**CCT College Dublin**

**Assessment Cover Page**

|  |  |
| --- | --- |
| **Module Title:** | *Databases & OOC Integrated CA* |
| **Assessment Title:** | Integrated Databases and OCC |
| **Lecturer Name:** | James Garza (james@cct.ie)  Kayoum Khbuli (kkhbuli@cct.ie) |
| **Student Full Name:** | Rhuan Mendanha Eli Raimundo  Henrique Queiroz de Morais |
| **Student Number:** | Rhuan: 2023186  Henrique: 2023288 |
| **Assessment Due Date:** | Sunday, 22th December 2024 @ 23:59 |
| **Date of Submission:** |  |

**Declaration**

|  |
| --- |
| By submitting this assessment, I confirm that I have read the CCT policy on Academic Misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source. I declare it to be my own work and that all material from third parties has been appropriately referenced. I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution. |

Table of Contents

[Introduction: 3](#_Toc185414710)

[1 - System Design & Architecture: 3](#_Toc185414711)

[Class Structure: 3](#_Toc185414712)

[Encapsulation: 3](#_Toc185414713)

[Polymorphism: 3](#_Toc185414714)

[2 - Database Schema Design: 4](#_Toc185414715)

[Entities & Relationships: 4](#_Toc185414716)

[Normalization: 4](#_Toc185414717)

[3 – Chen Diagram & Relational Model: 6](#_Toc185414718)

[1st Checkpoint Chen diagram: 6](#_Toc185414719)

[2nd Chen diagram: 6](#_Toc185414720)

[4 – Backend Development: 7](#_Toc185414721)

[CRUD Operations: 7](#_Toc185414722)

[Database Connection: 7](#_Toc185414723)

[Error Handling: 7](#_Toc185414724)

[5 – Tax Calculation: 7](#_Toc185414725)

[Business Logic: 7](#_Toc185414726)

[Database Logging: 8](#_Toc185414727)

[6 – Screencast: 8](#_Toc185414728)

[7 – GitHub Repository: 8](#_Toc185414729)

[Rhuan: 8](#_Toc185414730)

[Henrique: 8](#_Toc185414731)

[8 – Testing & Validation: 8](#_Toc185414732)

[9 – Challenges & Solutions: 8](#_Toc185414733)

[10 – Conclusion: 9](#_Toc185414734)

[11 – References & Citations: 9](#_Toc185414735)

# Introduction:

The User Management System is a software program for controlling the activities of user accounts and the information related to them in an organization. Developed in Java and MySQL, the system is implemented using an object-oriented programming (OOP) approach, which allows it’s easy...” etc. English writing skills. The system enables CRUD (Create, Read, Update and Delete) operations on user’s records, different user roles (Admin and RegularUser) are supported, and tax calculation features are added into the system. The system, utilizing strong validation and error-handling mechanisms, presents a quick, safe, and simple approach to managing and controlling user data.

# 1 - System Design & Architecture:

## Class Structure:

4The project is organised using Object-Oriented Principles:

User: A base class representing any user in the system with identifiable properties (e.g., userId, username, role).

Admin: Inherits from User and adds some interface functions specific to admin, including viewing all users and deleting users.

RegularUser: Inherits from User and allows restricted usage of the system, which may include updating the user’s details.

DatabaseConnection: Responsible for connecting to the MySQL database.

UserService: Active in several aspects, such as adding users, fetching users from the system, updating users, and deletion of users.

## Encapsulation:

Sensitive information of users like passwords has been effectively encrypted and this mechanism implements the control on attributes of the user account through the use of Accessor methods. The connection parameters used for the database are hidden from the other pieces of code and exposed only in the DatabaseConnection class.

## Polymorphism:

For example, polymorphism is achieved by method overriding in Admin and RegularUser. Admins can perform all CRUD operations for every user in the system. On the other hand, Regular users can update their own profile only. The methods of the UserService are focused on different user categories in terms of the assumed application role.

# 2 - Database Schema Design:

Entities & Relationships:  
A screenshot of a computer

Description automatically generated

## Normalization:

First of all, I created a table in Excel with users that we would like to add to the database, thinking of a company, we added the necessary information so that our database was designed with this idea in mind. The information was: user\_id, user\_name, title first\_name, last\_\_name , initial, gender language email phone\_numbe, Address, state, position, role, user\_Type, department, manager, salary, start\_date , end\_date and password.

A screenshot of a computer

Description automatically generatedThe structure was based on the user’s table, taking as an example the Checkpoint 1 database where all the information was stored in just one table (employees) and thinking about the normalization process, which causes redundancy.  
  
  
A screenshot of a computer code

Description automatically generated

As the 1NF normalisation process requires, we need there to be no divisible values ​​in the table. Then, proceed by removing the groups that repeat themselves and creating tables such as Role and Departments and using foreign keys to connect them.  
  
Role contains information regarding the users: Admin, Manager and Regular  
Department contains four different departments: HR, IT, Finance and Marketing.  
The status tells us whether the user is a Leaver or a Current

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated   
  
Now thinking about removing the non-key attributes that we still had and trying to agree with 2NF I created a Status table that tells us if the user is a Leaver or Current and we added status as a foreign key to the user table.  
  
A screenshot of a computer

Description automatically generated   
  
  
In the end, I eliminated all attributes that could be stored in other tables. At this point, I realized that city in country was quite redundant in the users table, so I created the addresses table with the foreign key in the users table, as well as contacts, and it is redundant to have email and phone number in the users table.   
  
A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated  
  
This way we can reduce the redundancy and risks of inconsistency that we could have in the first checkpoint, in addition to having the possibility of adding data and maintaining data much simpler and more efficient.

# 3 – Chen Diagram & Relational Model:

1st Checkpoint Chen diagram: We based our diagram and database on my diagram at the first checkpoint (Henrique):

A diagram of a company

Description automatically generated

2nd Chen diagram:  
When creating new tables and trying to reduce redundancy, tables like Departments, Roles, and Contacts were created to eliminate the duplicate columns in the first diagram.

Also, when creating the status table, we removed the partial dependencies that we still have. After that, we still had non-key columns that depended on another non-key column, which was city\_name and country\_name. When creating the Addresses table, we removed these.  
 A diagram of a company

Description automatically generated

# 4 – Backend Development:

## CRUD Operations:

Create: start new users by inserting them into the database.

Read: retrieve a user or a set of users by selecting them from the database.

Update: Here, a user may need to change their username. For that, they can update the details.

Delete: using user\_id, a user may delete an account through the DELETE process.

## Database Connection:

Connection is established via the DatabaseConnection class using the JDBC driver:

String url = "jdbc:mysql://localhost:3306/OOC\_DB";

Connection connection = DriverManager.getConnection(url, "ooc2023", "ooc2023");

## Error Handling:

An invalid input error, a connection failure, or an SQL error is managed by defining try-catch methods.

For instance, if the database connection cannot be established, the message is output on the console.

# 5 – Tax Calculation:

## Business Logic:

The most suitable approach to implementing a tax calculation feature may be in the following manner:

Gross Income: The amount of money that the user declares as their income.

Tax Credits: These are then deducted from the gross income.

Taxes, such as PAYE, USC, and PRSI, are determined by the taxation rates already set.

## Database Logging:

All tax calculations, the parameters and their results may be recorded in a new structure, which may have such fields:

calculation\_id (PK)

user\_id (FK)

gross\_income

tax\_credits

tax\_owed

calculation\_date

This provides a complete history of all activities associated with tax operations.

# 6 – Screencast:

# 7 – GitHub Repository:

Rhuan: <https://github.com/CCT-Dublin/ooc-db-RhuanMendanha.git>

Henrique: <https://github.com/CCT-Dublin/ooc-db-henr1quemorais.git>

# 8 – Testing & Validation:

When testing the connection, we noted that we could connect to the database, but we got an Error fetching users: Column 'role' not found.  
  
A black text on a white background

Description automatically generated  
  
Then Henrique proceeded to reformulate the tables and foreign keys, considering the normalisation processes.

# 9 – Challenges & Solutions:

The core problem for the client in this solution was to ensure proper working of high interaction between the Java application and MySQL Database. Such issues as invalid queries, connection or data integrity issues were set aside by developing a specialised database connection class and extensive testing of SQL commands. Another challenge to address was the need to come up with an efficient and secure layout to deal with various user types. This was achieved through the use of OOP principles such as inheritance and polymorphism in which Admin and RegularUser roles were implemented with different behaviours. Also, due to the existence of such mechanisms as input validation and error handlers, it was possible to use the system without entering invalid information and operating smoothly.

# 10 – Conclusion:

This project successfully shows the scalable and maintainable user management system applied with Java and MySQL to accomplish information retrieval more efficiently. Also, the incorporation of OOP principles ensures that the system will be more adaptable and extensible for future developments. By accommodating the difficulties faced during its deployment, the project has provided a good basis for the safe storage and execution of user data and business logic. Thus, it is ideal for business environments demanding effective user guidance and activity recording.

# 11 – References & Citations:

docs.oracle.com. Java JDBC API. [online] Available at: <https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/>.

GeeksforGeeks (2021). Types of Statements in JDBC. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/types-of-statements-in-jdbc/>.

w3schools (2019). SQL Tutorial. [online] W3schools.com. Available at: <https://www.w3schools.com/sql/>.

W3Schools. SQL joins. [online] W3schools.com. Available at: <https://www.w3schools.com/sql/sql_join.asp>.

www.youtube.com Programming with Mosh - YouTube. [online] Available at: <https://www.youtube.com/c/programmingwithmosh>.

www.youtube.com Java Brains - YouTube. [online] Available at: <https://www.youtube.com/user/koushks>.

www.tutorialspoint.com Java - Exceptions - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/java/java_exceptions.htm>.

Oracle (2019). Collections Framework Overview. [online] Oracle.com. Available at: <https://docs.oracle.com/javase/8/docs/technotes/guides/collections/overview.html>.

docs.oracle.com. Classes (The JavaTM Tutorials > Learning the Java Language > Classes and Objects). [online] Available at: <https://docs.oracle.com/javase/tutorial/java/javaOO/classes.html>.

downloads.mysql.com. *MySQL: Download MySQL Connector/J (Archived Versions)*. [online] Available at: <https://downloads.mysql.com/archives/c-j/>.

docs.oracle.com. Classes (The JavaTM Tutorials > Learning the Java Language > Classes and Objects). [online] Available at: <https://docs.oracle.com/javase/tutorial/java/javaOO/classes.html>.

GEEKSFORGEEKS (2021). How to Print Colored Text in Java Console? [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/how-to-print-colored-text-in-java-console/>.