Dalhousie University

Lynda Ofume

B00738568

CSCI 4140: Advanced Databases

December 3, 2022

Table of Contents

Assignment part declaration

The assignment section that I will be submitting is Part A specs.

Description of Java Single Program Emulation

Based on discussion with Prof Peter, we decided that another way to clearly submit to the coordinator and log the information from console was to use a log file for each scenario.

The code file is described as follows:

Company X:

1. The user first signs on to the system using the method *loginIn568()*, and then the user chooses which company they are logging to.
2. When you choose your company (i.e., Company X), the console will display "Get Ready X" and would log the sequence, the message "Get Ready X" and the timestamp (allowing for accuracy of information to be logged & updated).
3. Next, the user must wait for the next dialogue to then display the function options to choose from.
    a. Function #1: Prints parts list
    b. Function #2: Helps the user submit a Purchase Order
    c. Function #3: List Purchase Order
    d. Function #4: List Lines of a Purchase Order
4. Since we are committing a purchase order (PO), this is when we want to understand whether the user is wanting to commit to the next phase of ordering.
5. The user selects option 2 to proceed in the console.
6. In *function #2 (starting at line 75 of code)* there are break statements for the user to confine by – only proceeding if the right paraments are met. So, if choice equals 2 and *companyNo is 1* this will proceed for Company X.
7. The coordinator asks the user "Ready to commit by X?", the user then must answer yes or no. "Ready to commit by X?" is submitted to the logRecordX.txt as well as
    a. If *yes* is selected: The response of "yes" is logged to the record file with the sequence number and timestamp. The user will wait to proceed to input the information necessary to submit a PO. Then upon completion the user will receive a "Complete!" message.
    b. If *no* is selected: The response of "no" will be recorded to the log record and an "ABORT X" message will be displayed and this message will also be logged to the console with the sequence number and timestamp.
8. All other functions outlined in #3 of this description will proceed the same as they have in A3. Printing the parts lists for the company it is signed on to, listing the purchase order, and listing the lines of a PO.

Company Y:

- All functionality listed in Company X description above is copied over for Company Y, but switch the company X for Company Y where stated. The only difference is the log records for company Y are stored in logRecordY.txt. Keeping the log files separate allows for all companies to have an accurate record of the information being exchanged by the coordinator and the user in the console log.

Company Z:

- All functionality listed in Company X description above is copied over for Company Z, but switch the company X for Company Z where stated. The only difference is the log records for company Z are stored in logRecordZ.txt. Keeping the log files separate allows for all companies to have an accurate record of the information being exchanged by the coordinator and the user in the console log. There is also the listing of parts to be combined from X and Y due to Company Z being the reseller of the parts and being able to submit an order and display those options in 3 and 4 function.

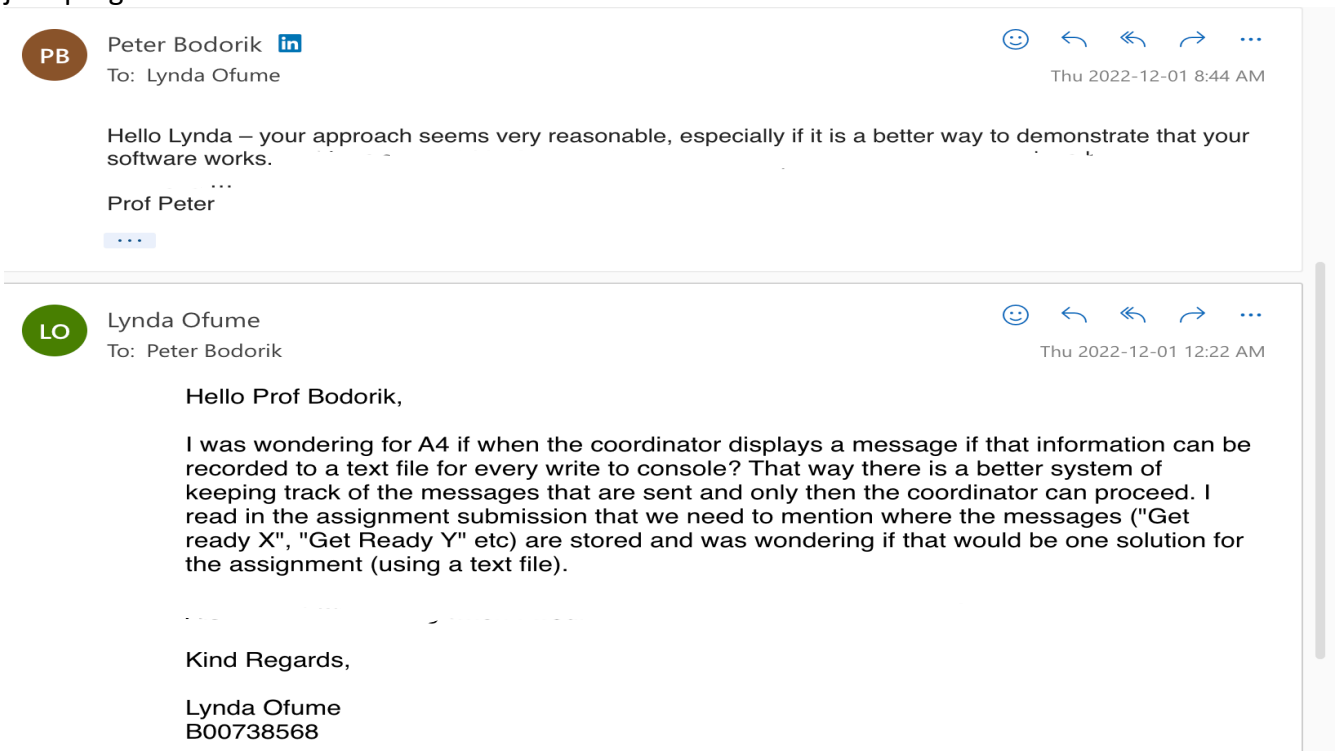
List of DB Server and Software

Framework used: IntelliJ IDEA Community Edition, JDBC, MySQL Workbench used on Localhost 3306, and DB Browser Plugin

Description of DB Schema

The database was created using the MySQL Workbench on the local server. When using the system all company functionality and insert statements are added. Company X, Y, and Z have their own separate functionality that allows for input in the console.

When the user signs in and/or submits a PO, it will log the information using console log and write the 2-PC log records to LogRecordX.txt, LogRecordY.txt, and LogRecordZ.txt for this single java program. As confirmed here:



PB Peter Bodorik
To: Lynda Ofume
Thu 2022-12-01 8:44 AM

Hello Lynda – your approach seems very reasonable, especially if it is a better way to demonstrate that your software works.

Prof Peter

LO Lynda Ofume
To: Peter Bodorik
Thu 2022-12-01 12:22 AM

Hello Prof Bodorik,

I was wondering for A4 if when the coordinator displays a message if that information can be recorded to a text file for every write to console? That way there is a better system of keeping track of the messages that are sent and only then the coordinator can proceed. I read in the assignment submission that we need to mention where the messages ("Get ready X", "Get Ready Y" etc) are stored and was wondering if that would be one solution for the assignment (using a text file).

Kind Regards,

Lynda Ofume
B00738568

Description of UI (single JAVA program)

The UI used in this instance is through the console log system when the program is run on IntelliJ IDEA CE. The user must read the messages sent from the coordinator to respond accordingly in the dialog box. These messages will show up as questions to the user for logging records and for submission or displaying of information they will be statements.

There are specific indicators in the messages displayed to the console of how and what the user should respond with. If the user fails to answer correctly, they will be unable to proceed with the system. Please view the Working Software Screenshots for more information.

Software used for Development (single JAVA program)

IntelliJ IDEA 2022.2.3 (Community Edition)
Build #IC-222.4345.14, built on October 5, 2022
Runtime version: 17.0.4.1+7-b469.62 x86_64
VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.
macOS 13.0
GC: G1 Young Generation, G1 Old Generation
Memory: 1024M
Cores: 8
Metal Rendering is ON
Non-Bundled Plugins:
  PythonCore (222.4345.14)
  DBN (3.3.2160.0)
Kotlin: 222-1.7.10-release-334-IJ4345.14

REST Services specification

No REST Services were used.

Accessing Git Repository

Gitlab access: https://git.cs.dal.ca/ofume/csci_4140_b00738568_a4.git

User needs to be logged in to the Dalhousie Faculty of Computer Science GitLab website. The user can then use the above link to access the database. Download or view the code through GitLab, Web IDE, or by downloading and opening it up on IntelliJ IDEA CE or any equivalent server that recognizes java and the dependencies listed above. Navigate to folder CSCI 4140-A4 for the source code. All record files are in csci_4140_b00738568_a4.

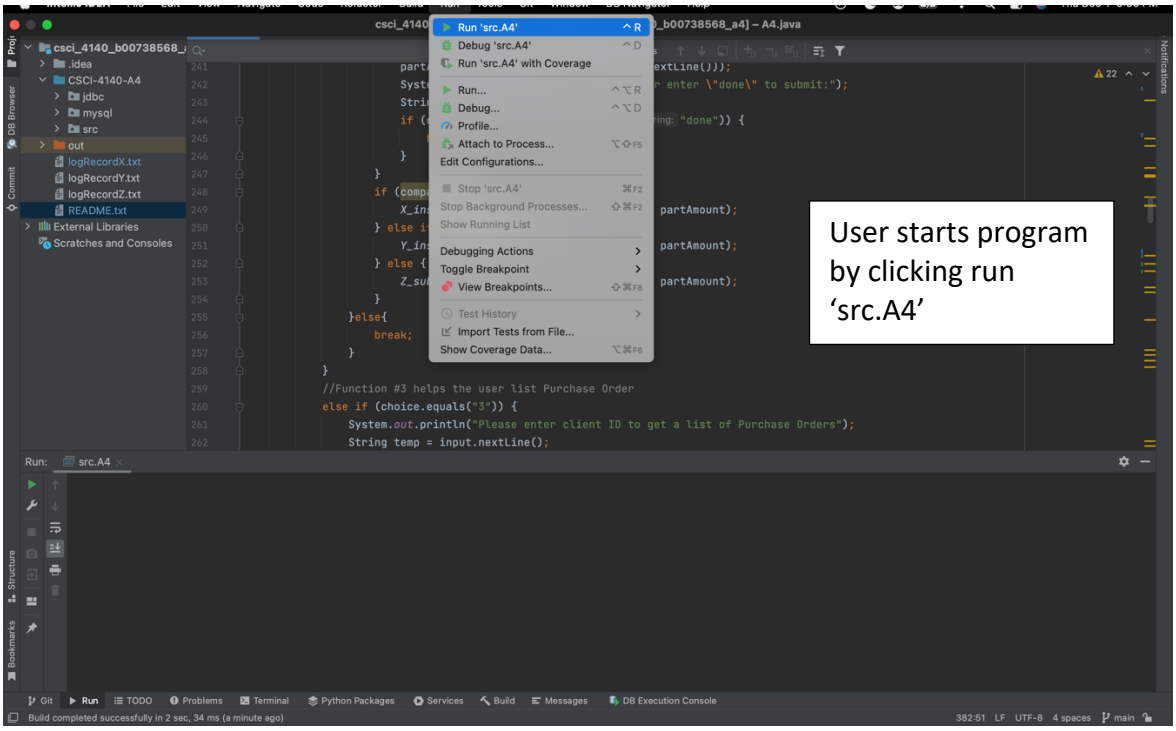
Issues displayed from the assignment

- No issues found in program

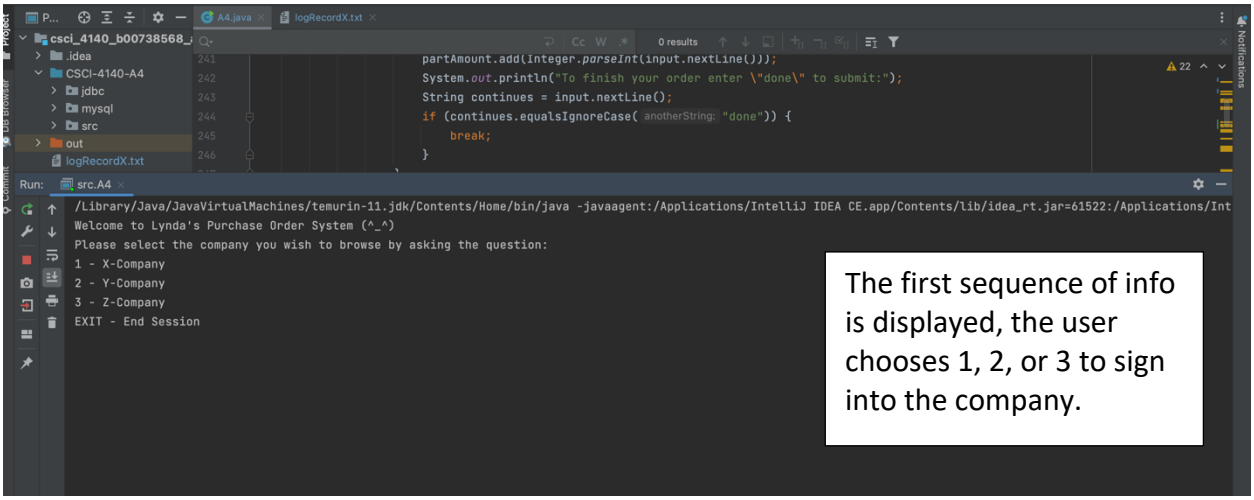
Working Software Screenshots

Input is highlighted in green in the console with the number of the action and response that is inputted as a string to console. Some information is outputted to the log record and /or database which is shown in screenshots.

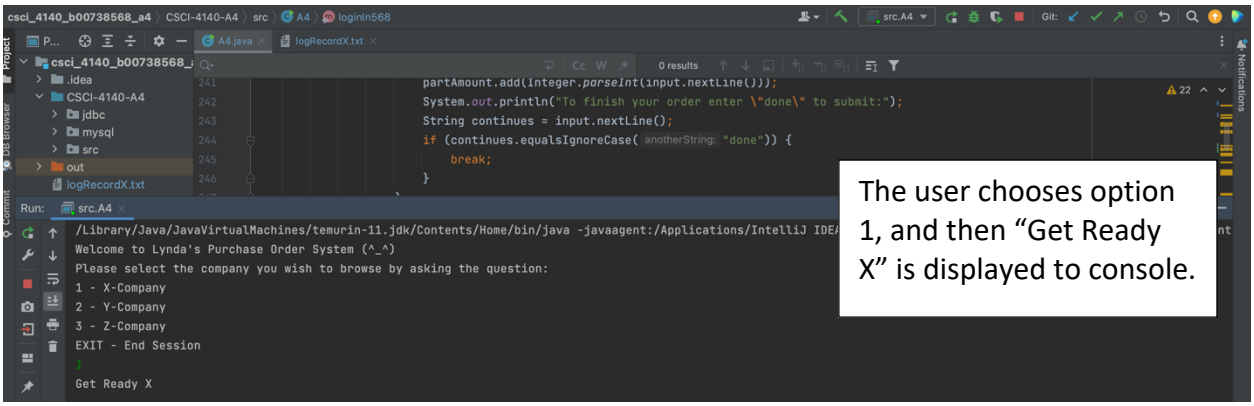
Company X Functionality



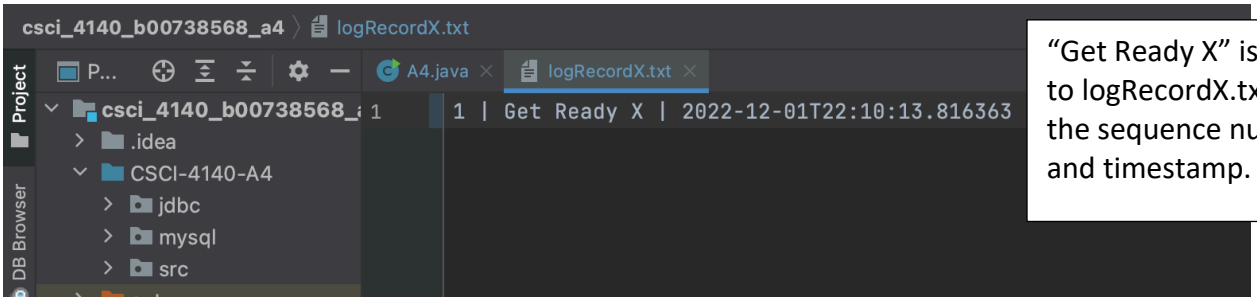
User starts program by clicking run 'src.A4'



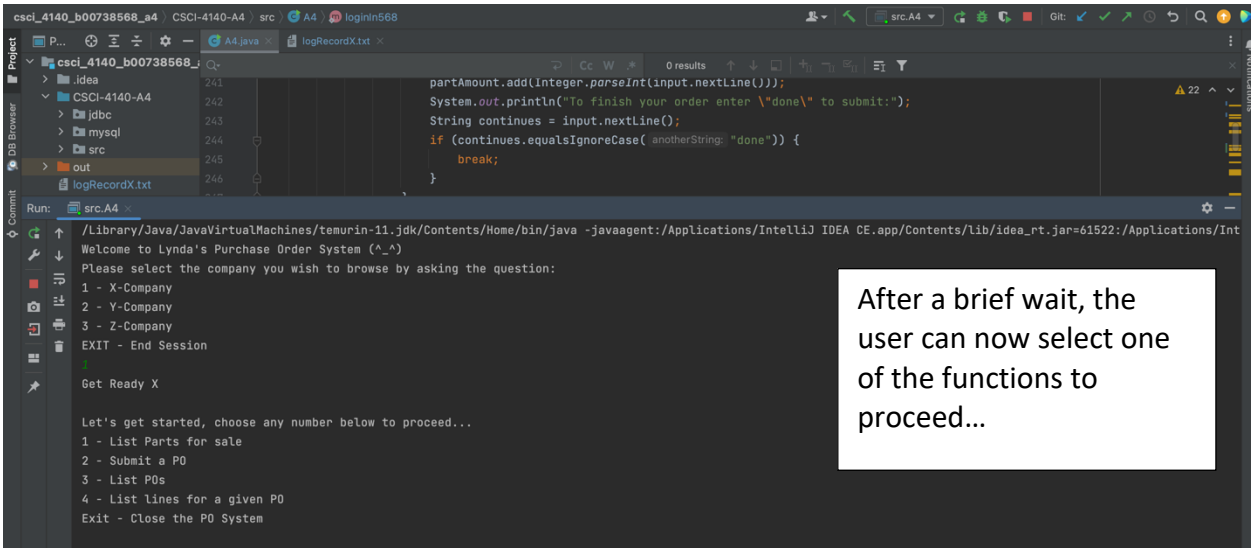
The first sequence of info is displayed, the user chooses 1, 2, or 3 to sign into the company.



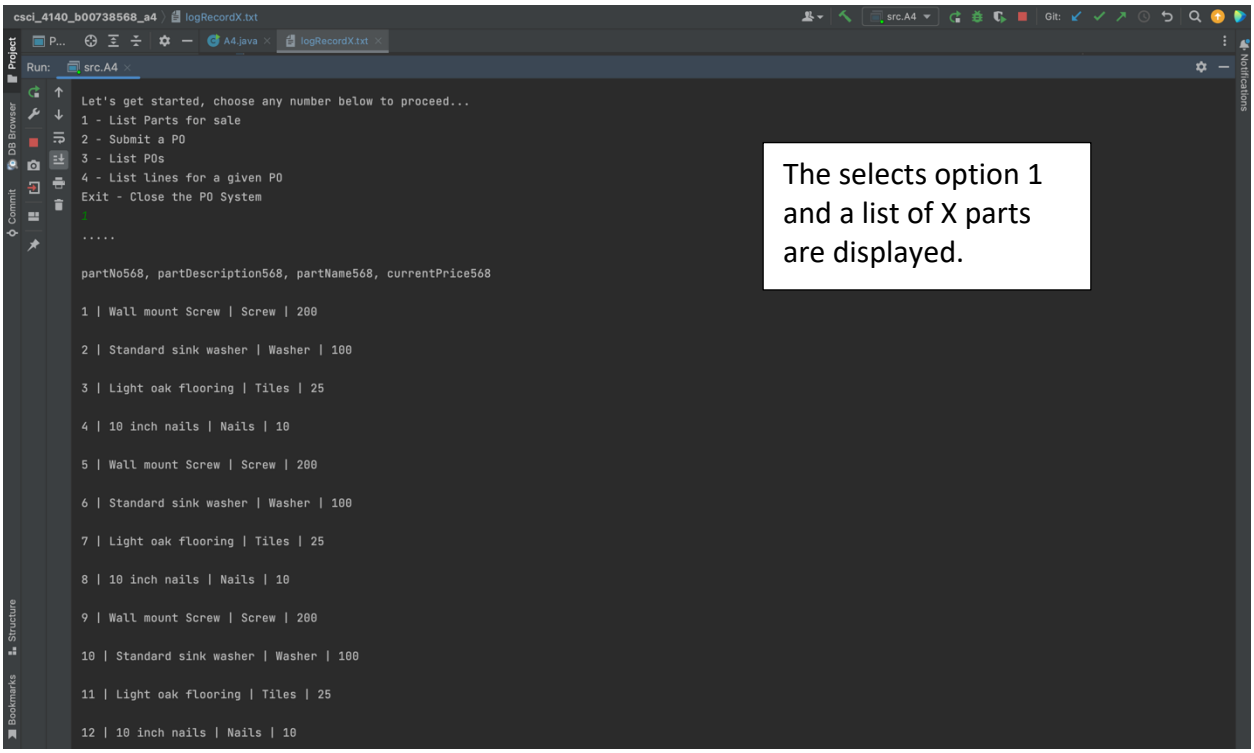
The user chooses option 1, and then "Get Ready X" is displayed to console.



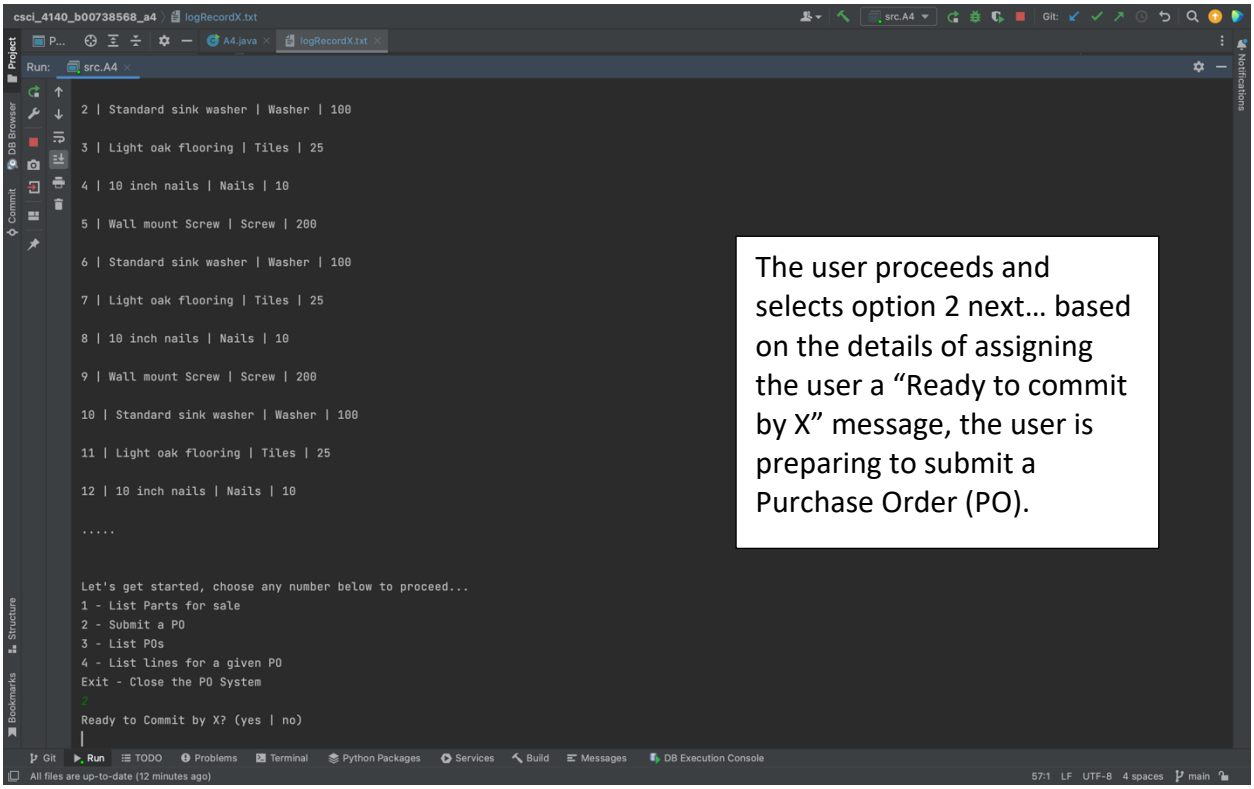
"Get Ready X" is logged to logRecordX.txt with the sequence number and timestamp.



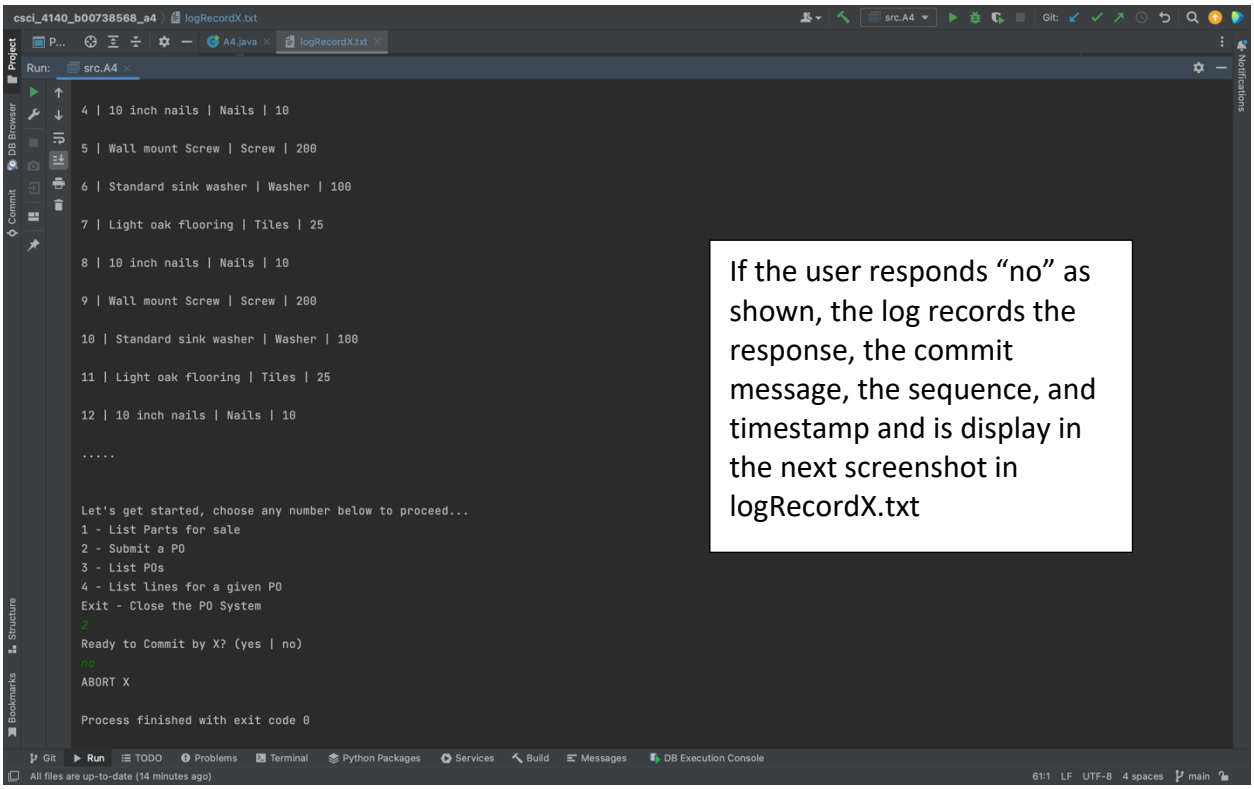
After a brief wait, the user can now select one of the functions to proceed…



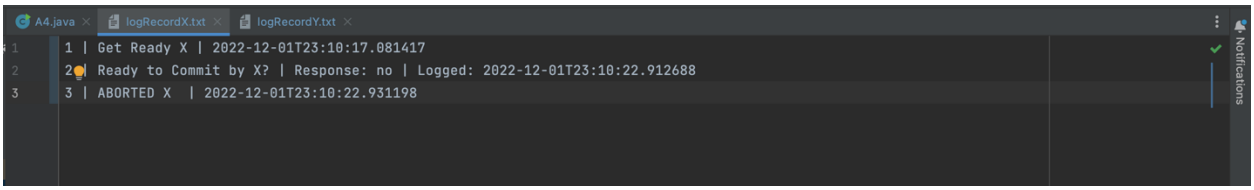
The selects option 1 and a list of X parts are displayed.

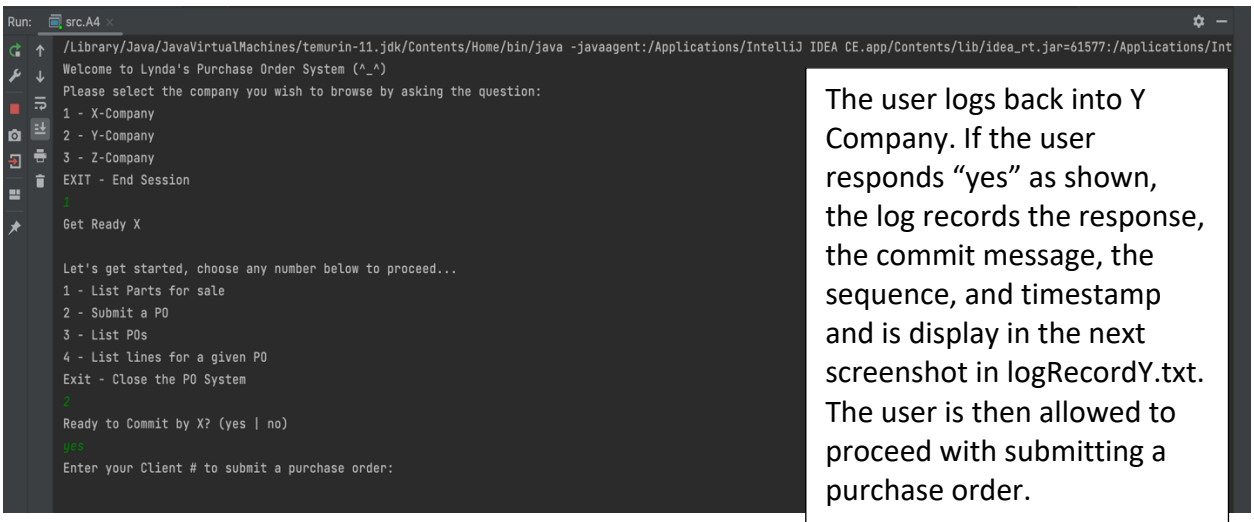


The user proceeds and selects option 2 next… based on the details of assigning the user a “Ready to commit by X” message, the user is preparing to submit a Purchase Order (PO).

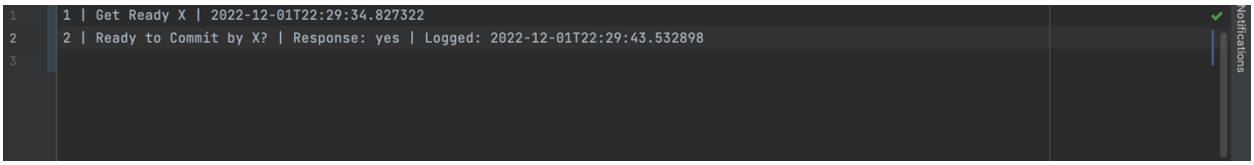


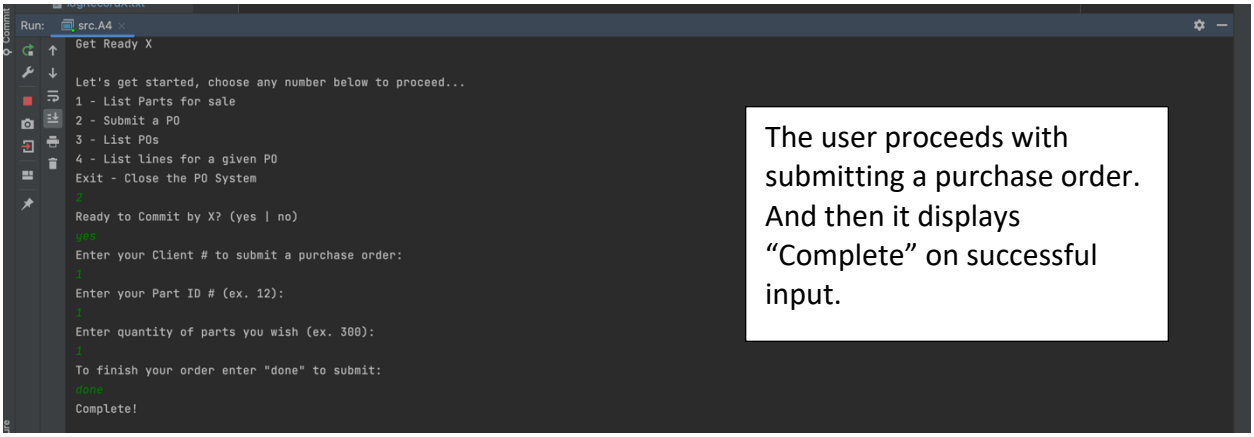
If the user responds “no” as shown, the log records the response, the commit message, the sequence, and timestamp and is display in the next screenshot in logRecordX.txt



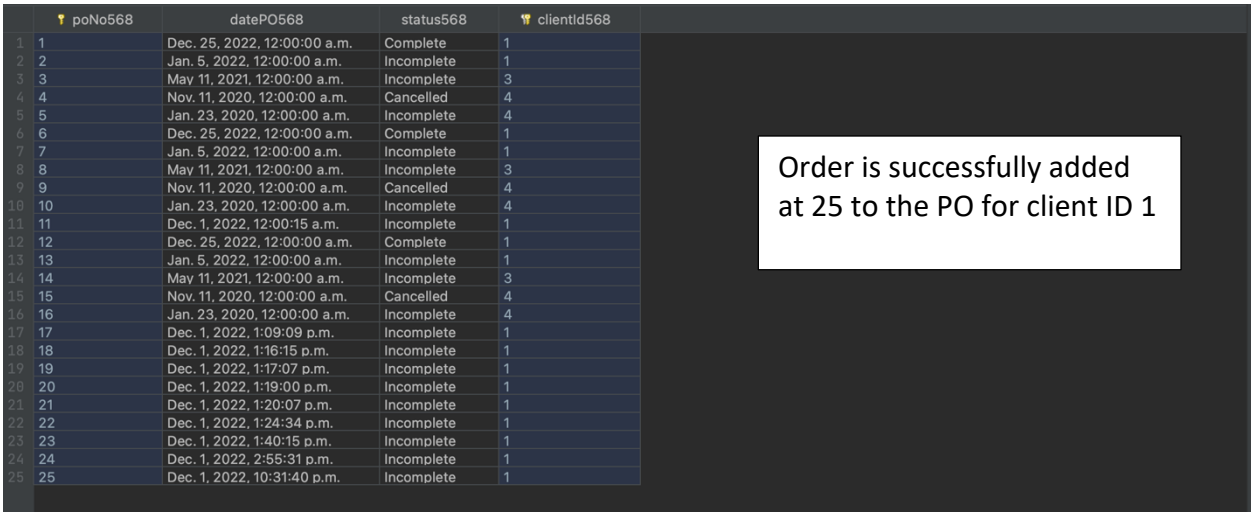


The user logs back into Y Company. If the user responds “yes” as shown, the log records the response, the commit message, the sequence, and timestamp and is display in the next screenshot in logRecordY.txt. The user is then allowed to proceed with submitting a purchase order.



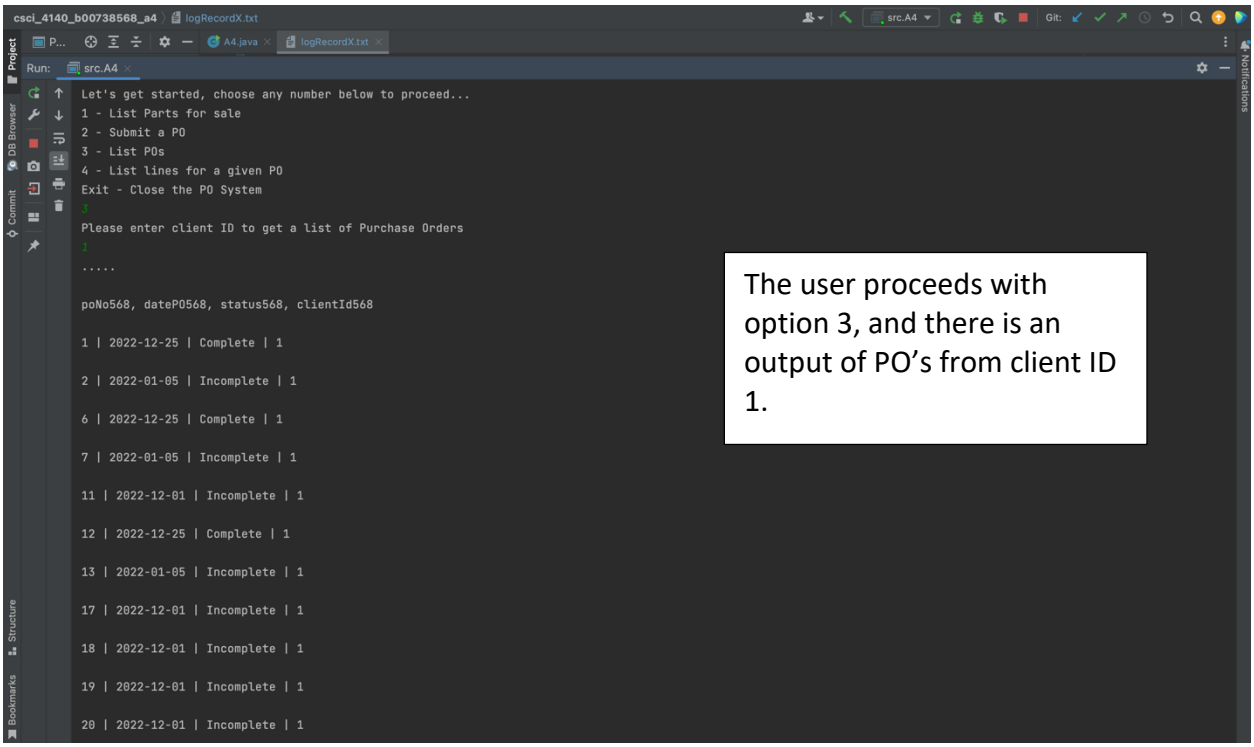


The user proceeds with submitting a purchase order. And then it displays “Complete” on successful input.

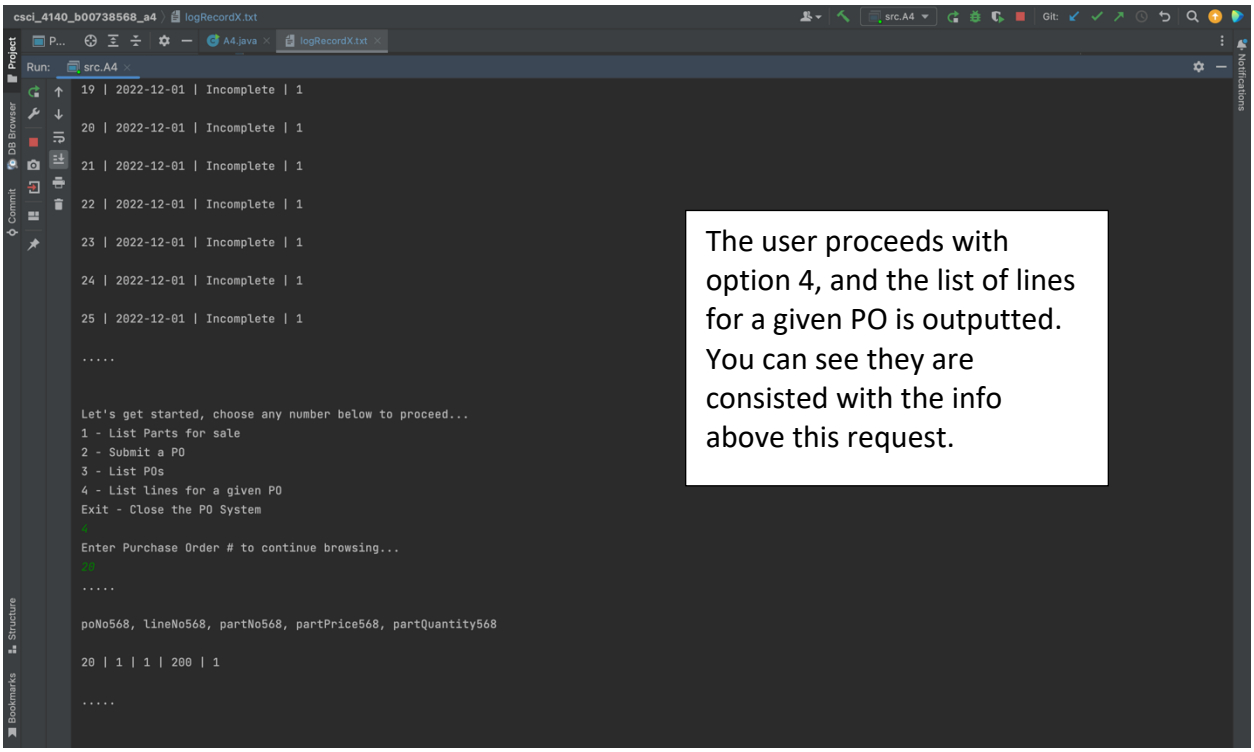


	poNo568	datePO568	status568	clientId568
1	1	Dec. 25, 2022, 12:00:00 a.m.	Complete	1
2	2	Jan. 5, 2022, 12:00:00 a.m.	Incomplete	1
3	3	May 11, 2021, 12:00:00 a.m.	Incomplete	3
4	4	Nov. 11, 2020, 12:00:00 a.m.	Cancelled	4
5	5	Jan. 23, 2020, 12:00:00 a.m.	Incomplete	4
6	6	Dec. 25, 2022, 12:00:00 a.m.	Complete	1
7	7	Jan. 5, 2022, 12:00:00 a.m.	Incomplete	1
8	8	May 11, 2021, 12:00:00 a.m.	Incomplete	3
9	9	Nov. 11, 2020, 12:00:00 a.m.	Cancelled	4
10	10	Jan. 23, 2020, 12:00:00 a.m.	Incomplete	4
11	11	Dec. 1, 2022, 12:00:15 a.m.	Incomplete	1
12	12	Dec. 25, 2022, 12:00:00 a.m.	Complete	1
13	13	Jan. 5, 2022, 12:00:00 a.m.	Incomplete	1
14	14	May 11, 2021, 12:00:00 a.m.	Incomplete	3
15	15	Nov. 11, 2020, 12:00:00 a.m.	Cancelled	4
16	16	Jan. 23, 2020, 12:00:00 a.m.	Incomplete	4
17	17	Dec. 1, 2022, 1:09:09 p.m.	Incomplete	1
18	18	Dec. 1, 2022, 1:16:15 p.m.	Incomplete	1
19	19	Dec. 1, 2022, 1:17:07 p.m.	Incomplete	1
20	20	Dec. 1, 2022, 1:19:00 p.m.	Incomplete	1
21	21	Dec. 1, 2022, 1:20:07 p.m.	Incomplete	1
22	22	Dec. 1, 2022, 1:24:34 p.m.	Incomplete	1
23	23	Dec. 1, 2022, 1:40:15 p.m.	Incomplete	1
24	24	Dec. 1, 2022, 2:55:31 p.m.	Incomplete	1
25	25	Dec. 1, 2022, 10:31:40 p.m.	Incomplete	1

Order is successfully added at 25 to the PO for client ID 1



The user proceeds with option 3, and there is an output of PO's from client ID 1.

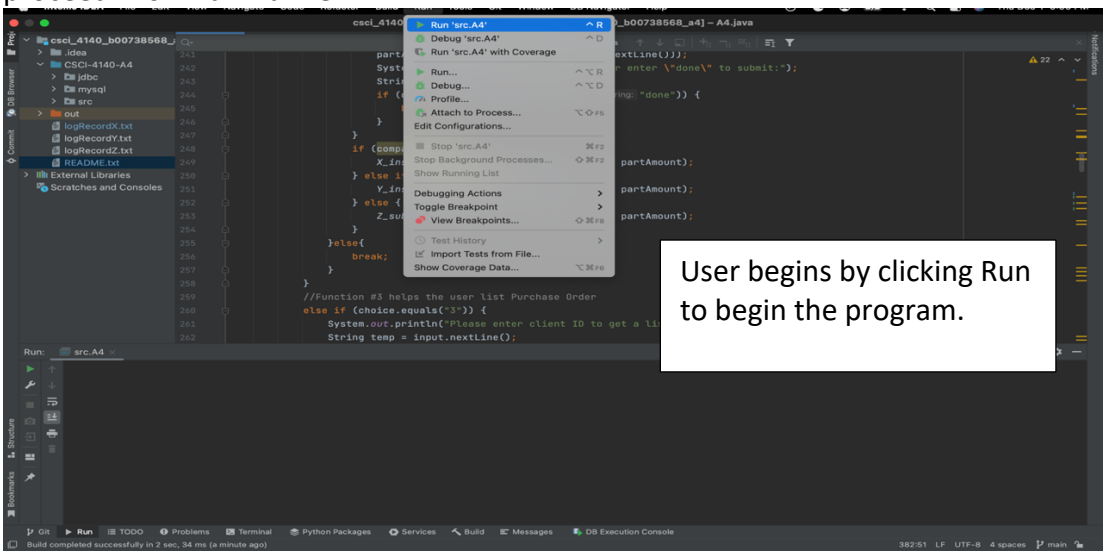


The user proceeds with option 4, and the list of lines for a given PO is outputted. You can see they are consisted with the info above this request.

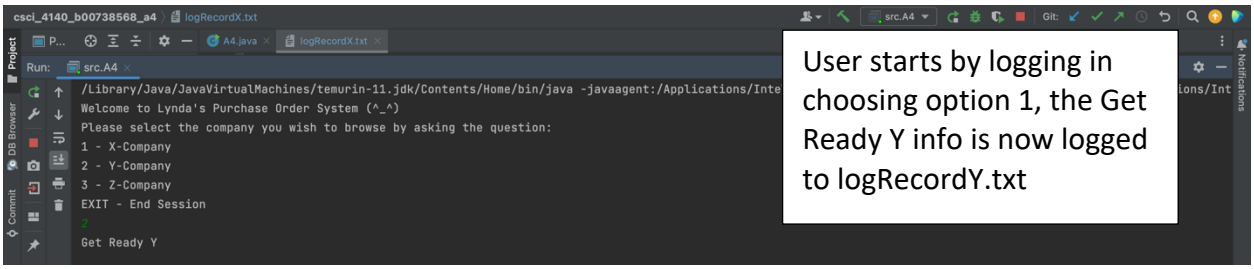
The user proceeds with option Exit, a message is displayed, and the user has finished with Company X.

Company Y Functionality

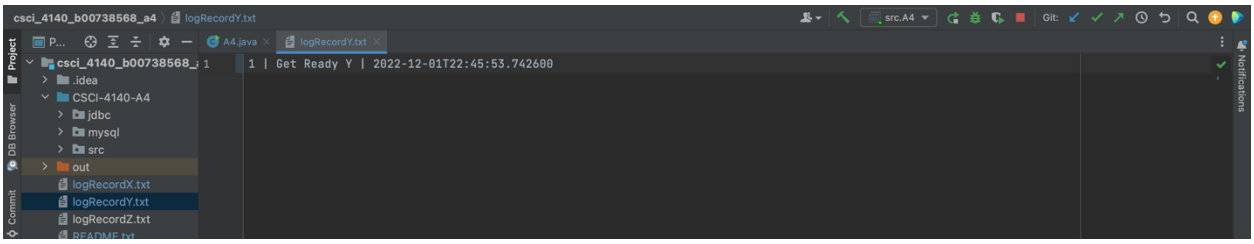
The information in this functionality is replicated from Company X functionality and will proceed in similar manner.

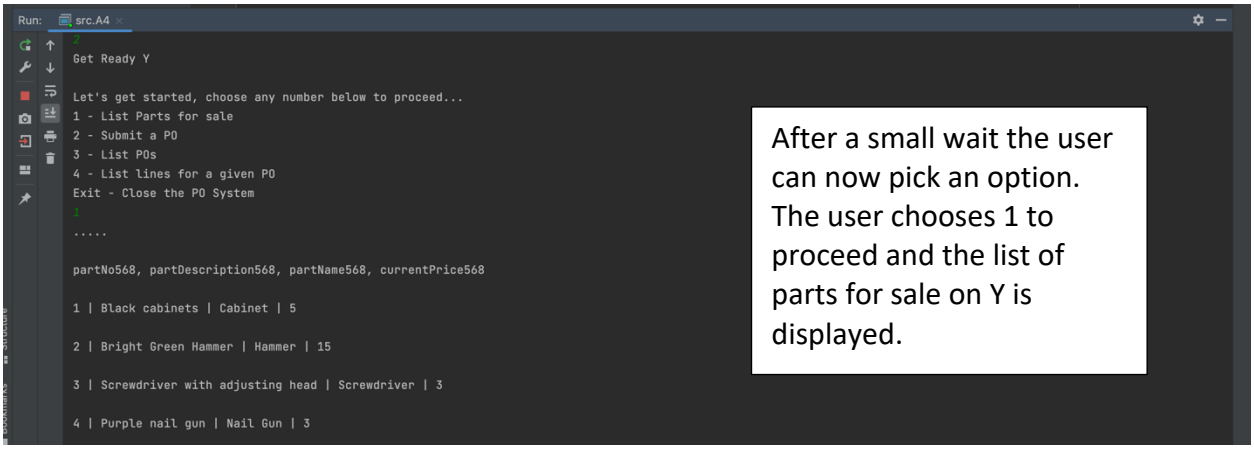


User begins by clicking Run to begin the program.

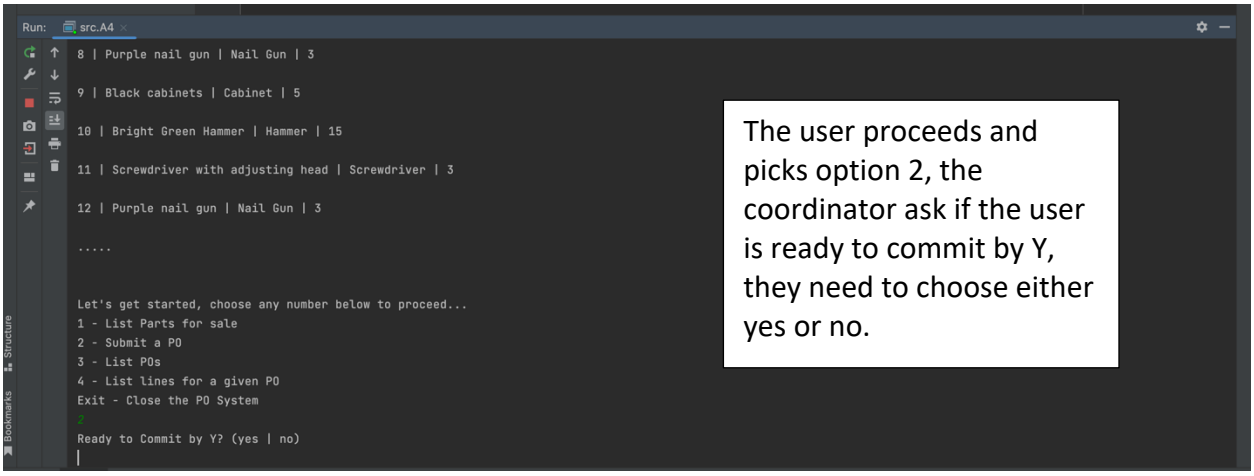


User starts by logging in choosing option 1, the Get Ready Y info is now logged to logRecordY.txt

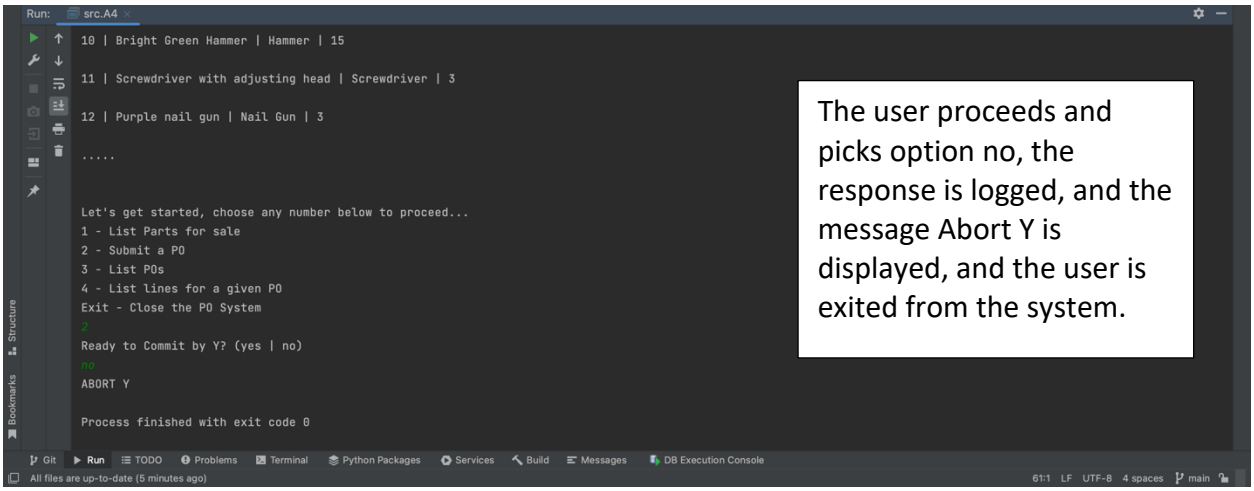




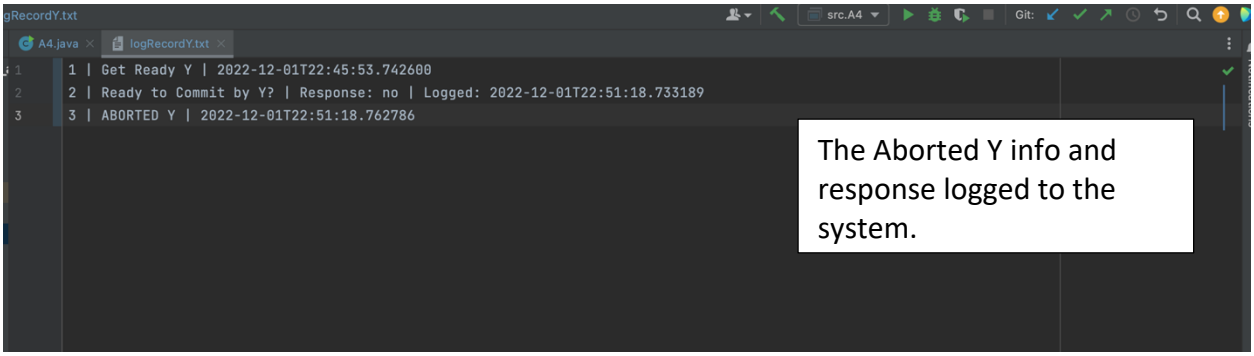
After a small wait the user can now pick an option. The user chooses 1 to proceed and the list of parts for sale on Y is displayed.



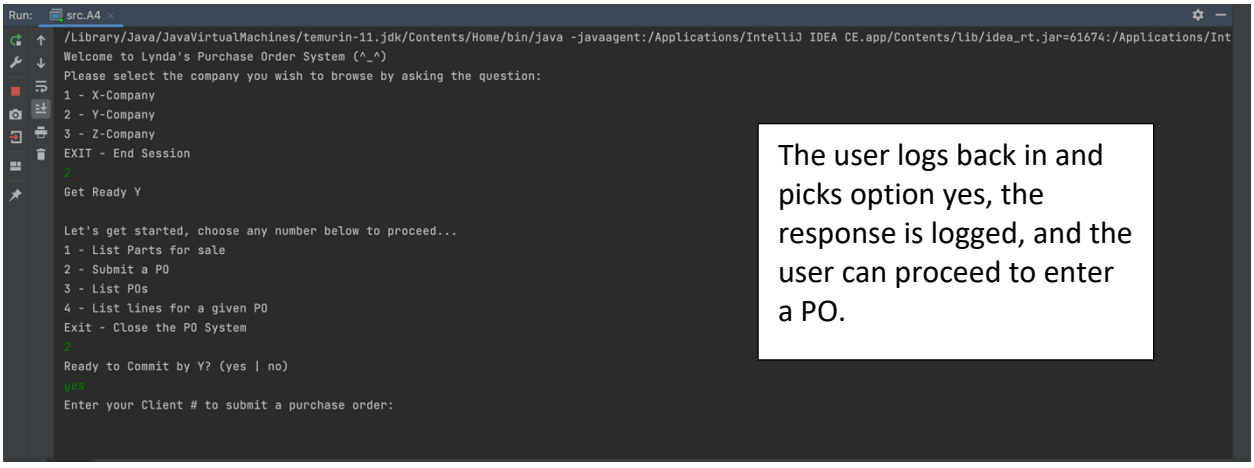
The user proceeds and picks option 2, the coordinator ask if the user is ready to commit by Y, they need to choose either yes or no.



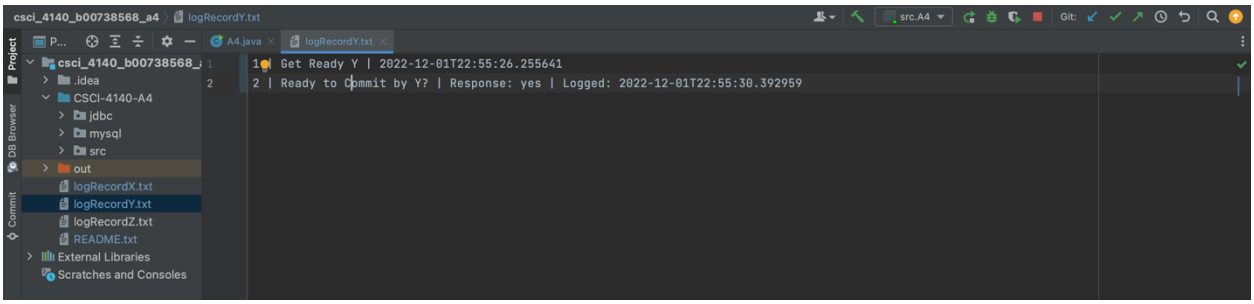
The user proceeds and picks option no, the response is logged, and the message Abort Y is displayed, and the user is exited from the system.



The Aborted Y info and response logged to the system.

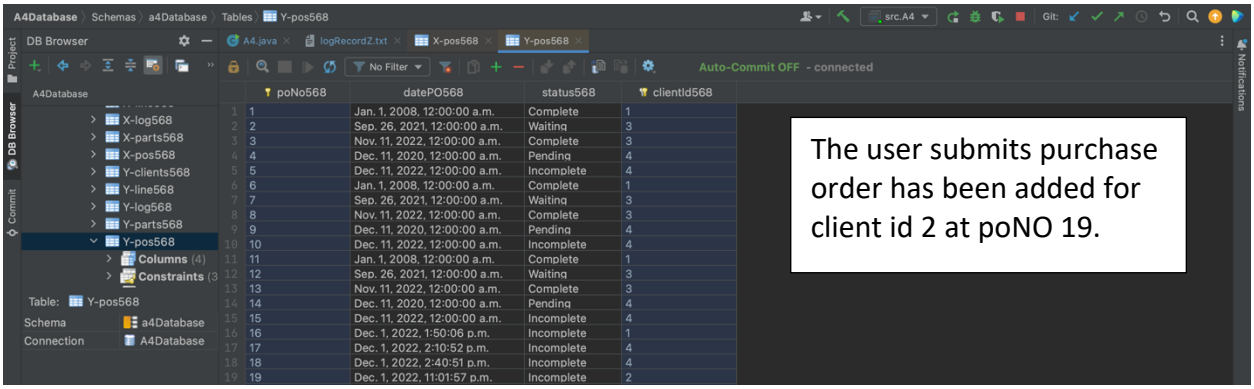


The user logs back in and picks option yes, the response is logged, and the user can proceed to enter a PO.



```
Let's get started, choose any number below to proceed...
1 - List Parts for sale
2 - Submit a PO
3 - List POs
4 - List lines for a given PO
Exit - Close the PO System
2
Ready to Commit by Y? (yes | no)
yes
Enter your Client # to submit a purchase order:
2
Enter your Part ID # (ex. 12):
2
Enter quantity of parts you wish (ex. 300):
3
To finish your order enter "done" to submit:
done
Complete!

Let's get started, choose any number below to proceed...
1 - List Parts for sale
2 - Submit a PO
3 - List POs
4 - List lines for a given PO
Exit - Close the PO System
```

The user submits purchase order and then the success message displays Complete!



The user submits purchase order has been added for client id 2 at poNO 19.

```
Let's get started, choose any number below to proceed...
1 - List Parts for sale
2 - Submit a PO
3 - List POs
4 - List lines for a given PO
Exit - Close the PO System
3
Please enter client ID to get a list of Purchase Orders
2
.....

poNo568, datePO568, status568, clientId568

19 | 2022-12-01 | Incomplete | 2

.....
```

User picks option 3 and then types ClientID and then the information is displayed

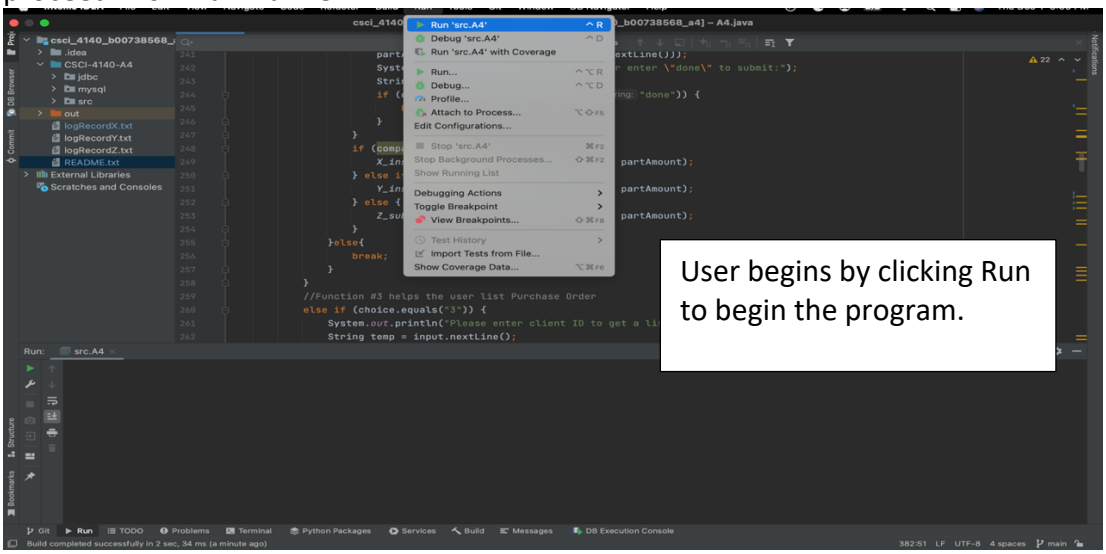
```
Let's get started, choose any number below to proceed...
1 - List Parts for sale
2 - Submit a PO
3 - List POs
4 - List lines for a given PO
Exit - Close the PO System
4
Enter Purchase Order # to continue browsing...
19
.....

poNo568, lineNo568, partNo568, partPrice568, partQuantity568

19 | 1 | 2 | 15 | 3

.....
```

User picks option 4 and then types purchase number and then the information is displayed. (Similar to the order we just submitted.

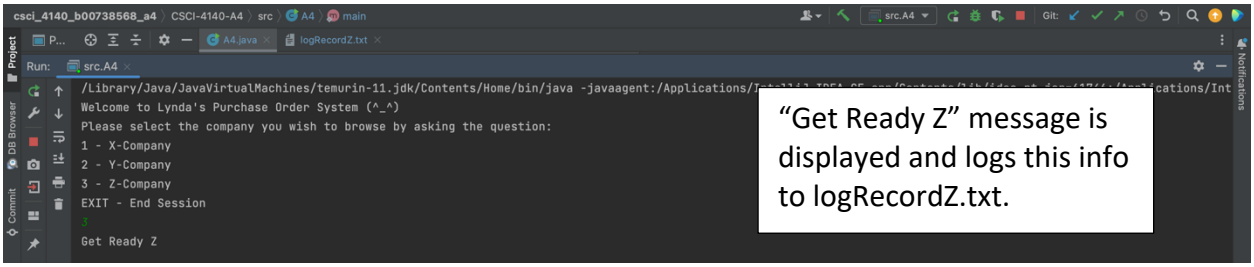
User picks option Exit, and is exited from the console.

Company Z Functionality

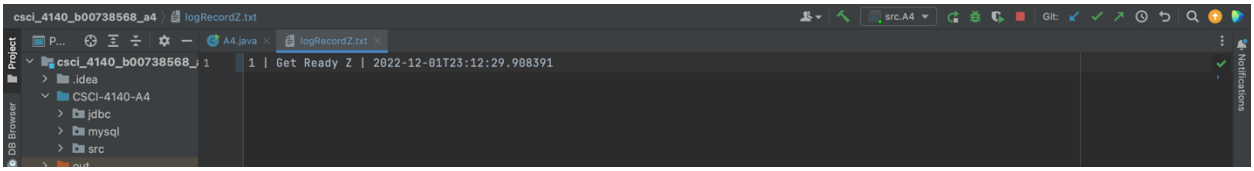
The information in this functionality is replicated from Company X and Y functionality and will proceed in similar manner.

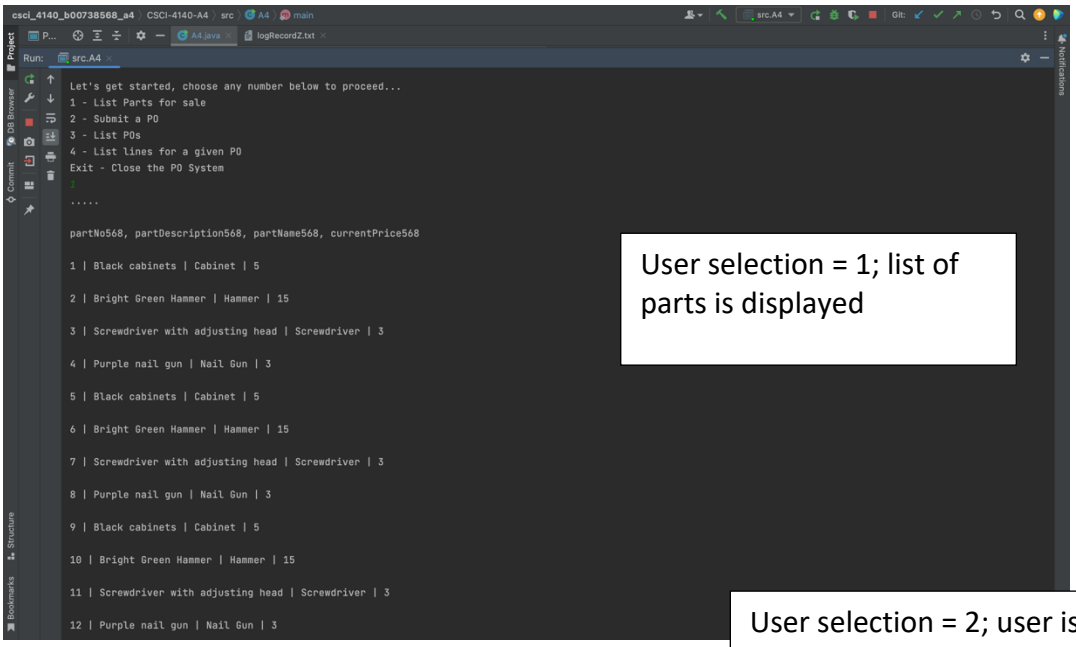


User begins by clicking Run to begin the program.



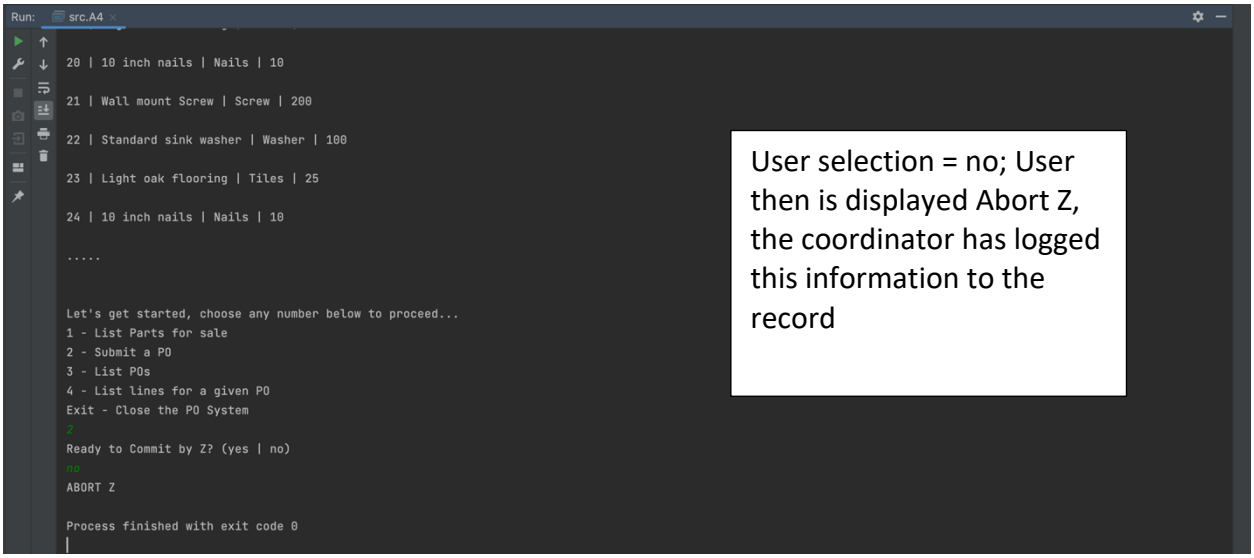
"Get Ready Z" message is displayed and logs this info to logRecordZ.txt.



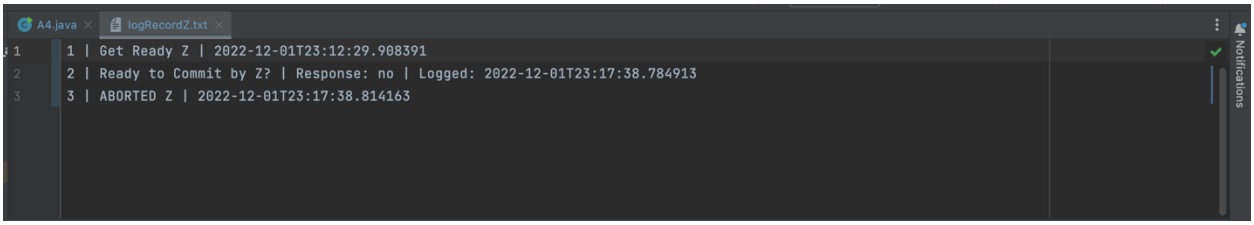


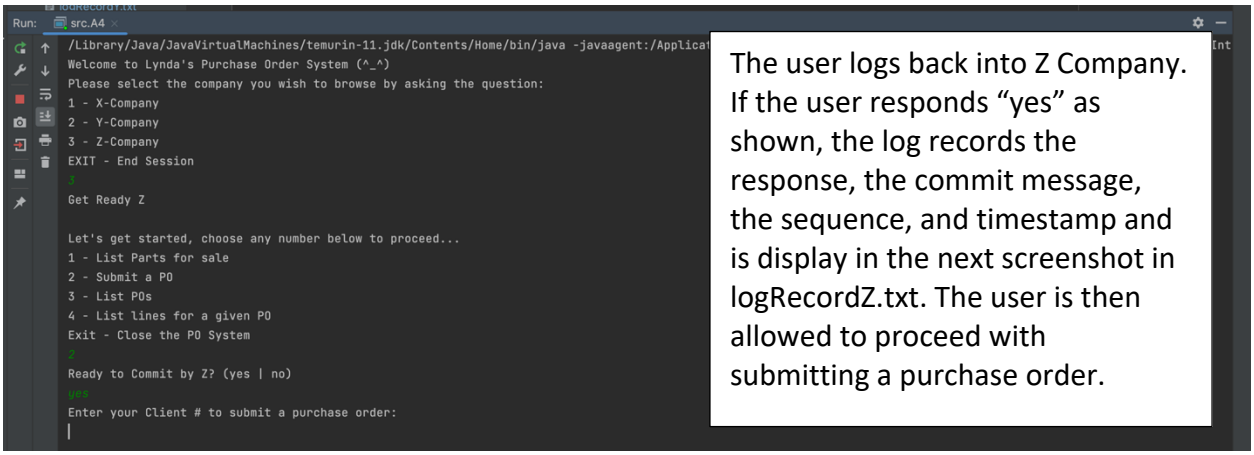
User selection = 1; list of parts is displayed

User selection = 2; user is asked if they are ready to commit by Z they must choose either yes or no to then have information logged to record.

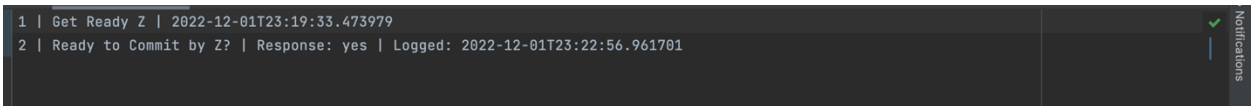


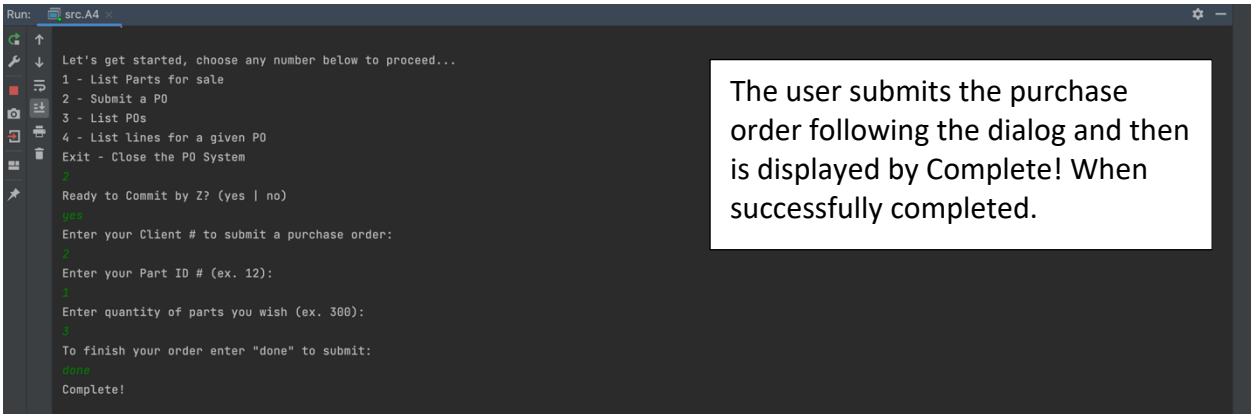
User selection = no; User then is displayed Abort Z, the coordinator has logged this information to the record



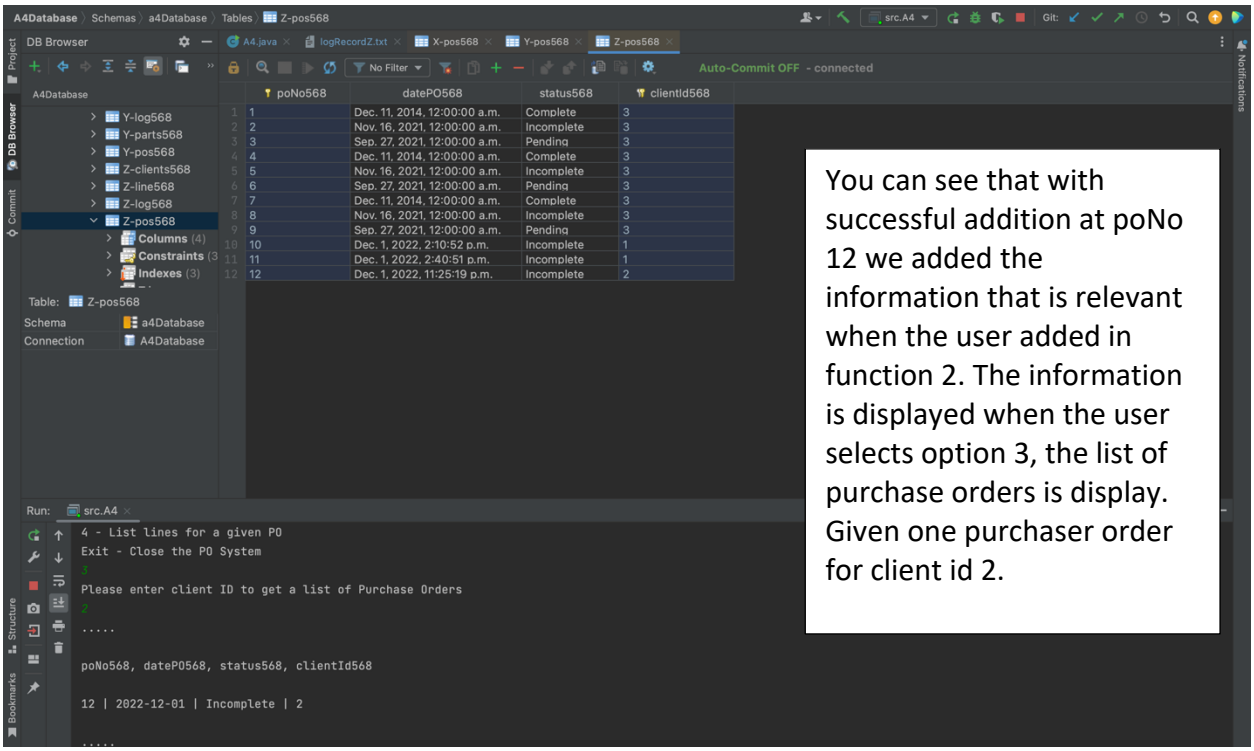


The user logs back into Z Company. If the user responds “yes” as shown, the log records the response, the commit message, the sequence, and timestamp and is display in the next screenshot in logRecordZ.txt. The user is then allowed to proceed with submitting a purchase order.

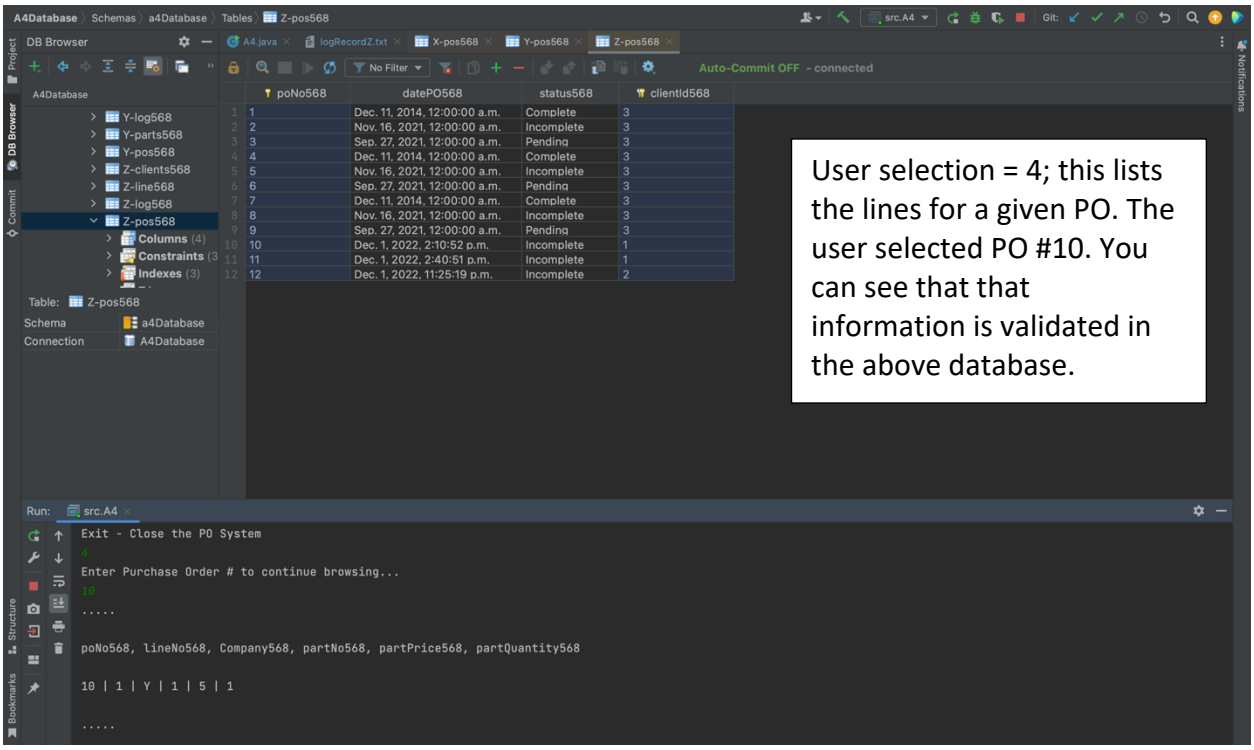




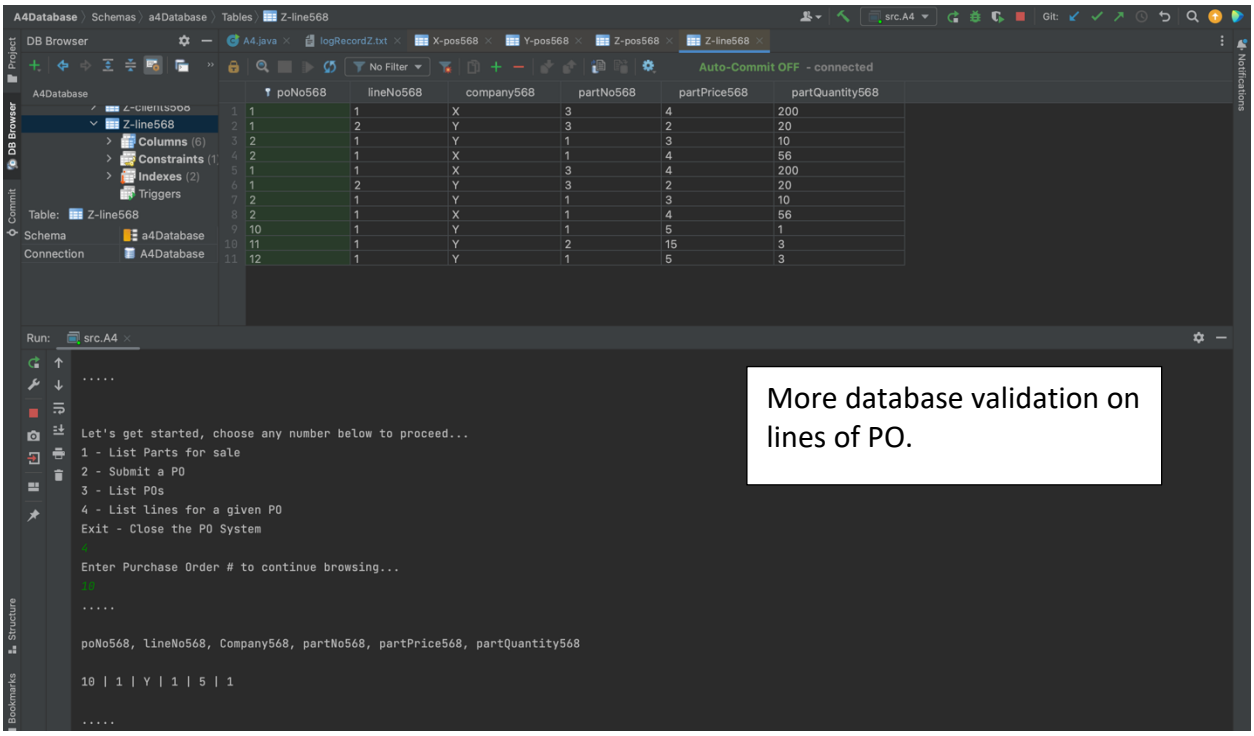
The user submits the purchase order following the dialog and then is displayed by Complete! When successfully completed.

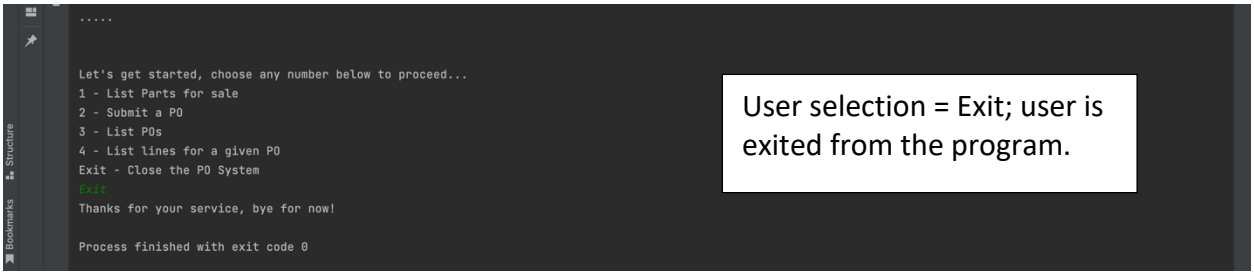


You can see that with successful addition at poNo 12 we added the information that is relevant when the user added in function 2. The information is displayed when the user selects option 3, the list of purchase orders is display. Given one purchaser order for client id 2.



User selection = 4; this lists the lines for a given PO. The user selected PO #10. You can see that that information is validated in the above database.





SQL Script (with insert statements)

File also visible through GitLab link. `csci_4140_b00738568_a4 > CSCI-4140-A4 > mysql > A4Database.sql`

-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,

FOREIGN_KEY_CHECKS=0;

```
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE
,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';
CREATE SCHEMA IF NOT EXISTS `a4Database` DEFAULT CHARACTER SET utf8mb4
COLLATE utf8mb4_0900_ai_ci ;
USE `a4Database` ;
-- -----------------------------------------------------
-- Table `a4Database`.`X-clients568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`X-clients568` (
  `clientId568` INT NOT NULL AUTO_INCREMENT,
  `clientCity568` VARCHAR(45) NULL DEFAULT NULL,
  `dollarsOnOrder568` INT NULL DEFAULT NULL,
  `clientStatus568` VARCHAR(45) NULL DEFAULT NULL,
  `clientName568` VARCHAR(45) NULL DEFAULT NULL,
  `clientCompPassword568` VARCHAR(45) NULL DEFAULT NULL,
  `moneyOwed568` INT NULL DEFAULT NULL,
  PRIMARY KEY (`clientId568`),
  UNIQUE INDEX `clientId_UNIQUE` (`clientId568` ASC) VISIBLE)
ENGINE = InnoDB
AUTO_INCREMENT = 1
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
```

```
-- -----------------------------------------------------
-- Table `a4Database`.`X-parts568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`X-parts568` (
  `partNo568` INT NOT NULL AUTO_INCREMENT,
  `partDescription568` VARCHAR(45) NULL DEFAULT NULL,
  `QoH568` VARCHAR(45) NULL DEFAULT NULL,
  `partName568` VARCHAR(45) NULL DEFAULT NULL,
  `currentPrice568` INT NULL DEFAULT NULL,
  PRIMARY KEY (`partNo568`),
  UNIQUE INDEX `partNo_UNIQUE` (`partNo568` ASC) VISIBLE)
ENGINE = InnoDB
AUTO_INCREMENT = 1
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
-- -----------------------------------------------------
-- Table `a4Database`.`X-pos568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`X-pos568` (
  `poNo568` INT NOT NULL AUTO_INCREMENT,
  `datePO568` DATETIME NULL DEFAULT NULL,
  `status568` VARCHAR(45) NULL DEFAULT NULL,
  `clientId568` INT NOT NULL,
```

```
  PRIMARY KEY (`poNo568`, `clientId568`),
  UNIQUE INDEX `poNo_UNIQUE` (`poNo568` ASC) VISIBLE,
  INDEX `fk_POs568_Clients568_idx` (`clientId568` ASC) VISIBLE,
  CONSTRAINT `fk_POs568_Clients568`
    FOREIGN KEY (`clientId568`)
    REFERENCES `a4Database`.`X-clients568` (`clientId568`))
ENGINE = InnoDB
AUTO_INCREMENT = 1
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
-- -----------------------------------------------------
-- Table `a4Database`.`X-line568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`X-line568` (
  `poNo568` INT NOT NULL,
  `lineNo568` INT NOT NULL,
  `partNo568` INT NOT NULL,
  `partPrice568` INT NOT NULL,
  `partQuantity568` INT NOT NULL,
  INDEX `fk_Line568_POs5681_idx` (`poNo568` ASC, `lineNo568` ASC) VISIBLE,
  INDEX `fk_Line568_Parts5681_idx` (`partNo568` ASC) VISIBLE,
  CONSTRAINT `fk_Line568_Parts5681`
    FOREIGN KEY (`partNo568`)
```

```
    REFERENCES `a4Database`.`X-parts568` (`partNo568`),
  CONSTRAINT `fk_Line568_POs5681`
    FOREIGN KEY (`poNo568`)
    REFERENCES `a4Database`.`X-pos568` (`poNo568`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;



-- -----------------------------------------------------------------------------------------------------
-- -----------------------------------------------------
-- Table `a4Database`.`Y-clients568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`Y-clients568` (
  `clientId568` INT NOT NULL AUTO_INCREMENT,
  `clientCity568` VARCHAR(45) NULL DEFAULT NULL,
  `dollarsOnOrder568` INT NULL DEFAULT NULL,
  `clientStatus568` VARCHAR(45) NULL DEFAULT NULL,
  `clientName568` VARCHAR(45) NULL DEFAULT NULL,
  `clientCompPassword568` VARCHAR(45) NULL DEFAULT NULL,
  `moneyOwed568` INT NULL DEFAULT NULL,
  PRIMARY KEY (`clientId568`),
  UNIQUE INDEX `clientId_UNIQUE` (`clientId568` ASC) VISIBLE)
```

```
ENGINE = InnoDB
AUTO_INCREMENT = 1
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
-- -----------------------------------------------------
-- Table `a4Database`.`Y-parts568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`Y-parts568` (
  `partNo568` INT NOT NULL AUTO_INCREMENT,
  `partDescription568` VARCHAR(45) NULL DEFAULT NULL,
  `QoH568` VARCHAR(45) NULL DEFAULT NULL,
  `partName568` VARCHAR(45) NULL DEFAULT NULL,
  `currentPrice568` INT NULL DEFAULT NULL,
  PRIMARY KEY (`partNo568`),
  UNIQUE INDEX `partNo_UNIQUE` (`partNo568` ASC) VISIBLE)
ENGINE = InnoDB
AUTO_INCREMENT = 1
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
-- -----------------------------------------------------
-- Table `a4Database`.`Y-pos568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`Y-pos568` (
```

```
  `poNo568` INT NOT NULL AUTO_INCREMENT,
  `datePO568` DATETIME NULL DEFAULT NULL,
  `status568` VARCHAR(45) NULL DEFAULT NULL,
  `clientId568` INT NOT NULL,
  PRIMARY KEY (`poNo568`, `clientId568`),
  UNIQUE INDEX `poNo_UNIQUE` (`poNo568` ASC) VISIBLE,
  INDEX `fk_Y-POs568_Clients568_idx` (`clientId568` ASC) VISIBLE,
  CONSTRAINT `fk_Y-POs568_Clients568`
    FOREIGN KEY (`clientId568`)
    REFERENCES `a4Database`.`Y-clients568` (`clientId568`))
ENGINE = InnoDB
AUTO_INCREMENT = 1
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
-- -----------------------------------------------------
-- Table `a4Database`.`Y-line568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`Y-line568` (
  `poNo568` INT NOT NULL,
  `lineNo568` INT NOT NULL,
  `partNo568` INT NOT NULL,
  `partPrice568` INT NOT NULL,
  `partQuantity568` INT NOT NULL,
```

```
  INDEX `fk_Y-Line568_POs5681_idx` (`poNo568` ASC, `lineNo568` ASC) VISIBLE,
  INDEX `fk_Y-Line568_Parts5681_idx` (`partNo568` ASC) VISIBLE,
  CONSTRAINT `fk_Y-Line568_Parts5681`
    FOREIGN KEY (`partNo568`)
    REFERENCES `a4Database`.`Y-parts568` (`partNo568`),
  CONSTRAINT `fk_Y-Line568_POs5681`
    FOREIGN KEY (`poNo568`)
    REFERENCES `a4Database`.`Y-pos568` (`poNo568`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-- -----------------------------------------------------------------------------------------------------------

-- -----------------------------------------------------
-- Table `a4Database`.`Z-clients568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`Z-clients568` (
  `clientId568` INT NOT NULL AUTO_INCREMENT,
  `clientCity568` VARCHAR(45) NULL DEFAULT NULL,
  `dollarsOnOrder568` INT NULL DEFAULT NULL,
  `clientStatus568` VARCHAR(45) NULL DEFAULT NULL,
  `clientName568` VARCHAR(45) NULL DEFAULT NULL,
  `clientCompPassword568` VARCHAR(45) NULL DEFAULT NULL,
  `moneyOwed568` INT NULL DEFAULT NULL,
```

```
  PRIMARY KEY (`clientId568`),
  UNIQUE INDEX `clientId_UNIQUE` (`clientId568` ASC) VISIBLE)
ENGINE = InnoDB
AUTO_INCREMENT = 1
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
-- -----------------------------------------------------
-- Table `a4Database`.`Z-pos568`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `a4Database`.`Z-pos568` (
  `poNo568` INT NOT NULL AUTO_INCREMENT,
  `datePO568` DATETIME NULL DEFAULT NULL,
  `status568` VARCHAR(45) NULL DEFAULT NULL,
  `clientId568` INT NOT NULL,
  PRIMARY KEY (`poNo568`, `clientId568`),
  UNIQUE INDEX `poNo_UNIQUE` (`poNo568` ASC) VISIBLE,
  INDEX `fk_Z-POs568_Clients568_idx` (`clientId568` ASC) VISIBLE,
  CONSTRAINT `fk_Z-POs568_Clients568`
    FOREIGN KEY (`clientId568`)
    REFERENCES `a4Database`.`Z-clients568` (`clientId568`))
ENGINE = InnoDB
AUTO_INCREMENT = 1
DEFAULT CHARACTER SET = utf8mb4
```

```
COLLATE = utf8mb4_0900_ai_ci;

-- -----------------------------------------------------

-- Table `a4Database`.`Z-line568`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `a4Database`.`Z-line568` (

  `poNo568` INT NOT NULL,

  `lineNo568` INT NOT NULL,

  `company568` VARCHAR(45) NOT NULL,

  `partNo568` INT NOT NULL,

  `partPrice568` INT NOT NULL,

  `partQuantity568` INT NOT NULL,

  INDEX `fk_Z-Line568_POs5681_idx` (`poNo568` ASC, `lineNo568` ASC) VISIBLE,

  INDEX `fk_Z-Line568_Parts5681_idx` (`partNo568` ASC) VISIBLE,

  CONSTRAINT `fk_Z-Line568_POs5681`

    FOREIGN KEY (`poNo568`)

    REFERENCES `a4Database`.`Z-pos568` (`poNo568`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8mb4

COLLATE = utf8mb4_0900_ai_ci;

SET SQL_MODE=@OLD_SQL_MODE;

SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;

SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

-- -----------------------------------------------------
```

```
-- Creating X-Parts for Company X
-- -----------------------------------------------------
INSERT INTO `x-parts568` (partDescription568, QoH568, partName568, currentPrice568)
VALUES ('Wall mount Screw', 200, 'Screw', 200);
INSERT INTO `x-parts568` (partDescription568, QoH568, partName568, currentPrice568)
VALUES ('Standard sink washer', 100, 'Washer', 100);
INSERT INTO `x-parts568` (partDescription568, QoH568, partName568, currentPrice568)
VALUES ('Light oak flooring', 1400, 'Tiles', 25);
INSERT INTO `x-parts568` (partDescription568, QoH568, partName568, currentPrice568)
VALUES ('10 inch nails', 135, 'Nails', 10);
-- -----------------------------------------------------
-- Creating X-Clients for Company X
-- -----------------------------------------------------
INSERT INTO `x-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Halifax', 100, 'Active', 'Dan', 'Dan123', 2000);
INSERT INTO `x-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Victoria', 0, 'Inactive', 'Bobby', 'sugar23!', 350);
INSERT INTO `x-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Tennessee', 15, 'Active', 'Lynda', 'pASWod1003', 0);
```

```
INSERT INTO `x-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Toronto', 15, 'Active', 'Melissa', 'happyDays', 124);
-- -----------------------------------------------------
-- Creating X-Pos for Purchase order of Company X
-- -----------------------------------------------------
INSERT INTO `x-pos568` (datePO568, status568, clientId568)
VALUES ('2022-12-25', 'Complete', 1);

INSERT INTO `x-pos568` (datePO568, status568, clientId568)
VALUES ('2022-01-05', 'Incomplete', 1);
INSERT INTO `x-pos568` (datePO568, status568, clientId568)
VALUES ('2021-05-11', 'Incomplete', 3);
INSERT INTO `x-pos568` (datePO568, status568, clientId568)
VALUES ('2020-11-11', 'Cancelled', 4);
INSERT INTO `x-pos568` (datePO568, status568, clientId568)
VALUES ('2020-01-23', 'Incomplete', 4);
-- -----------------------------------------------------
-- Creating X-Lines for Company X
-- -----------------------------------------------------
INSERT INTO `x-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (1, 1, 1, 10, 2);
INSERT INTO `x-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
```

```
VALUES (1, 2, 2, 24, 3);
INSERT INTO `x-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (2, 1, 3, 3, 4);
INSERT INTO `x-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (3, 1, 2, 25, 5);
INSERT INTO `x-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (4, 1, 3, 4, 200);
INSERT INTO `x-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (5, 1, 3, 6, 200);
-- -----------------------------------------------------
-- Creating y-Parts for Company Y
-- -----------------------------------------------------
INSERT INTO `y-parts568` (partDescription568, QoH568, partName568, currentPrice568)
VALUES ('Black cabinets', 12, 'Cabinet', 5);
INSERT INTO `y-parts568` (partDescription568, QoH568, partName568, currentPrice568)
VALUES ('Bright Green Hammer', 150, 'Hammer', 15);
INSERT INTO `y-parts568` (partDescription568, QoH568, partName568, currentPrice568)
VALUES ('Screwdriver with adjusting head', 45, 'Screwdriver', 3);
INSERT INTO `y-parts568` (partDescription568, QoH568, partName568, currentPrice568)
VALUES ('Purple nail gun', 50, 'Nail Gun', 3);
-- -----------------------------------------------------
-- Creating y-Clients for Company Y
-- -----------------------------------------------------
```

```
INSERT INTO `y-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Vancouver', 180, 'Active', 'Michael', 'Micha3l123', 22000);
INSERT INTO `y-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Chicago', 0, 'Inactive', 'Taylor', 'Tgirly2012!', 0);
INSERT INTO `y-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Lexington', 90, 'Active', 'Liv', 'livvy2002', 3300);
INSERT INTO `y-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Fredericksburg', 0, 'Active', 'Company Z', '!128923!', 0);
-- -----------------------------------------------------
-- Creating y-Pos for Company Y
-- -----------------------------------------------------
INSERT INTO `y-pos568` (datePO568, status568, clientId568)
VALUES ('2008-01-01', 'Complete', 1);
INSERT INTO `y-pos568` (datePO568, status568, clientId568)
VALUES ('2021-09-26', 'Waiting', 3);
INSERT INTO `y-pos568` (datePO568, status568, clientId568)
VALUES ('2022-11-11', 'Complete', 3);
INSERT INTO `y-pos568` (datePO568, status568, clientId568)
VALUES ('2020-12-11', 'Pending', 4);
```

```
INSERT INTO `y-pos568` (datePO568, status568, clientId568)
VALUES ('2022-12-11', 'Incomplete', 4);
-- -----------------------------------------------------
-- Creating y-Lines for Company Y
-- -----------------------------------------------------
INSERT INTO `y-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (1, 1, 1, 5, 10);
INSERT INTO `y-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (1, 2, 2, 26, 10);
INSERT INTO `y-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (2, 1, 3, 3, 50);
INSERT INTO `y-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (3, 1, 2, 26, 1);
INSERT INTO `y-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (4, 1, 3, 3, 20);
INSERT INTO `y-line568` (poNo568, lineNo568, partNo568, partPrice568, partQuantity568)
VALUES (5, 1, 1, 5, 10);
-- -----------------------------------------------------
-- Creating z-Clients for Company Z
-- -----------------------------------------------------
INSERT INTO `z-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Wolfville', 0, 'Active', 'Neil', 'NeiltheThri!!', 20);
```

```
INSERT INTO `z-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Sydney', 0, 'InActive', 'Morgan', 'Morgan201223', 0);
INSERT INTO `z-clients568` (clientCity568, dollarsOnOrder568, clientStatus568,
clientName568, clientCompPassword568, moneyOwed568)
VALUES ('Bedford', 0, 'Active', 'Patrick', 'karson321', 1004);
-- -----------------------------------------------------
-- Creating Pos for Company Z
-- -----------------------------------------------------
INSERT INTO `z-pos568` (datePO568, status568, clientId568)
VALUES ('2014-12-11', 'Complete', 3);
INSERT INTO `z-pos568` (datePO568, status568, clientId568)
VALUES ('2021-11-16', 'Incomplete', 3);
INSERT INTO `z-pos568` (datePO568, status568, clientId568)
VALUES ('2021-09-27', 'Pending', 3);
-- -----------------------------------------------------
-- Creating Lines for Company Z
-- -----------------------------------------------------
INSERT INTO `z-line568` (poNo568, lineNo568, company568, partNo568, partPrice568,
partQuantity568)
VALUES (1, 1, "X", 3, 4, 200);
INSERT INTO `z-line568` (poNo568, lineNo568, company568, partNo568, partPrice568,
partQuantity568)
```

```
VALUES (1, 2, "Y", 3, 2, 20);
INSERT INTO `z-line568` (poNo568, lineNo568, company568, partNo568, partPrice568,
partQuantity568)
VALUES (2, 1, "Y", 1, 3, 10);
INSERT INTO `z-line568` (poNo568, lineNo568, company568, partNo568, partPrice568,
partQuantity568)
VALUES (2, 1, "X", 1, 4, 56);
```