks	54	5.19	8013
1016	Crank Cribs	888	0.29	8005
1018	Orf Dappers	32	19.98	8018
1020	Knit Piks	66	6.75	8015
1022	Tic Tocs	87	1.36	8014
1024	Thinga-ma-bobs	40	10.98	8012
1026	Barn Bins	45	88.67	8006
1028	Pong Pangs	2345	0.1	8002
1030	Nic Nacs	238	2.99	8015
1032	Lilly Larks	56	12.99	8007
1034	Cork Handles	654	2.66	8016
1036	Erk Orks	498	3.99	8017
1038	Googly Eyes	756	6.99	8001
1040	Inny Outies	219	3.45	8010

Reading all suppliers from the table:

Suppliers:

8001	Grommet Builders	788	30th St., SE, Calgary	Fred
8002	Pong Works	749	Dufferin Blvd., SE, Calgary	Bart
8003	Wiley Inc.	26	40th St., SE, Calgary	BillyBob
8004	Winork Manufacturing Inc.	156	51st Ave., SE, Calgary	Marty
8005	Grab Bag Inc.	138	42nd Ave., NE, Calgary	Monty
8006	Paddy's Manufacturing	311	McCall Way, NE, Calgary	Barney
8007	Smickles Industries	966	Smed Lane, SE, Calgary	Lisa
8008	Thangs Inc.	879	37th Ave., NE, Calgary	Thelma
8009	Flip Works Inc.	472	Ogden Dale Rd., SE, Calgary	Rory
8010	Irkle Industries	754	Sunridge Way, NE, Calgary	Irma
8011	Biff Builders	690	19th St., NE, Calgary	Borjn
8012	Twinkles Inc.	318	29th St., NE, Calgary	Barkley
8013	Piddles Industries	238	112th Ave., NE, Calgary	Parnell
8014	Tic Tac Manufacturing	598	Palmer Rd., NE, Calgary	Teisha
8015	Knock Knock Inc.	363	42nd Ave., NE, Calgary	Ned
8016	Corky Things Inc.	35	McCall Way, NE, Calgary	Corrine
8017	E & O Industries	883	44th St., SE, Calgary	Stan
8018	Fiddleys Makes Stuff Inc.	350	27th St., NE, Calgary	Fredda
8019	Horks and Stuff Manufacturing	997	42nd Ave., NE, Calgary	Harold
8020	Wings Works	754	48th St., SE, Calgary	Wing

Searching table for tool 1002: should return 'Grommets'

Search Result: 1002 Grommets 20 23.45 8001

Searching table for tool 9000: should fail to find a tool

Search Failed to find a tool matching id: 9000

The program is finished running

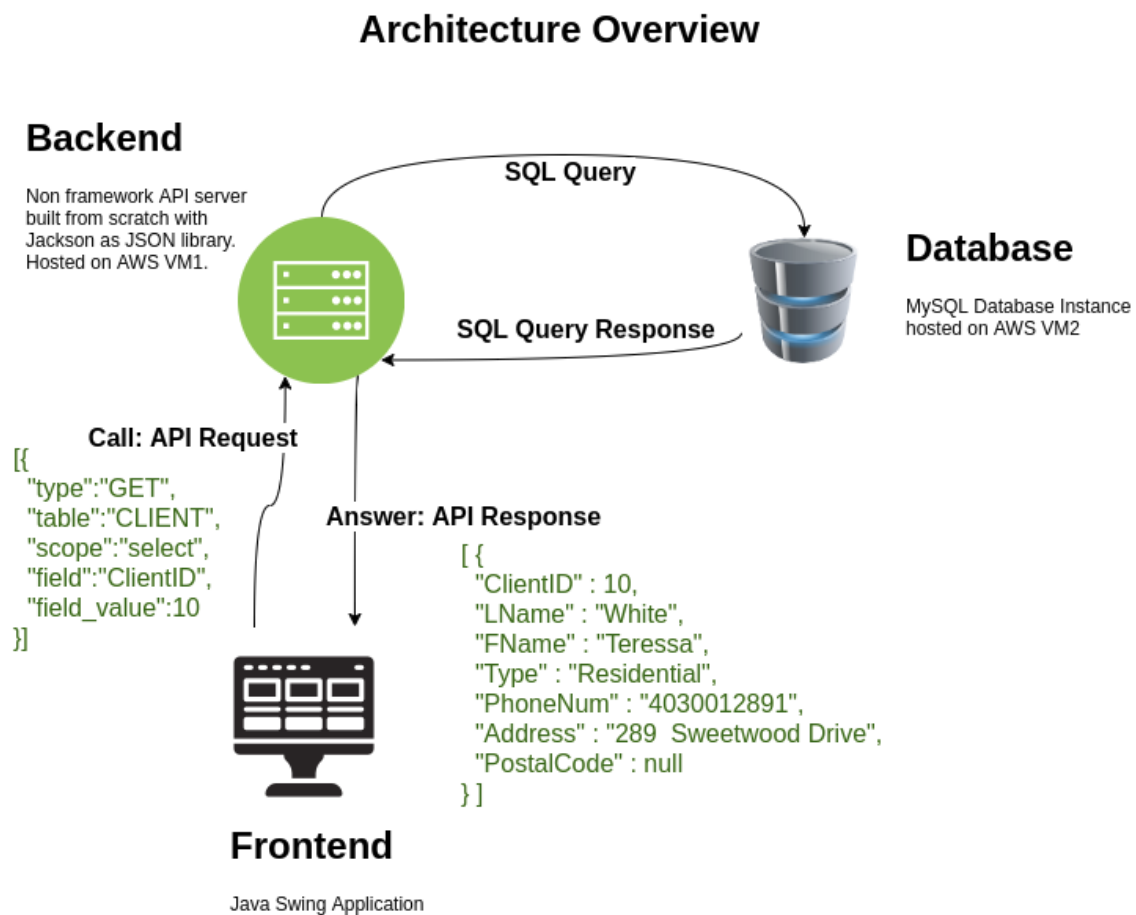
Process finished with exit code 0

Summary

Inventory management system that has two functionalities:

- Allows owner to add new customers/update their information and allows customers to search for tools from the data base and purchase.
- If the quantity of an item goes below 40 items then the program automatically generates an order line for that item

Architecture



Front-End

The front end is a GUI application that will send messages in a JSON format to the server via sockets, then it will receive a response. Example if search by ToolId is selected a we are searching for a ToolID of 8000 then it will send a JSON message as follow:

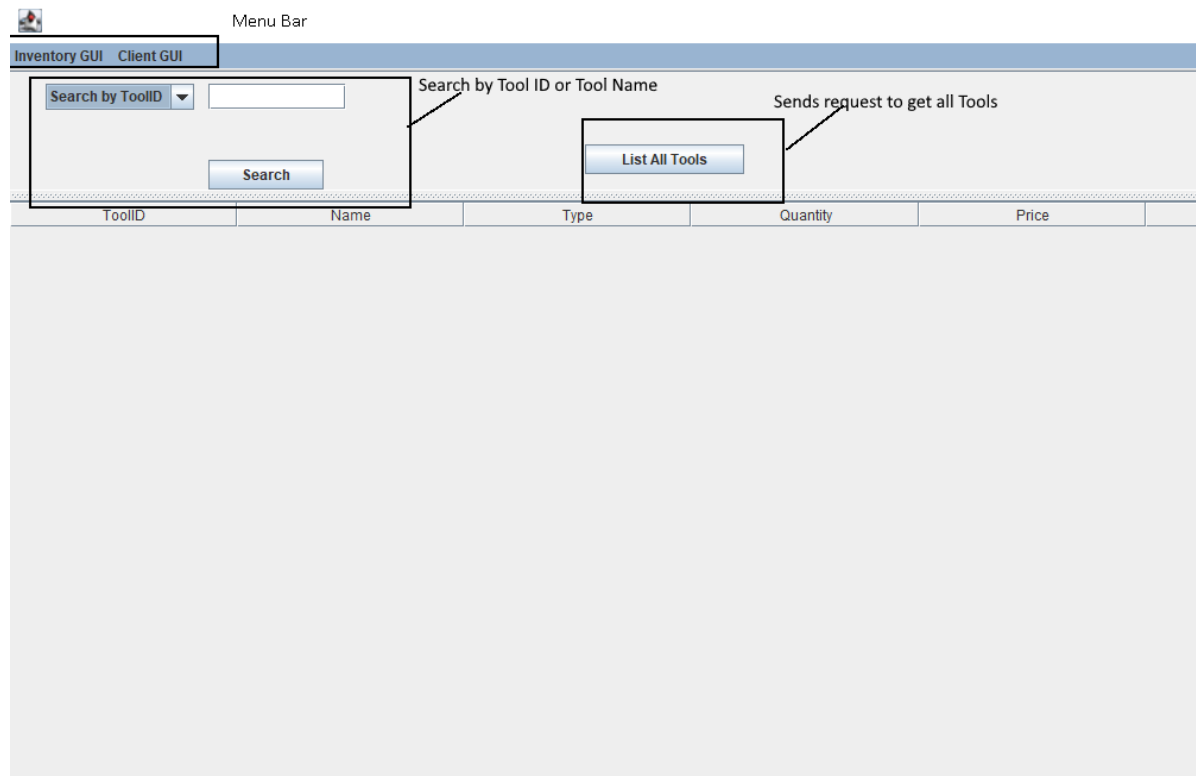
```
{ "type" : "GET",
  "table" : "TOOL" ,
  "scope": "select", "field": "ToolID", "field_value": "8000"};
```

The response is a JSON Node as follows [{ "ToolID" : 8000, "Name" : "Knock Bitzzz", "Type" : "Electric", "Quantity" : 70, "Price" : 15.0, "SupplierID" : 8004, "PowerType" : null}]

And it will be printed in the GUI

GUI Panels

1. Inventory Panel



Note: If you press List All Tools or Search (with correct ToolID or Tool Name) the following table is generated

The screenshot shows the 'Inventory GUI Client GUI' window with the same search and list all tools buttons. A note with an arrow points to the table, stating 'If you press on any row , it will direct you to Purchase panel'. The table contains the following data:

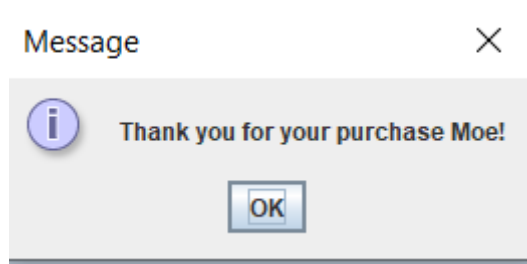
ToolID	Name	Type	Quantity	Price	SupplierID	PowerType
8000	Knock Bitzzz	Electric	70	15.0	8004	
1095	Heavy Duty Blower	Electric	50	150.0	8022	
1094	Tester	Electric	50	150.0	8022	Battery
1042	Circular Saw	Electric	42	300.0	8022	
1041	Cordless Drill	Electric	50	200.0	8022	USB
1040	Inny Outies	Non_Electric	219	3.45	8010	
1039	Handy Pandles	Non_Electric	298	4.35	8017	
1038	Googly Eyes	Non_Electric	726	6.99	8001	
1037	Foo Figs	Non_Electric	194	5.89	8018	
1036	Erk Orks	Non_Electric	498	3.99	8017	
1035	Ding Dams	Non_Electric	1208	0.15	8019	
1034	Cork Handles	Non_Electric	654	2.66	8016	
1033	Minnie Morks	Non_Electric	800	1.95	8007	
1032	Lilly Larks	Non_Electric	56	12.99	8007	
1031	Tork Tins	Non_Electric	376	0.95	8012	
1030	Nic Nacs	Non_Electric	238	2.99	8015	
1029	Oof Tonges	Non_Electric	345	8.49	8011	
1028	Pong Pangs	Non_Electric	2345	0.1	8002	
1027	Flap Wrappers	Non_Electric	89	44.88	8009	
1026	Barn Bins	Non_Electric	45	88.67	8006	
1025	Fling Flobs	Non_Electric	254	2.33	8009	
1024	Thinga-ma-bobs	Non_Electric	40	10.98	8012	
1023	Tin Wits	Non_Electric	50	5.67	8014	
1022	Tic Tocs	Non_Electric	87	1.36	8014	
1021	Piddley Pins	Non_Electric	340	0.25	8020	
1020	Knit Piks	Non_Electric	66	6.75	8015	

2. Purchase Panel

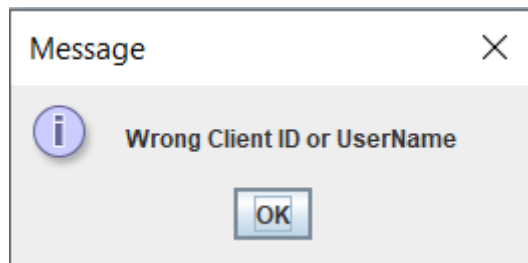
After selecting a row from the table, purchase panel will appear

The screenshot shows a window titled "Inventory GUI Client GUI". At the top, there is a search bar with a dropdown menu labeled "Search by ToolID" and a text input field. Below the search bar are two buttons: "Search" and "List All Tools". The main area is divided into two sections. On the left, under "Tool Info", there is a list of tool details: ToolID 1095, Tool Name Heavy Duty Blower, Type Electric, Price 150.0, Quantity 22, and Total 3300.0. On the right, under "Customer Info", there is a "Purchase" button. A callout box points to the "Quantity" field with the text "Quantity required to purchase". Another callout box points to the "ClientID" and "First Name" fields with the text "Enter Customer Correct ID and FirstName".

if the customer ID/First Name are correct then purchase will be generated and the following pop-up will appear

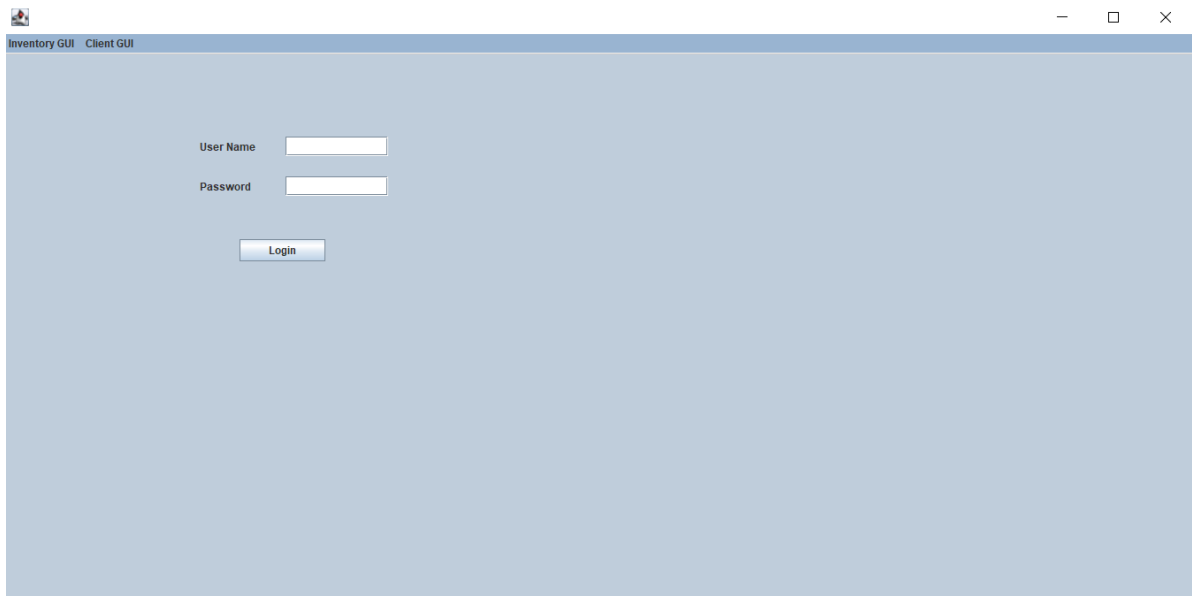


else this pop-up will appear

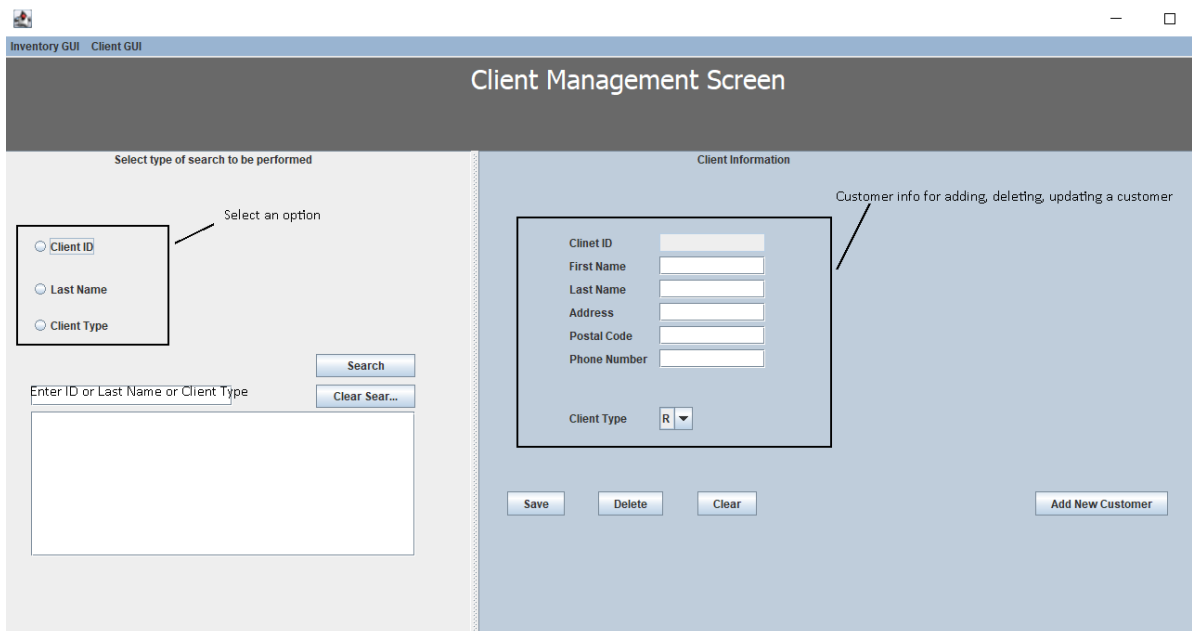


3. CMS Panel

press on ClientGUI option from the menu bar, it will direct you to a Login panel

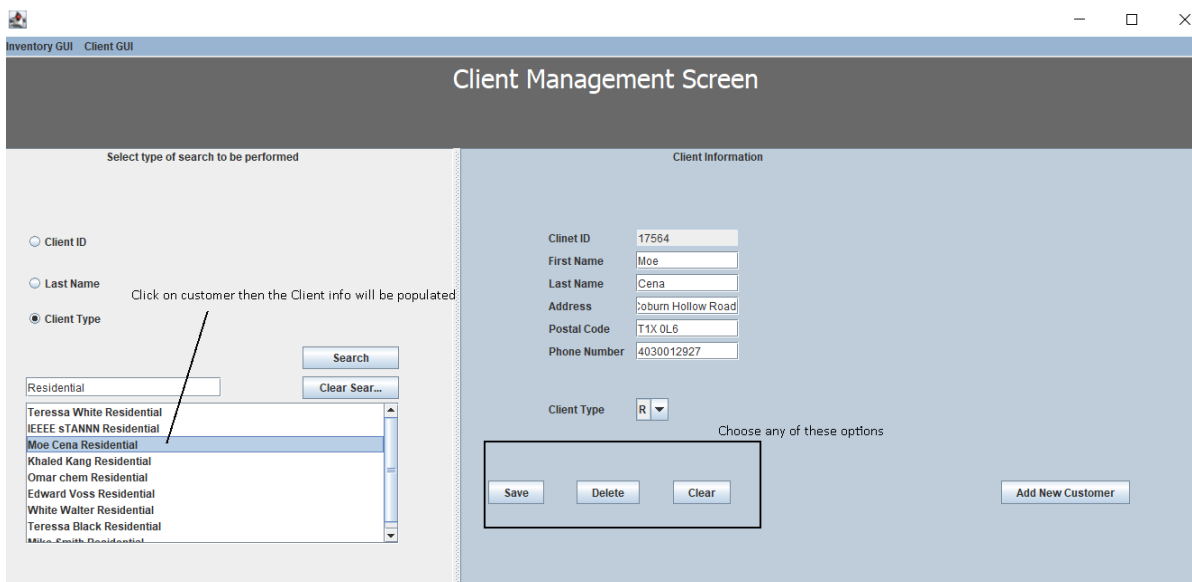


If username/password are correct then the client management panel will show



Updating/Deleting a customer

First search for a customer then select the customer from the JList. After selecting a row, the Client Information in the right side will be populated automatically



Adding new customer

Simply fill in all the fields required, and click Add New Customer. Note: a unique client Id will be generated

Client Information

Client ID

First Name

Last Name

Address

Postal Code

Phone Number

Client Type

Click Add Customer

Double check by searching for the newly added customer

Select type of search to be performed

☐ Client ID

☒ Last Name

☐ Client Type

Ziad Chemali Residential

Customer successfully added

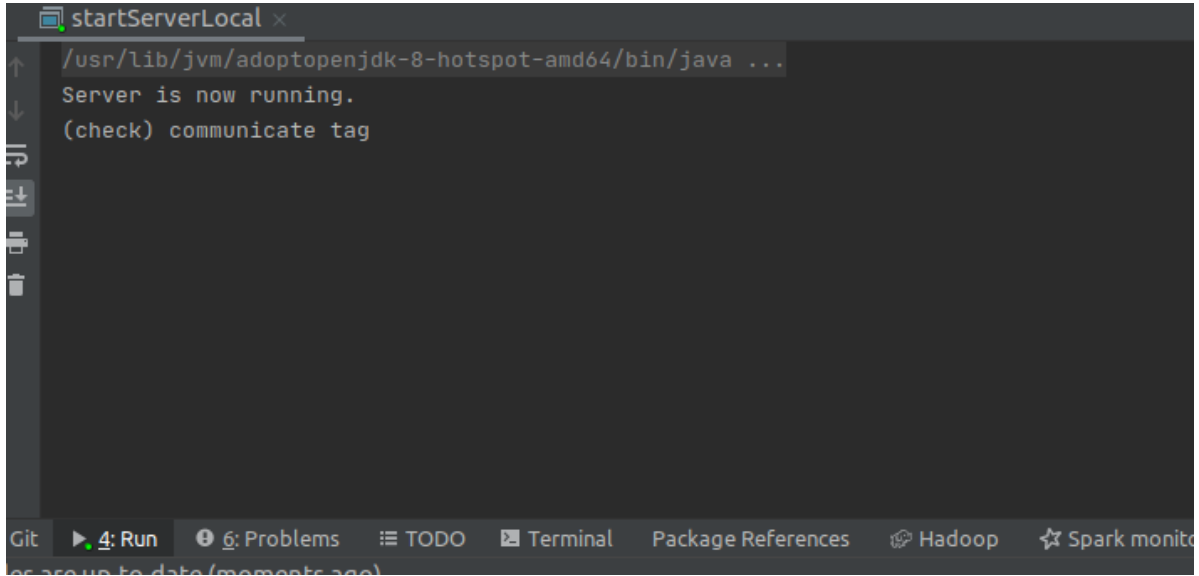
Back-End

How to start local backend server

to start the local backend server, compile and run

```
src/server/startServerLocal.java
```

if the console prints the following, it means the server is ready.



```
startServerLocal x
/usr/lib/jvm/adoptopenjdk-8-hotspot-amd64/bin/java ...
Server is now running.
(check) communicate tag
```

The screenshot shows an IDE interface with a terminal window titled 'startServerLocal'. The terminal output shows the command execution path, a confirmation message 'Server is now running.', and a prompt '(check) communicate tag'. The IDE's status bar at the bottom includes icons for Git, Run, Problems, TODO, Terminal, Package References, Hadoop, and Spark monitor.

How to start remote backend server


Note: For temporary demo purpose, Prof & TA can access:


Our jenkins build server <http://54.185.156.100:8080/>

username: ensf


password: ensf607

If server is not running, build the job "Final Project Demo" to start the backend server.

 Jenkins



Stan Chen

 log out

Jenkins

New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Lockable Resources

New View

Build Queue

No builds in the queue.
















Build Executor Status

1 Idle

2 Idle

All

+

S	W	Name	Last Success	Last Failure	Last Duration	
		Final Project Demo	N/A	N/A	N/A	
		live_test_2	N/A	4 days 5 hr - #9	9.4 sec	
		test stan dev	N/A	1 day 17 hr - #14	8.1 sec	
		toolshop_staging	N/A	7 days 14 hr - #1	10 sec	
		ziad_dev_test	N/A	N/A	N/A	

Icon: S M L

Legend

Atom feed for all

Atom feed for failures

Atom feed for just latest builds

add description

click into the project

Back to Project

Status

Changes


Console Output


Edit Build Information


Git Build Data


No Tags

Previous Build

 Build #17

 No changes.

 Started by user [Star](#)

 git

Revision: d0dbc7ec

• refs/remotes/

In this job page, click `Console Output` ,

```

[1;33mWARNING[m]
[1;33mWARNING[m] It is highly recommended to fix these problems because they threaten the stability of your
build.
[1;33mWARNING[m]
[1;33mWARNING[m] For this reason, future Maven versions might no longer support building such malformed projects.
[1;33mWARNING[m]
[1;34mINFO[m]
[1;34mINFO[m] [1m-----< [0;36morg.example:toolshop-server[0;1m >-----[m]
[1;34mINFO[m] [1mBuilding toolshop-server 1.0-SNAPSHOT[m]
[1;34mINFO[m] [1m-----[ jar ]-----[m]
[1;34mINFO[m] [1m
[1;34mINFO[m]
[1;34mINFO[m] [1m-- [0;32mexec-maven-plugin:3.0.0:java[m [1m(default-cli)[m @ [36mtoolshop-server[0;1m --
-[m]
Server is now running.
(check) communicate tag

```

REST API Jenkins 2.249.3

If the end of the console log prints the logs as above, this means the server is ready to serve.

Testing

Jenkins were introduced as a tool to help with the deployment/testing purposes. Script used for compiling and running the backend server in Jenkins build execute shell:

```

# export CLASSPATH=jars/*.jar
mvn -v
echo "building maven package"
mvn clean install -U
mvn clean package
echo "initing server"
echo "building maven package"
mvn package
echo "initing server"
mvn exec:java -Dexec.mainClass="server.controller.ServerController" -
Dexec.classpathScope=runtime

```

Develop setup

1. this is classic java project. `jars/mysql-connector-java-8.0.22.jar` needs to be added to class path.
2. Alternatively, if using maven, just use pom.xml to fullfil the dependencies.