Faculty of
Electronic Engineering
كلية الهندسة الإلكترونية

جامعة المنوفية
كلية الهندسة الإلكترونية بمنوف
قسم هندسة و علوم حاسب

Work Space System Menouf

Supervised by:

د. أحمد محمد عبد العظيم غزيه

2023/2024

GitHub Repository Link:

MohamedHelmy296/Learning-Java: Learning Java and OOP (github.com)

Team Memebers:

ID	Name	Email
2000317	Mohamed Helmy Hussein Mostafa	mohamed.helmy1296@gmail.com
2000319	Mohamed Rabee Mohamed Abdelkareem	mohamedrabee20012025@gmail.com

Team name : Work space system Menouf

Project Description:

The project is a **Java-based Work space management system** designed to efficiently organize and manage a **Work space**'s collection of items and users.

The system allows users (administrators-backend enginnering) to perform various operations related to the Work space's inventory and user base.

The core entities in the system are Work space Items (books, magazines, CDs) and Work space Users (students, professors, staff, visitors).

Users can interact with the system through a menu-driven command-line interface, enabling them to add, delete, and display information about different types of items and users.

The project embraces object-oriented programming principles by defining classes for each entity, fostering code modularity and maintainability.

Exception handling is implemented to manage potential input errors, providing a more user-friendly experience.

Capabilities:

1. Item Management:

- Add books, magazines, and CDs to the Work space's collection, specifying details such as title, author, version, artist, and page count.
- Delete items from the collection based on user input, ensuring efficient inventory management.

2. User Management:

- Add Work space users, including students, professors, staff, and visitors, details such as name, user ID, academic number, department, position, and purpose.
- Delete items from the collection based on user input, ensuring efficient inventory management.

3. Display Information:

- Display detailed information about Work space items (books, magazines, CDs) and Work space users (students, professors, staff, visitors) separately, offering a comprehensive view of the Work space's assets and user community.

4. Input Validation and Exception Handling:

- Validate user inputs to prevent errors, ensuring that only valid and appropriate data is accepted during operations.
- This proactive approach to error management not only prevents program crashes but also contributes to a more user-friendly experience, guiding users on correct inputs and interactions with the Work space management system.

5. Modular Design:

- Employ a modular design with well-organized classes, methods, and inheritance to enhance code readability, maintainability, and scalability.

6. User Interface:

- Provide a menu-driven command-line interface that allows users to intuitively navigate through different options, making the system accessible to both experienced and novice users.

7. Object-Oriented Principles:

- Utilize object-oriented programming principles, such as encapsulation and inheritance, to model the entities in the system, ensuring a clear and extensible code structure.

Team: Work space system Menouf

WorkspaceUser class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getName()		String	Retrieves the current value of the name attribute. **Return** type of String	**Mohamed Rabee**
setName()	Name :String		Sets the value of the name attribute to the provided string. **Input** Name :String	**Mohamed Rabee**
getUserId()		int	Retrieves the current value of the userId attribute. **Return** type of int	**Mohamed Rabee**
setUserId()	userId : int		Sets the value of the userId attribute to the provided integer. **Input** Name : int	**Mohamed Rabee**

Student class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getAcademicNumber()		int	Retrieves the current value of the academicNumber attribute. **Return** type of int	**Mohamed Rabee**
setAcademicNumber()	academicNumber : int		Sets the value of the academicNumber attribute to the provided integer. **Input** academicNumber: int	**Mohamed Rabee**
equals()		int	Overrides the default equals method to compare two Student objects based on the equality of their issn attributes. It assumes the existence of an issn attribute in the Student class.	**Mohamed Rabee**

Professor class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getDepartment()		`String`	Retrieves the current value of the `department` attribute. **Return** type of `String`	**Mohamed Rabee**
setDepartment()	`Name:String`		Sets the value of the `department` attribute to the provided string. **Input** `Name :String`	**Mohamed Rabee**
equals()			Overrides the default `equals` method to compare two `Professor` objects based on the equality of their `issn` attributes. It assumes the existence of an `issn` attribute in the `Professor` class.	**Mohamed Rabee**

Visitor class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getPurpose()		`String`	Retrieves the current value of the `purpose` attribute. **Return** type of `String`	**Mohamed Rabee**
setPurpose()	`purpose :String`		Sets the value of the `purpose` attribute to the provided string **Input** `purpose:String`	**Mohamed Rabee**
equals()		`int`	Overrides the default `equals` method to compare two `Visitor` objects based on the equality of their `issn` attributes. It assumes the existence of an `issn` attribute in the `Visitor` class.	**Mohamed Rabee**

Staff class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getPosition()		`String`	Retrieves the current value of the `position` attribute. **Return** type of `String`	**Mohamed Rabee**
setPosition()	`position:String`		Sets the value of the `position` attribute to the provided string. **Input** `Position:String`	**Mohamed Rabee**
equals()			Overrides the default `equals` method to compare two `Staff` objects based on the equality of their `issn` attributes. It assumes the existence of an `issn` attribute in the `Staff` class.	**Mohamed Rabee**

WorkspaceItem class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getTitle()		`String`	Retrieves the current value of the `title` attribute. **Return** type of `String`	**Mohamed Helmy**
setTitle()	`title:String`		Sets the value of the `title` attribute to the provided string. **Input** `title:String`	**Mohamed Helmy**
getAuthor()		`String`	Retrieves the current value of the `author` attribute. **Return** type of `String`	**Mohamed Helmy**
setAuthor()	`author:String`		Sets the value of the `author` attribute to the provided string. **Input** `author:String`	**Mohamed Helmy**

Book class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getPageCount()		int	Retrieves the current value of the `pageCount` attribute. **Return** type of `int`	**Mohamed Helmy**
setPageCount()	`pageCount: int`		Sets the value of the `pageCount` attribute to the provided integer. **Input** `pageCount: int`	**Mohamed Helmy**
equals()			Overrides the default `equals` method to compare two `Book` objects based on the equality of their `issn` attributes. It assumes the existence of an `issn` attribute in the `Book` class.	**Mohamed Helmy**

CD class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getArtist()		`String`	Retrieves the current value of the `artist` attribute. **Return** type of `String`	**Mohamed Helmy**
setArtist()	`artist:String`		Sets the value of the `artist` attribute to the provided string. **Input** `artist:String`	**Mohamed Helmy**
equals()			Overrides the default `equals` method to compare two `CD` objects based on the equality of their `issn` attributes. It assumes the existence of an `issn` attribute in the `CD` class.	**Mohamed Helmy**

Magazine class

Module Name	Input Parameters	Output (Return) of the module	Description	Team member
getArtist()		`int`	Retrieves the current value of the `version` attribute. **Return** type of `int`	**Mohamed Helmy**
setArtist()	`version: int`		Sets the value of the `version` attribute to the provided integer. **Input** `pageCount: int`	**Mohamed Helmy**
equals()			Overrides the default `equals` method to compare two `Magazine` objects based on the equality of their `issn` attributes. It assumes the existence of an `issn` attribute in the `Magazine` class.	**Mohamed Helmy**

WorkspaceSystem class

Module Name	Input Parameters	Output (Return) of the module	Dscription	Team member
getConnection()		**Connection**	The **getConnection** module is a Java method designed to establish a connection to a MySQL database using JDBC. It dynamically loads the MySQL JDBC driver class and then uses the **DriverManager.getConnection** method to connect to the specified MySQL database with the provided credentials**.**	**Mohamed Helmy**

WorkspaceSystem class

Module Name	Input Parameters	Output (Return) of the module	Dscription	Team member
fill_book ( )	**Connection : Connection**	**void**	The **fill_book** method is designed to fetch data from the "Book" table in a MySQL database using the provided **Connection** object. It uses a **Statement** to execute a SQL query (**"select * from Book"**) and retrieves the results in a **ResultSet**.	**Mohamed Helmy**
fill_magazine ( )	**Connection : Connection**	**void**	The **fill_magazine** method is designed to fetch data from the "Magazine" table in a MySQL database using the provided **Connection** object. It uses a **Statement** to execute a SQL query (**"select * from Magazine"**) and retrieves the results in a **ResultSet**.	**Mohamed Helmy**
fill_cd ( )	**Connection : Connection**	**void**	The **fill_cd** method is designed to fetch data from the "CD" table in a MySQL database using the provided **Connection** object. It uses a **Statement** to execute a SQL query (**"select * from CD"**) and retrieves the results in a **ResultSet**.	**Mohamed Helmy**
fill_student ( )	**Connection : Connection**	**void**	The **fill_student** method is designed to fetch data from the "Student" table in a MySQL database using the provided **Connection** object. It uses a **Statement** to execute a SQL query (**"select * from Student"**) and retrieves the results in a **ResultSet**.	**Mohamed Helmy**
fill_professor ( )	**Connection : Connection**	**void**	database The fill_professor method is designed to fetch data from the "Professor" table in a MySQL database using the provided Connection object. It uses a Statement to execute a SQL query ("select * from Professor") and retrieves the results in a ResultSet.	**Mohamed Helmy**

WorkspaceSystem class

Module Name	Input Parameters	Output (Return) of the module	Dscription	Team member
fill_staff ( )	**Connection : Connection**	**void**	The **fill_staff** method is designed to fetch data from the "Staff" table in a MySQL database using the provided Connection object. It uses a Statement to execute a SQL query ("select * from Staff") and retrieves the results in a **ResultSet**.	**Mohamed Helmy**
fill_visitor( )	**Connection : Connection**	**void**	The **fill_visitor** method is designed to fetch data from the "Visitor" table in a MySQL database using the provided Connection object. It uses a Statement to execute a SQL query ("select * from Visitor") and retrieves the results in a **ResultSet**.	**Mohamed Helmy**
addBook( )	**Connection : Connection** **book: Book**	**void**	The **addBook** method is designed to add a new book to the "Book" table in a MySQL database using the provided Connection object. It uses a **PreparedStatement** to execute a parameterized SQL insert query, setting the values of the book's title, author, and page count. After executing the query, it adds the provided Book object to a collection (presumably named books).	**Mohamed Helmy**
addMagazine( )	**Connection : Connection** **magazine: Magazine**	**void**	The **addMagazine** method is similar to **addBook** but is tailored for adding records to the "Magazine" table. It uses a **PreparedStatement** to execute a parameterized SQL insert query, setting the values of the magazine's title, author, and version. After executing the query, it adds the provided Magazine object to a collection (presumably named magazines).	**Mohamed Helmy**

WorkspaceSystem class

Module Name	Input Parameters	Output (Return) of the module	Dscription	Team member
addCD( )	**Connection : Connection** **cd: CD**	**void**	The **addCD** method is similar to **addBook** but is tailored for adding records to the "CD" table. It uses a **PreparedStatement** to execute a parameterized SQL insert query, setting the values of the CD's title, author, and artist. After executing the query, it adds the provided CD object to a collection (presumably named **cds**).	**Mohamed Rabee**
deleteBook( )	**Connection : Connection** **book: Book**	**void**	The **deleteBook** method checks if the specified book is present in the books collection. If it is, it uses a **PreparedStatement** to execute a parameterized SQL delete query, deleting the book with matching title, author, and page count. After executing the query, it removes the provided Book object from the books collection.	**Mohamed Rabee**
deleteMagazine( )	**Connection : Connection** **magazine: Magazine**	**void**	The **deleteMagazine** method checks if the specified magazine is present in the magazines collection. If it is, it uses a **PreparedStatement** to execute a parameterized SQL delete query, deleting the magazine with matching title, author, and version. After executing the query, it removes the provided Magazine object from the magazines collection.	**Mohamed Rabee**
deleteCD( )	**Connection : Connection** **cd: CD**	**void**	The **deleteCD** method checks if the specified CD is present in the **cds** collection. If it is, it uses a **PreparedStatement** to execute a parameterized SQL delete query, deleting the CD with matching title, author, and artist. After executing the query, it removes the provided CD object from the **cds** collection.	**Mohamed Rabee**

WorkspaceSystem class

Module Name	Input Parameters	Output (Return) of the module	Dscription	Team member
addStudent( )	**Connection : Connection** **student: Student**	**void**	The **addStudent** method is designed to add a new student to the "Student" table in a MySQL database using the provided Connection object. It uses a **PreparedStatement** to execute a parameterized SQL insert query, setting the values of the student's name, **userId**, and academic number. After executing the query, it adds the provided Student object to a collection (presumably named students).	**Mohamed Rabee**
addProfessor ( )	**Connection : Connection** **professor: Professor**	**void**	The **addProfessor** method is similar to **addStudent** but is tailored for adding records to the "Professor" table. It uses a **PreparedStatement** to execute a parameterized SQL insert query, setting the values of the professor's name, **userId**, and department. After executing the query, it adds the provided Professor object to a collection (presumably named professors).	**Mohamed Helmy**
addStaff( )	**Connection : Connection** **staffMember:Staff**	**void**	The **addStaff** method is similar to **addProfessor** but is tailored for adding records to the "Staff" table. It uses a **PreparedStatement** to execute a parameterized SQL insert query, setting the values of the staff member's name, **userId**, and position. After executing the query, it adds the provided Staff object to a collection (presumably named staff).	**Mohamed Helmy**
addVisitor( )	**Connection : Connection** **visitor: Visitor**	**void**	The **addVisitor** method is similar to **addStaff** but is tailored for adding records to the "Visitor" table. It uses a **PreparedStatement** to execute a parameterized SQL insert query, setting the values of the visitor's name, **userId**, and purpose. After executing the query, it adds the provided Visitor object to a collection (presumably named visitors).	**Mohamed Helmy**

WorkspaceSystem class

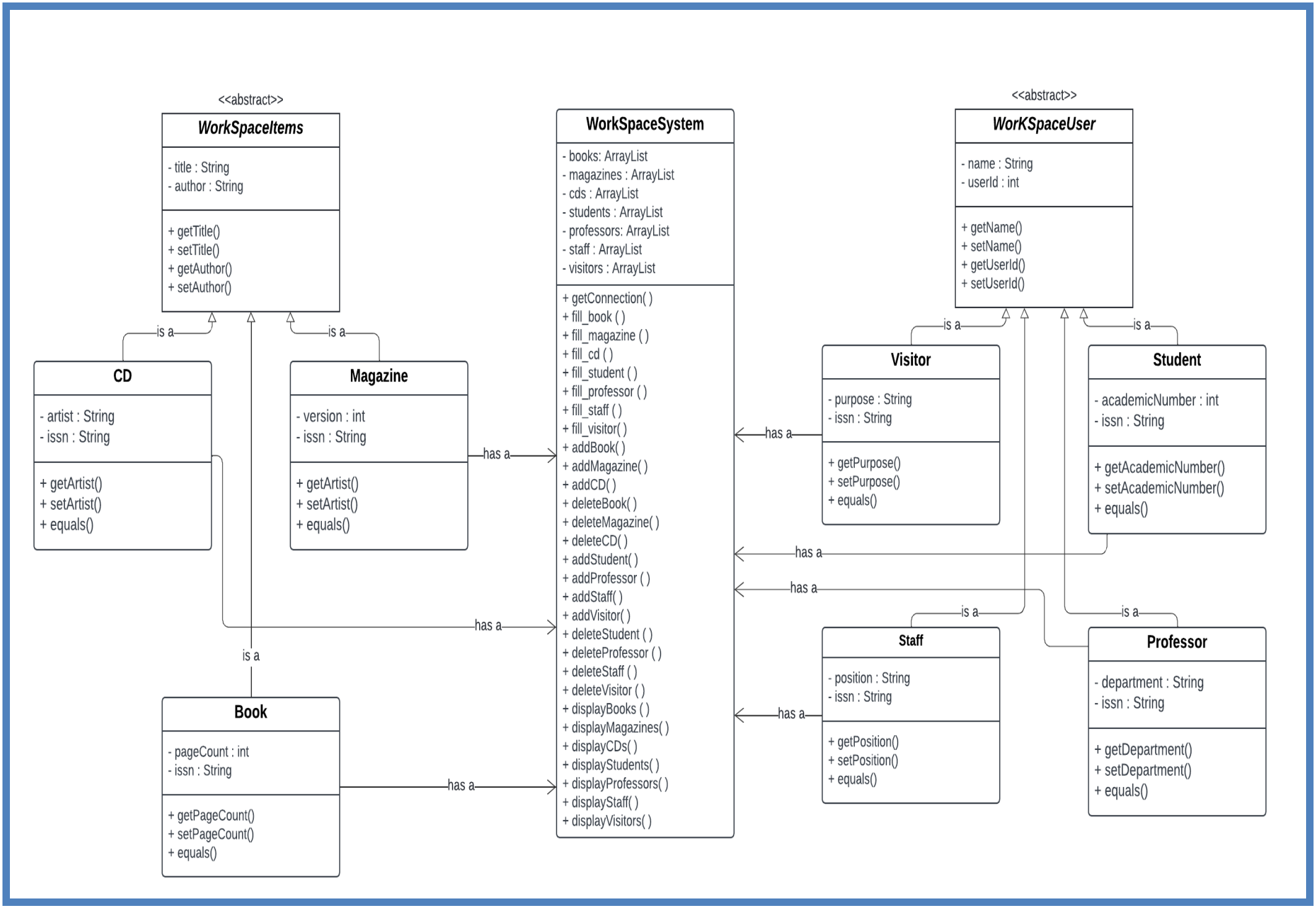
Module Name	Input Parameters	Output (Return) of the module	Dscription	Team member
deleteStudent ( )	**Connection : Connection** **student: Student**	**void**	The **deleteStudent** method checks if the specified student is present in the **students** collection. If it is, it uses a **PreparedStatement** to execute a parameterized SQL delete query, deleting the student with matching name, **userId**, and academic number. After executing the query, it removes the provided Student object from the students collection.	**Mohamed Helmy**
deleteProfessor ( )	**Connection : Connection** **professor: Professor**	**void**	The **deleteProfessor** method checks if the specified professor is present in the **professors** collection. If it is, it uses a **PreparedStatement** to execute a parameterized SQL delete query, deleting the professor with matching name, **userId**, and department. After executing the query, it removes the provided Professor object from the professors collection.	**Mohamed Helmy**
deleteStaff ( )	**Connection : Connection** **staffMember:Staff**	**void**	The **deleteStaff** method checks if the specified staff member is present in the staff collection. If it is, it uses a **PreparedStatement** to execute a parameterized SQL delete query, deleting the staff member with matching name, **userId**, and position. After executing the query, it removes the provided Staff object from the staff collection.	**Mohamed Helmy**
deleteVisitor ( )	**Connection : Connection** **visitor: Visitor**	**void**	The **deleteVisitor** method checks if the specified visitor is present in the visitors collection. If it is, it uses a **PreparedStatement** to execute a parameterized SQL delete query, deleting the visitor with matching name, **userId**, and purpose. After executing the query, it removes the provided Visitor object from the visitors collection.	**Mohamed Helmy**

WorkspaceSystem class

Module Name	Input Parameters	Output (Return) of the module	Dscription	Team member
displayBooks ( )	**None**	**void**	The **displayBooks** method prints information about each book in the books collection to the console. It iterates through the books collection, retrieves the title, author, and page count of each Book object, and prints this information in a formatted manner.	**Mohamed Helmy**
displayMagazines( )	**None**	**void**	The **displayMagazines** method prints information about each magazine in the magazines collection to the console. It iterates through the magazines collection, retrieves the title, author, and version (issue number) of each Magazine object, and prints this information in a formatted manner.	**Mohamed Helmy**
displayCDs( )	**None**	**void**	The **displayCDs** method prints information about each CD in the **cds** collection to the console. It iterates through the **cds** collection, retrieves the title, author, and artist of each CD object, and prints this information in a formatted manner.	**Mohamed Rabee**
displayStudents( )	**None**	**void**	The **displayStudents** method prints information about each student in the **students** collection to the console. It iterates through the students collection, retrieves the name, user ID, and academic number of each Student object, and prints this information in a formatted manner.	**Mohamed Rabee**

WorkspaceSystem class

Module Name	Input Parameters	Output (Return) of the module	Dscription	Team member
displayProfessors( )	None	void	The **displayProfessors** method prints information about each professor in the professors collection to the console. It iterates through the professors collection, retrieves the name, user ID, and department of each Professor object, and prints this information in a formatted manner.	**Mohamed Rabee**
displayStaff( )	None	void	The **displayStaff** method prints information about each staff member in the staff collection to the console. It iterates through the staff collection, retrieves the name, user ID, and position of each Staff object, and prints this information in a formatted manner.	**Mohamed Rabee**
displayVisitors( )	None	void	The **displayVisitors** method prints information about each visitor in the visitors collection to the console. It iterates through the visitors collection, retrieves the name, user ID, and purpose of each Visitor object, and prints this information in a formatted manner.	**Mohamed Rabee**


<<abstract>>
WorkSpaceItems
- title : String
- author : String
+ getTitle()
+ setTitle()
+ getAuthor()
+ setAuthor()
is a
is a
CD
- artist : String
- issn : String
+ getArtist()
+ setArtist()
+ equals()
Magazine
- version : int
- issn : String
+ getArtist()
+ setArtist()
+ equals()
has a
has a
is a
Book
- pageCount : int
- issn : String
+ getPageCount()
+ setPageCount()
+ equals()
has a
WorkSpaceSystem
- books: ArrayList
- magazines : ArrayList
- cds : ArrayList
- students : ArrayList
- professors: ArrayList
- staff : ArrayList
- visitors : ArrayList
+ getConnection( )
+ fill_book ( )
+ fill_magazine ( )
+ fill_cd ( )
+ fill_student ( )
+ fill_professor ( )
+ fill_staff ( )
+ fill_visitor( )
+ addBook( )
+ addMagazine( )
+ addCD( )
+ deleteBook( )
+ deleteMagazine( )
+ deleteCD( )
+ addStudent( )
+ addProfessor ( )
+ addStaff( )
+ addVisitor( )
+ deleteStudent ( )
+ deleteProfessor ( )
+ deleteStaff ( )
+ deleteVisitor ( )
+ displayBooks ( )
+ displayMagazines( )
+ displayCDs( )
+ displayStudents( )
+ displayProfessors( )
+ displayStaff( )
+ displayVisitors( )
has a
has a
has a
has a
<<abstract>>
WorKSpaceUser
- name : String
- userId : int
+ getName()
+ setName()
+ getUserId()
+ setUserId()
is a
is a
Visitor
- purpose : String
- issn : String
+ getPurpose()
+ setPurpose()
+ equals()
Student
- academicNumber : int
- issn : String
+ getAcademicNumber()
+ setAcademicNumber()
+ equals()
is a
is a
Staff
- position : String
- issn : String
+ getPosition()
+ setPosition()
+ equals()
Professor
- department : String
- issn : String
+ getDepartment()
+ setDepartment()
+ equals()