APRIL 21, 2018

PROJECT REPORT

SUTD WEB BANK APP

MAURITZ ZACHRISSON
50.531 SECURE SOFTWARE ENGINEERING

Table of Contents

Peer Review

Because my partner Roy withdrew from the course, I have done the entire project on my own. I had some contact with Roy, but because I could not get ahold of him for a period of several weeks before he notified his withdrawal, I ended up doing all the coding myself.

System Functionality and Security

Functionality and Security Improvements

AUTHENTICATION AND AUTHORIZATION

Both authentication and authorization has been implemented for the SUTD Web Bank App. As per industry standard:

- The term *authentication* refers to requiring a user to have provided their identity before they can access resources. For the Bank Web App, this means that we require the user to login before they can access a particular resource.
- The term *authorization* refers to ensuring not just that a user is logged in, but that they are actually allowed to access the resource. For the Bank Web App, this means ensuring users must have the “client” or the “staff” permission, respectively, to access any client or staff functionality.

In more practical terms, these authentication and authorization requirements are set up:

- For the Client Dashboard, authentication is now required.
- For the Client Dashboard, authorization ensures the user has the “client” role.
- The New Transaction view has both authentication and authorization (client permission)
- The New Transaction POST endpoint also has authentication and authorization, which means an attacker cannot forge a POST request to create a new transaction without being an authenticated client.
- The Staff Dashboard, both view and POST endpoint, now have authentication and authorization, ensuring anyone who tries to view or modify client registrations and transactions is a logged in bank employee.

NEW TRANSACTION FUNCTIONALITY COMPLETE

The new transaction functionality provided in the starter code is very lacking, and has been fully completed according to the specification, with the exception of batch transaction processing which is not implemented.

The changes made to the new transaction functionality are the following:

- **Transaction code validation**
  When a client makes a new transaction, the transaction code is validated against the database to make sure it both exists, belongs to the current user and is not used.
- **From account selection**
  When making a new transfer, the client has to select which of their accounts they want to deduct funds from, from a list.

- **Recipient account validation**
  When making a new transfer, the system checks to make sure that the recipient account actually exists.
- **Funds check**
  Before a transaction can be made, a check will be made to make sure the user's account has sufficient funds to complete the transaction.
- **Instant transfer for small transactions**
  If the transaction is less than $10,000, the system will approve it and transfer the funds automatically.
- **Large transactions become pending**
  If the transaction is greater than $10,000, no funds will be transferred, and the transaction will be pending a bank employee's approval.

EMPLOYEE TRANSACTION APPROVAL TRANSFERS FUNDS

When a bank employee logs in to their dashboard and accepts a transaction, funds are transferred from the sender's account to the recipient's account. Funds are only transferred if they are available, otherwise the transaction will just be accepted with no money changing hands.

CLIENTS HAVE MULTIPLE ACCOUNTS

Each client can hold more than one bank account, each with their separate balance and account number. When logging in to the client dashboard, balances can be viewed for all accounts. When making a transfer, the client can choose which account to transfer from.

BUG FIXES

- **Fix:** Resolved a bug where ID mismatches between the Client Info and User tables would result in staff being unable to approve or deny new customer requests.
- **Fix:** Resolved a bug where ID mismatches would render the transaction history list empty in the client dashboard view.

Testing

Manual Testing

To ensure system functionality and correctness, a number of different scenarios were tested for each function. Given more time, these scenarios would have been automated in the form of integration testing, but for this project implementation, they were performed "by hand" by interacting with the Bank Web App in a browser.

SIGN UP AND LOGIN

To ensure signup and login worked properly, the following test scenarios were performed:

- Try to sign up for a new account with both incomplete and complete details. For incomplete details, the signup form rejects the application as expected. For complete details, the signup form accepts the application, and places the account as pending approval, and sends out the transaction codes.
- For login, the four scenarios tested were:
  - invalid information,
  - valid but not yet approved account details,
  - valid client account,
  - valid staff account
- All of them behaved as expected, with the first two resulting in rejection and the second two working.

AUTHENTICATION AND AUTHORIZATION

To ensure access control worked properly, the following test scenarios were performed:

- Without logging in, trying to access the client dashboard, new transaction and staff dashboard endpoints. Each of them resulted in a "forbidden" exception, as expected.
- Logged in as a client, the staff dashboard was accessed. This resulted in a "forbidden" exception, as expected.

NEW TRANSACTION

The following test scenarios were performed and verified:

- Invalid transaction code was entered. The system rejected this request, as expected.
- An already used transaction code was entered. The system rejected this, as intended.
- Valid transaction code but bogus recipient account entered. Rejected, as intended.
- Valid code and recipient, but larger amount than what the selected account holds. Triggers an "insufficient funds" error, as expected.

- Making a valid request for transfer for any amount less than $10 000 instantly approves the transaction and transfers the funds, as expected.
- Making a valid request for transfer for any amount above $10 000 will mark the transaction as pending employee approval, and won't transfer any funds, as intended.

CLIENT AND TRANSACTION APPROVAL

- Client registration requests can be approved or denied, as per the specification.
- Denying a transaction means no funds are transferred, as intended.
- Approving a transaction for which the client has available funds transfers the funds and marks the transaction as approved, as intended.
- Approving a transaction which the client could afford at the time of requesting the transfer, but can no longer afford, will mark the transaction as approved but will show an error message to the bank employee, informing them that the accounts were not updated.

Static Analysis

The static analysis tool FindBugs, a common tool for the analysis of Java programs, was used on the codebase. The static analysis tool found 24 warnings in total for the codebase. 11 of the 24 warnings were fields that should be declared final but were not, or getters exposing the reference to the private field by returning the reference directly. The rest of the warnings were mixed. The report in full is attached at the end of this report.

FindBugs Report

Produced using FindBugs3.0.1.

Project: sutdbank[sutdbank]

Metrics

1192 lines of code analyzed, in 40 classes, in 5 packages.

Metric	Total	Density*
High Priority Warnings	13	10.91
Medium Priority Warnings	11	9.23
Total Warnings	**24**	**20.13**

(Defects per Thousand lines of non-commenting source statements)*

Summary

Warning Type	Number
Bad practice Warnings	5
Experimental Warnings	5
Internationalization Warnings	1
Malicious code vulnerability Warnings	11
Performance Warnings	1
Dodgy code Warnings	1
Total	**24**

Warnings

Click on each warning link to see a full description of the issue, and details of how to resolve it.

Bad practice Warnings

Warning	Priority	Details
Store of non serializable	High	Store of non serializable sg.edu.sutd.bank.webapp.model.ClientInfo into HttpSession in

object into HttpSession		sg.edu.sutd.bank.webapp.servlet.ClientDashboardServlet.doGet(HttpServletRequest, HttpServletResponse) In file ClientDashboardServlet.java, line 43 In class sg.edu.sutd.bank.webapp.servlet.ClientDashboardServlet In method sg.edu.sutd.bank.webapp.servlet.ClientDashboardServlet.doGet(HttpServletRequest, HttpServletResponse) Actual type sg.edu.sutd.bank.webapp.model.ClientInfo At ClientDashboardServlet.java:[line 43] At ClientDashboardServlet.java:[line 43]
Store of non serializable object into HttpSession	High	Store of non serializable sg.edu.sutd.bank.webapp.model.ClientAccount into HttpSession in sg.edu.sutd.bank.webapp.servlet.NewTransactionServlet.doGet(HttpServletRequest, HttpServletResponse) In file NewTransactionServlet.java, line 54 In class sg.edu.sutd.bank.webapp.servlet.NewTransactionServlet In method sg.edu.sutd.bank.webapp.servlet.NewTransactionServlet.doGet(HttpServletRequest, HttpServletResponse) Actual type sg.edu.sutd.bank.webapp.model.ClientAccount At NewTransactionServlet.java:[line 54] At NewTransactionServlet.java:[line 54]
Store of non serializable object into HttpSession	High	Store of non serializable sg.edu.sutd.bank.webapp.model.ClientInfo into HttpSession in sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet.doGet(HttpServletRequest, HttpServletResponse) In file StaffDashboardServlet.java, line 69 In class sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet In method sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet.doGet(HttpServletRequest, HttpServletResponse) Actual type sg.edu.sutd.bank.webapp.model.ClientInfo At StaffDashboardServlet.java:[line 69] At StaffDashboardServlet.java:[line 69]
Store of non serializable object into HttpSession	High	Store of non serializable sg.edu.sutd.bank.webapp.model.ClientTransaction into HttpSession in sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet.doGet(HttpServletRequest, HttpServletResponse) In file StaffDashboardServlet.java, line 71 In class sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet In method sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet.doGet(HttpServletRequest, HttpServletResponse) Actual type sg.edu.sutd.bank.webapp.model.ClientTransaction

		At StaffDashboardServlet.java:[line 71] At StaffDashboardServlet.java:[line 71]
Method may fail to close database resource	Medium	sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl.loadAccountInfo(String) may fail to close PreparedStatement In file ClientInfoDAOImpl.java, line 112 In class sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl In method sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl.loadAccountInfo(String) Need to close java.sql.PreparedStatement At ClientInfoDAOImpl.java:[line 112] At ClientInfoDAOImpl.java:[line 112]

Experimental Warnings

Warning	Priority	Details
Method may fail to clean up stream or resource	Medium	sg.edu.sutd.bank.webapp.service.ClientAccountDAOImpl.findAllByUser(int) may fail to clean up java.sql.ResultSet In file ClientAccountDAOImpl.java, line 105 In class sg.edu.sutd.bank.webapp.service.ClientAccountDAOImpl In method sg.edu.sutd.bank.webapp.service.ClientAccountDAOImpl.findAllByUser(int) Reference type java.sql.ResultSet 1 instances of obligation remaining Obligation to clean up resource created at ClientAccountDAOImpl.java:[line 105] is not discharged Path continues at ClientAccountDAOImpl.java:[line 107] Path continues at ClientAccountDAOImpl.java:[line 109] Path continues at ClientAccountDAOImpl.java:[line 118] Remaining obligations: {ResultSet x 1}
Method may fail to clean up stream or resource	Medium	sg.edu.sutd.bank.webapp.service.ClientAccountDAOImpl.findById(int) may fail to clean up java.sql.ResultSet In file ClientAccountDAOImpl.java, line 80 In class sg.edu.sutd.bank.webapp.service.ClientAccountDAOImpl In method sg.edu.sutd.bank.webapp.service.ClientAccountDAOImpl.findById(int) Reference type java.sql.ResultSet 1 instances of obligation remaining Obligation to clean up resource created at ClientAccountDAOImpl.java:[line 80] is not discharged Path continues at ClientAccountDAOImpl.java:[line 82] Path continues at ClientAccountDAOImpl.java:[line 83] Remaining obligations: {ResultSet x 1}

Method may fail to clean up stream or resource	Medium	sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl.loadAccountInfo(String) may fail to clean up java.sql.ResultSet In file ClientInfoDAOImpl.java, line 101 In class sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl In method sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl.loadAccountInfo(String) Reference type java.sql.ResultSet 1 instances of obligation remaining Obligation to clean up resource created at ClientInfoDAOImpl.java:[line 101] is not discharged Path continues at ClientInfoDAOImpl.java:[line 103] Path continues at ClientInfoDAOImpl.java:[line 104] Path continues at ClientInfoDAOImpl.java:[line 105] Path continues at ClientInfoDAOImpl.java:[line 106] Path continues at ClientInfoDAOImpl.java:[line 108] Path continues at ClientInfoDAOImpl.java:[line 109] Path continues at ClientInfoDAOImpl.java:[line 110] Path continues at ClientInfoDAOImpl.java:[line 112] Path continues at ClientInfoDAOImpl.java:[line 113] Path continues at ClientInfoDAOImpl.java:[line 115] Obligation to clean up resource created at ClientInfoDAOImpl.java:[line 115] is not discharged Path continues at ClientInfoDAOImpl.java:[line 116] Path continues at ClientInfoDAOImpl.java:[line 118] Path continues at ClientInfoDAOImpl.java:[line 127] Path continues at ClientInfoDAOImpl.java:[line 128] Path continues at ClientInfoDAOImpl.java:[line 134] Path continues at ClientInfoDAOImpl.java:[line 135] Path continues at ClientInfoDAOImpl.java:[line 137] Path continues at ClientInfoDAOImpl.java:[line 138] Path continues at ClientInfoDAOImpl.java:[line 140] Remaining obligations: {Statement x 1,ResultSet x 1}
Method may fail to clean up stream or resource	Medium	sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl.loadAccountInfo(String) may fail to clean up java.sql.Statement In file ClientInfoDAOImpl.java, line 98 In class sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl In method sg.edu.sutd.bank.webapp.service.ClientInfoDAOImpl.loadAccountInfo(String) Reference type java.sql.Statement 1 instances of obligation remaining Obligation to clean up resource created at ClientInfoDAOImpl.java:[line 98] is not discharged Path continues at ClientInfoDAOImpl.java:[line 100] Path continues at ClientInfoDAOImpl.java:[line 101] Path continues at ClientInfoDAOImpl.java:[line 103] Path continues at ClientInfoDAOImpl.java:[line 104] Path continues at ClientInfoDAOImpl.java:[line 105] Path continues at ClientInfoDAOImpl.java:[line 106] Path continues at ClientInfoDAOImpl.java:[line 108] Path continues at ClientInfoDAOImpl.java:[line 109] Path continues at ClientInfoDAOImpl.java:[line 110]

		Path continues at ClientInfoDAOImpl.java:[line 112] Obligation to clean up resource created at ClientInfoDAOImpl.java:[line 112] is not discharged Path continues at ClientInfoDAOImpl.java:[line 113] Path continues at ClientInfoDAOImpl.java:[line 115] Path continues at ClientInfoDAOImpl.java:[line 116] Path continues at ClientInfoDAOImpl.java:[line 118] Path continues at ClientInfoDAOImpl.java:[line 127] Path continues at ClientInfoDAOImpl.java:[line 128] Path continues at ClientInfoDAOImpl.java:[line 134] Path continues at ClientInfoDAOImpl.java:[line 135] Path continues at ClientInfoDAOImpl.java:[line 137] Path continues at ClientInfoDAOImpl.java:[line 138] Path continues at ClientInfoDAOImpl.java:[line 140] Remaining obligations: {Statement x 1,ResultSet x 1}
Method may fail to clean up stream or resource	Medium	sg.edu.sutd.bank.webapp.service.TransactionCodesDAOImp.findByCode(String, int) may fail to clean up java.sql.ResultSet In file TransactionCodesDAOImp.java, line 74 In class sg.edu.sutd.bank.webapp.service.TransactionCodesDAOImp In method sg.edu.sutd.bank.webapp.service.TransactionCodesDAOImp.findByCode(String, int) Reference type java.sql.ResultSet 1 instances of obligation remaining Obligation to clean up resource created at TransactionCodesDAOImp.java:[line 74] is not discharged Path continues at TransactionCodesDAOImp.java:[line 76] Path continues at TransactionCodesDAOImp.java:[line 77] Remaining obligations: {ResultSet x 1}

Internationalization Warnings

Warning	Priority	Details
Reliance on default encoding	High	Found reliance on default encoding in sutdbank.DbCreator.executeCreateScript(): new java.io.FileReader(File) In file DbCreator.java, line 82 In class sutdbank.DbCreator In method sutdbank.DbCreator.executeCreateScript() Called method new java.io.FileReader(File) At DbCreator.java:[line 82] At DbCreator.java:[line 82]

Malicious code vulnerability Warnings

Warning	Priority	Details
Field isn't final but should be	High	sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.dbUrl isn't final but should be In file AbstractDAOImpl.java, line 41 In class sg.edu.sutd.bank.webapp.service.AbstractDAOImpl Field sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.dbUrl At AbstractDAOImpl.java:[line 41]
Field isn't final but should be	High	sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.driverClassName isn't final but should be In file AbstractDAOImpl.java, line 40 In class sg.edu.sutd.bank.webapp.service.AbstractDAOImpl Field sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.driverClassName At AbstractDAOImpl.java:[line 40]
Field isn't final but should be	High	sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.password isn't final but should be In file AbstractDAOImpl.java, line 45 In class sg.edu.sutd.bank.webapp.service.AbstractDAOImpl Field sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.password At AbstractDAOImpl.java:[line 45]
Field isn't final but should be	High	sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.schema isn't final but should be In file AbstractDAOImpl.java, line 42 In class sg.edu.sutd.bank.webapp.service.AbstractDAOImpl Field sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.schema At AbstractDAOImpl.java:[line 42]
Field isn't final but should be	High	sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.schemaUrl isn't final but should be In file AbstractDAOImpl.java, line 43 In class sg.edu.sutd.bank.webapp.service.AbstractDAOImpl Field sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.schemaUrl At AbstractDAOImpl.java:[line 43]
Field isn't final but should be	High	sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.username isn't final but

		should be In file AbstractDAOImpl.java, line 44 In class sg.edu.sutd.bank.webapp.service.AbstractDAOImpl Field sg.edu.sutd.bank.webapp.service.AbstractDAOImpl.username At AbstractDAOImpl.java:[line 44]
Field isn't final but should be	High	sutdbank.DbCreator.dbScript isn't final but should be In file DbCreator.java, line 20 In class sutdbank.DbCreator Field sutdbank.DbCreator.dbScript At DbCreator.java:[line 20]
May expose internal representation by returning reference to mutable object	Medium	sg.edu.sutd.bank.webapp.model.ClientInfo.getDateOfBirth() may expose internal representation by returning ClientInfo.dateOfBirth In file ClientInfo.java, line 51 In class sg.edu.sutd.bank.webapp.model.ClientInfo In method sg.edu.sutd.bank.webapp.model.ClientInfo.getDateOfBirth() Field sg.edu.sutd.bank.webapp.model.ClientInfo.dateOfBirth At ClientInfo.java:[line 51] At ClientInfo.java:[line 51]
May expose internal representation by returning reference to mutable object	Medium	sg.edu.sutd.bank.webapp.model.ClientTransaction.getDateTime() may expose internal representation by returning ClientTransaction.dateTime In file ClientTransaction.java, line 47 In class sg.edu.sutd.bank.webapp.model.ClientTransaction In method sg.edu.sutd.bank.webapp.model.ClientTransaction.getDateTime() Field sg.edu.sutd.bank.webapp.model.ClientTransaction.dateTime At ClientTransaction.java:[line 47] At ClientTransaction.java:[line 47]
May expose internal representation by incorporating reference to mutable object	Medium	sg.edu.sutd.bank.webapp.model.ClientInfo.setDateOfBirth(Date) may expose internal representation by storing an externally mutable object into ClientInfo.dateOfBirth In file ClientInfo.java, line 55 In class sg.edu.sutd.bank.webapp.model.ClientInfo In method sg.edu.sutd.bank.webapp.model.ClientInfo.setDateOfBirth(Date) Field sg.edu.sutd.bank.webapp.model.ClientInfo.dateOfBirth Local variable named dateOfBirth At ClientInfo.java:[line 55] At ClientInfo.java:[line 55]
May expose internal representation by	Medium	sg.edu.sutd.bank.webapp.model.ClientTransaction.setDateTime(Date) may

incorporating reference to mutable object		expose internal representation by storing an externally mutable object into ClientTransaction.dateTime In file ClientTransaction.java, line 51 In class sg.edu.sutd.bank.webapp.model.ClientTransaction In method sg.edu.sutd.bank.webapp.model.ClientTransaction.setDateTime(Date) Field sg.edu.sutd.bank.webapp.model.ClientTransaction.dateTime Local variable named dateTime At ClientTransaction.java:[line 51] At ClientTransaction.java:[line 51]

Performance Warnings

Warning	Priority	Details
Boxing/unboxing to parse a primitive	High	Boxing/unboxing to parse a primitive sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet.toIntegerArray(String[]) In file StaffDashboardServlet.java, line 125 In class sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet In method sg.edu.sutd.bank.webapp.servlet.StaffDashboardServlet.toIntegerArray(String[]) Called method Integer.intValue() Should call Integer.parseInt(String) instead At StaffDashboardServlet.java:[line 125] At StaffDashboardServlet.java:[line 125]

Dodgy code Warnings

Warning	Priority	Details
Method uses the same code for two branches	Medium	sg.edu.sutd.bank.webapp.servlet.DefaultServlet.doGet(HttpServletRequest, HttpServletResponse) uses the same code for two branches In file DefaultServlet.java, line 39 In class sg.edu.sutd.bank.webapp.servlet.DefaultServlet In method sg.edu.sutd.bank.webapp.servlet.DefaultServlet.doGet(HttpServletRequest, HttpServletResponse) At DefaultServlet.java:[line 39] At DefaultServlet.java:[line 41]

Warning Types

Boxing/unboxing to parse a primitive

A boxed primitive is created from a String, just to extract the unboxed primitive value. It is more efficient to just call the static parseXXX method.

Method uses the same code for two branches

This method uses the same code to implement two branches of a conditional branch. Check to ensure that this isn't a coding mistake.

Reliance on default encoding

Found a call to a method which will perform a byte to String (or String to byte) conversion, and will assume that the default platform encoding is suitable. This will cause the application behaviour to vary between platforms. Use an alternative API and specify a charset name or Charset object explicitly.

May expose internal representation by returning reference to mutable object

Returning a reference to a mutable object value stored in one of the object's fields exposes the internal representation of the object. If instances are accessed by untrusted code, and unchecked changes to the mutable object would compromise security or other important properties, you will need to do something different. Returning a new copy of the object is better approach in many situations.

May expose internal representation by incorporating reference to mutable object

This code stores a reference to an externally mutable object into the internal representation of the object. If instances are accessed by untrusted code, and unchecked changes to the mutable object would compromise security or other important properties, you will need to do something different. Storing a copy of the object is better approach in many situations.

Store of non serializable object into HttpSession

This code seems to be storing a non-serializable object into an HttpSession. If this session is passivated or migrated, an error will result.

Field isn't final but should be

This static field public but not final, and could be changed by malicious code or by accident from another package. The field could be made final to avoid this vulnerability.

Method may fail to clean up stream or resource

This method may fail to clean up (close, dispose of) a stream, database object, or other resource requiring an explicit cleanup operation.

In general, if a method opens a stream or other resource, the method should use a try/finally block to ensure that the stream or resource is cleaned up before the method returns.

This bug pattern is essentially the same as the OS_OPEN_STREAM and ODR_OPEN_DATABASE_RESOURCE bug patterns, but is based on a different (and hopefully better) static analysis technique. We are interested is getting feedback about the usefulness of this bug pattern. To send feedback, either:

- send email to findbugs@cs.umd.edu
- file a bug report: http://findbugs.sourceforge.net/reportingBugs.html

In particular, the false-positive suppression heuristics for this bug pattern have not been extensively tuned, so reports about false positives are helpful to us.

See Weimer and Necula, *Finding and Preventing Run-Time Error Handling Mistakes*, for a description of the analysis technique.

Method may fail to close database resource

The method creates a database resource (such as a database connection or row set), does not assign it to any fields, pass it to other methods, or return it, and does not appear to close the object on all paths out of the method. Failure to close database resources on all paths out of a method may result in poor performance, and could cause the application to have problems communicating with the database.