

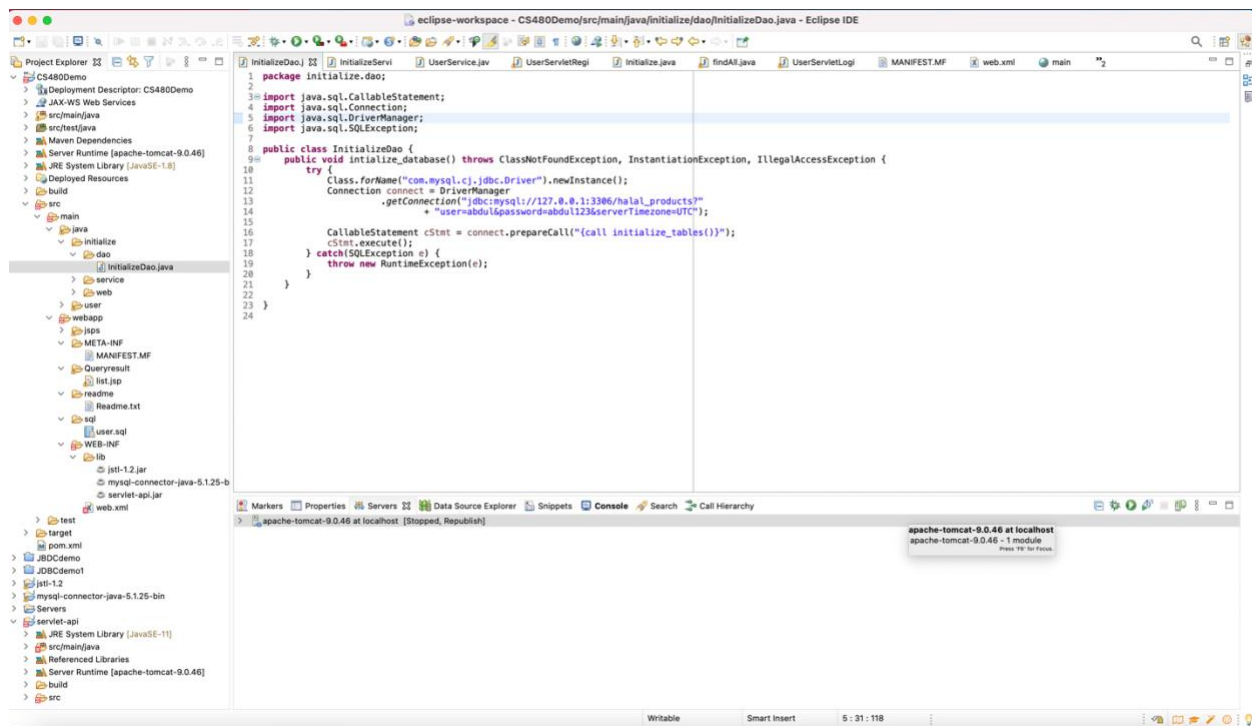
Steps to add a link for initializing database tables:

1. A new stored procedure was created which had all the scripts. The stored procedure name was “initialize_tables.sql”.

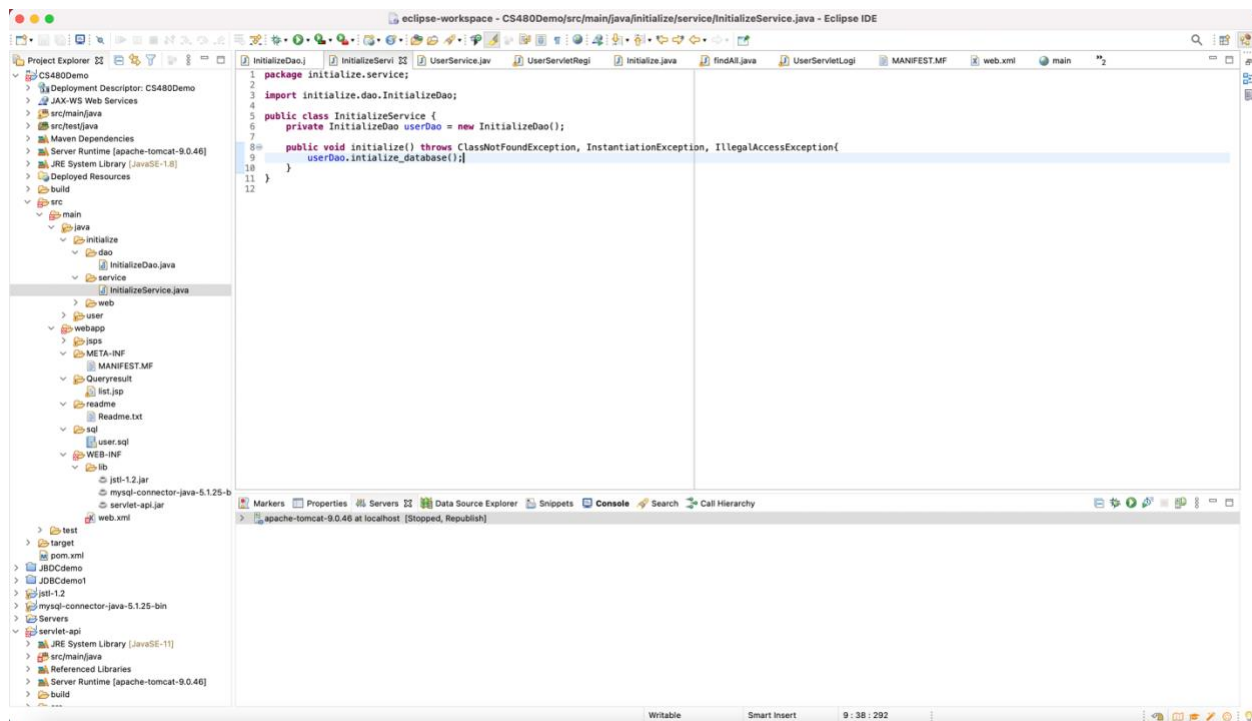
The screenshot displays the MySQL Modeler EER Diagram interface. The left sidebar shows a tree view of the database schema, including tables like 'halal_status', 'manufacturer', 'product', 'product_category', 'product_store', 'region', 'store', 'tb_user', and views like 'homework_1', 'homework_1a', 'sys', and 'test'. The main editor area shows the SQL code for the stored procedure 'initialize_tables'. The code includes comments for dropping and creating tables, and the actual SQL statements for creating 'halal_status', 'manufacturer', and 'product_category' tables. The bottom panel shows the 'Action Output' table with the following data:

	Time	Action	Response	Duration / Fetch Time
276	21:44:12	DROP TABLE 'halal_products', 'manufacturer'	0 row(s) affected	0.0044 sec
279	21:44:18	DROP TABLE 'halal_products', 'product_category'	0 row(s) affected	0.0043 sec
280	21:44:26	SELECT * FROM halal_products.tb_user LIMIT 0, 1000	1 row(s) returned	0.00094 sec / 0.000...
281	21:45:01	SELECT * FROM halal_products.halal_status LIMIT 0, 1000	3 row(s) returned	0.00034 sec / 0.000...

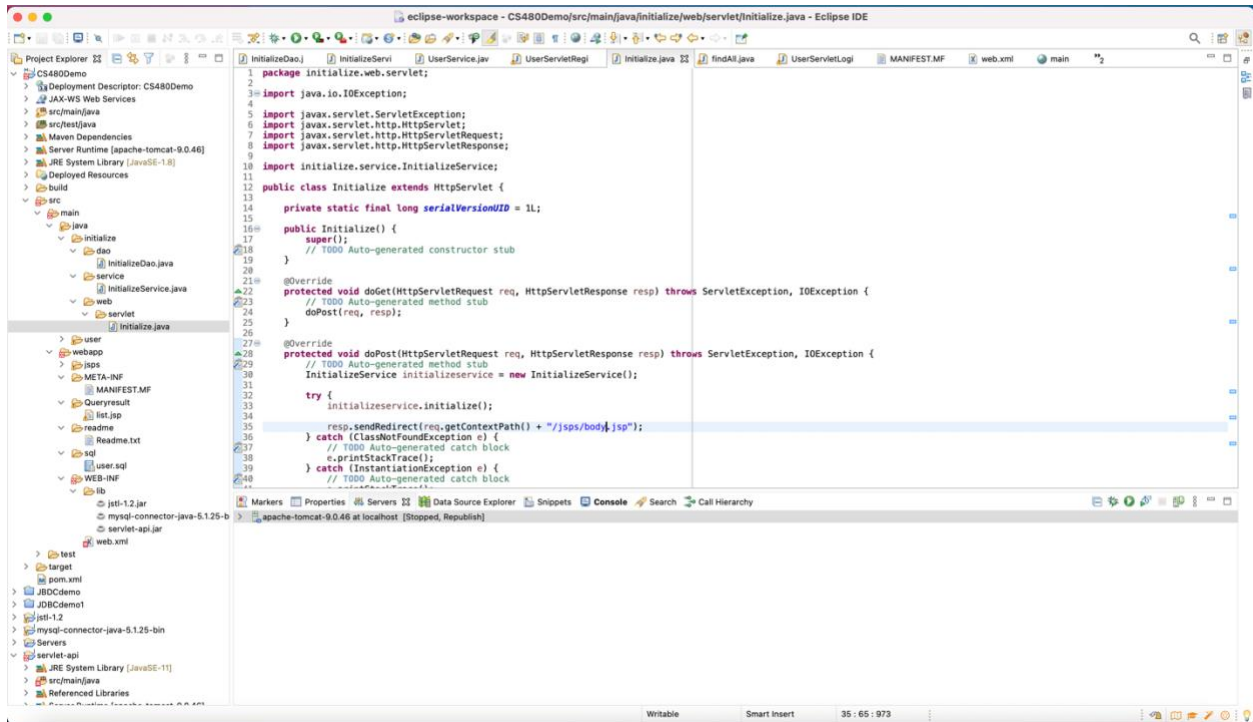
2. A “InitializeDao.java” class was created in the project which had the code to call the “initialize_tables.sql” stored procedure.



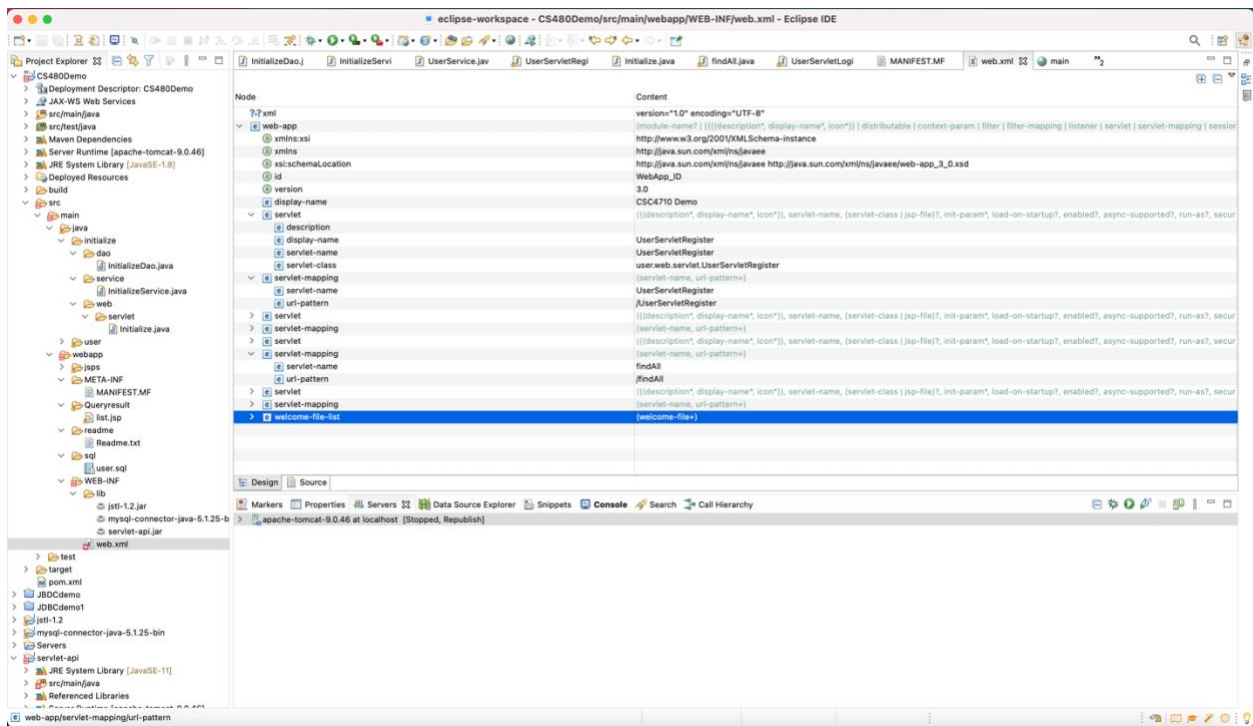
3. An “InitializeService.java” class was created which calls the “initialize_database” function from the “InitializeDao.java” class.



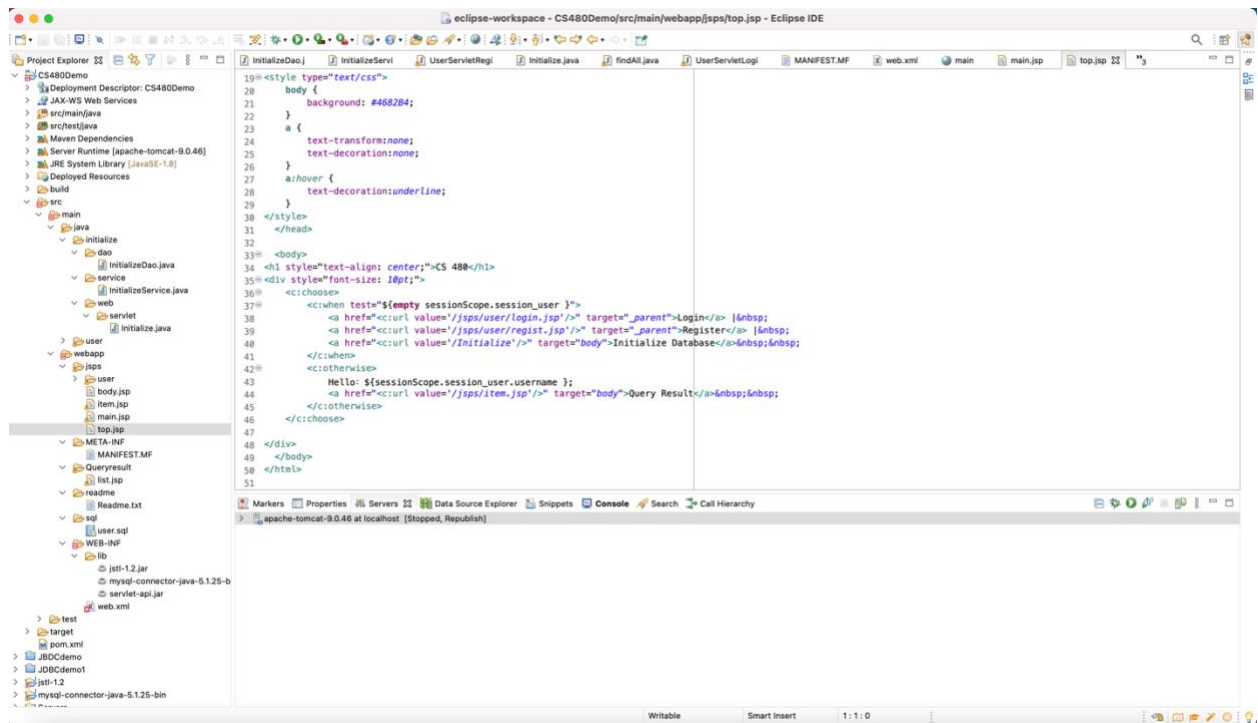
4. A new servlet class “Initialize.java” was created which called the “initialize” function from “InitializeService.java” class. It performs the same action for the post and get methods.

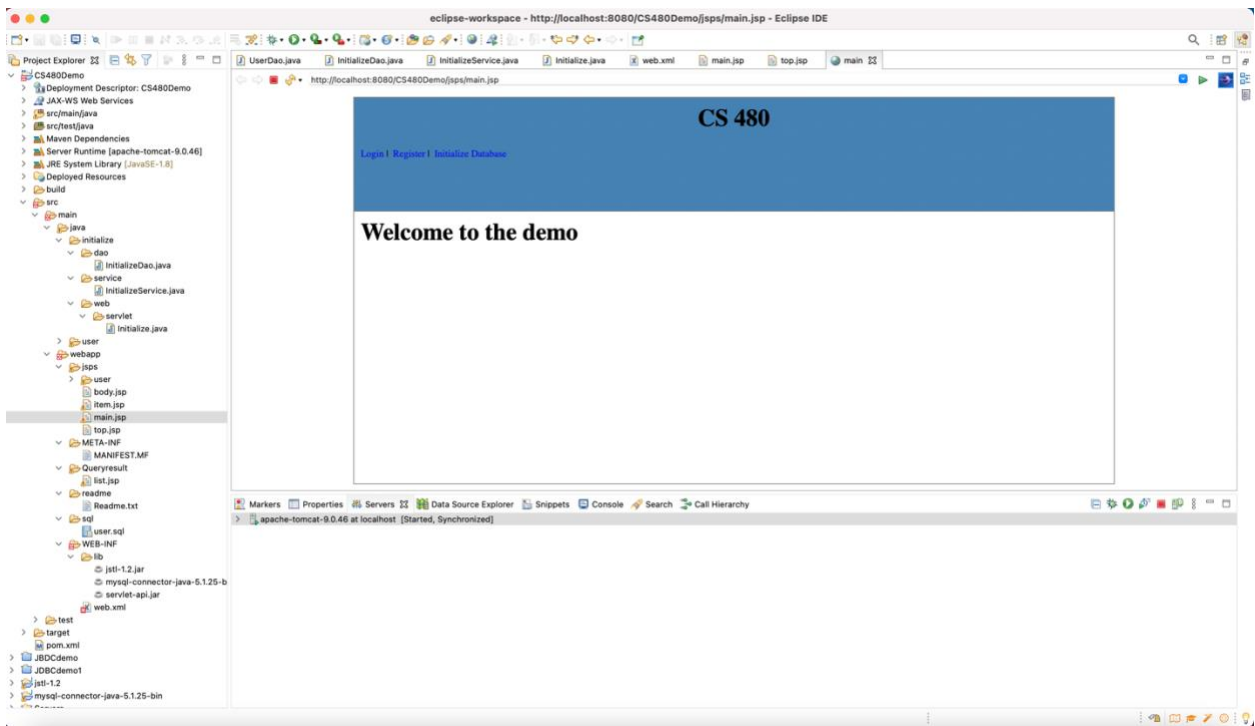


5. The new servlet class information was added to the “web.xml” file. This was done to register the servlet.

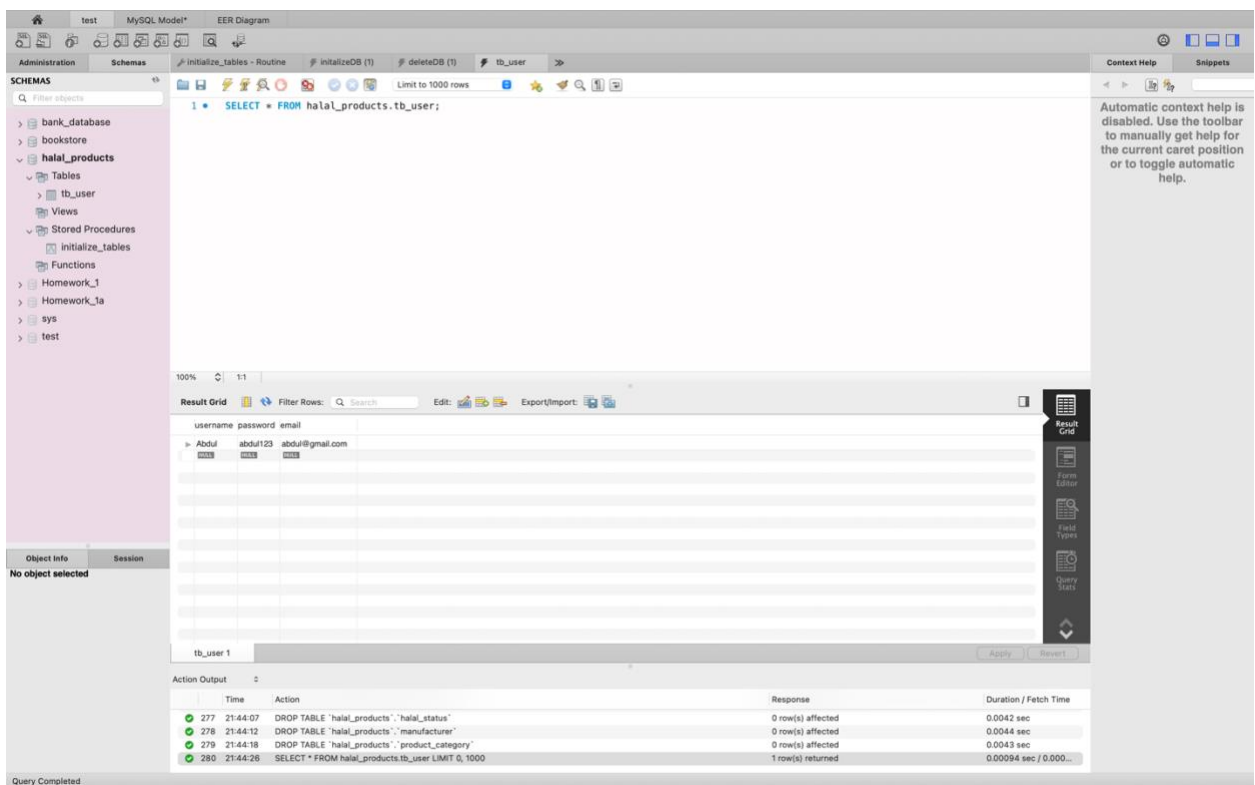


6. The last change was done in the “top.jsp” file. A link “Initialize Database” was added that would call the stored procedure when clicked.





7. Before clicking the database had only the “tb_user” table with a few records.



8. After clicking the link all the tables were created. Each had 3 records. If the table already existed, then the stored procedure would have dropped the tables before creating them again.

The screenshot shows the MySQL Model* EER Diagram interface. The left sidebar displays a tree view of the database schema, including tables like `bank_database`, `bookstore`, `halal_products`, `halal_status`, `manufacturer`, `product`, `product_category`, `product_store`, `region`, `store`, `tb_user`, `Views`, `Stored Procedures`, `Initialize_tables`, `Functions`, `Homework_1`, `Homework_1a`, `sys`, and `test`.

The main window displays a query editor with the following SQL query:

```
1 * SELECT * FROM halal_products.halal_status;
```

The query result is displayed in a table with the following data:

Idstatus
1 YES
2 NO
3 MAY...

The bottom section shows the Action Output table, which lists the actions performed by the stored procedure:

Time	Action	Response	Duration / Fetch Time
278 21:44:12	DROP TABLE 'halal_products'. 'manufacturer'	0 row(s) affected	0.0044 sec
279 21:44:18	DROP TABLE 'halal_products'. 'product_category'	0 row(s) affected	0.0043 sec
280 21:44:26	SELECT * FROM halal_products.tb_user LIMIT 0, 1000	1 row(s) returned	0.00094 sec / 0.000...
281 21:45:01	SELECT * FROM halal_products.halal_status LIMIT 0, 1000	3 row(s) returned	0.00034 sec / 0.000...

The status bar at the bottom indicates "Query Completed".