# Assignment 5: Software Design

## 100 points (Individual)

This assignment aims to identify applications of design pattern.

References:

* Lecture 11/12 Design patterns

Instructions

This assignment is an individual assignment. In this assignment, you need to add a design pattern to your term project.

1. Discuss with your teammates and pick an area. Ideally, you should pick a **functional** story from Sprint 2 that you are working on or a story from Sprint 1 you have worked. There cannot be any overlap (i.e., two different members cannot work on the same story).
2. Identify a design pattern that can help you improve the design of your code. Apply that pattern to your code. In the class, we have discussed ten design patterns. For a more comprehensive list, you can check: <https://www.geeksforgeeks.org/software-design-patterns/>
3. Create a new-branch named ({your\_name}\_homework4) and commit the changes that introduce a pattern to your term project. You commit message should also include the name of the pattern that you have applied.
4. To ensure that your introduction of pattern does not break the project, it is not necessary to integrate this commit to the master branch. If you can do that integration without breaking your project, that’s excellent. If you can’t due to conflicts, its fine with us.

Submission

In the canvas submit a report, the report must include.

1. A UML class diagram that shows how you applied the design pattern in your term project. The class name, method name, attributes must be the names that you have used in your project. We would verify the code as well as the diagram to make sure that you have correctly introduced the design pattern. Do not forget to include the name of the pattern in the caption.
2. Provide a brief explanation of how introduction of this pattern have helped your term project design.
3. Provide a brief explanation of consequences due to the application of this pattern.
4. Include the branch name and URI of the commit where you have introduced the pattern. For example a direct link to commit would be like <https://github.com/WSU-4110/PyMerge/commit/699b23141be7fd840fbe61c856bff5d0f6c771fb>

**Singleton**:

Private static DbConnection instance = null;

Private SQLConnection connection;

Private DbConnection(){

Connection = connectToDatabase(organization, password,);

}

Public static DbConnection getDbConnection(){

If(instance == null){

Instance = new DbConnection();

Return instance;

}

}

dbConnection connection = DbConnection.getDbConnection;

System.out.println("Registration Page");

System.out.printf("Organization: ");

String user = input.next();

System.out.printf("Password: ");

String pass = input.next();

System.out.printf("Confirm Password: ");

String conf = input.next();

int length = pass.length();

int passInt = Integer.parseInt(pass);

int confInt = Integer.parseInt(conf);

try

{

System.out.println("You registered succesfully");

x.close();

}

catch(Exception e)

{

e.printStackTrace();

}

}

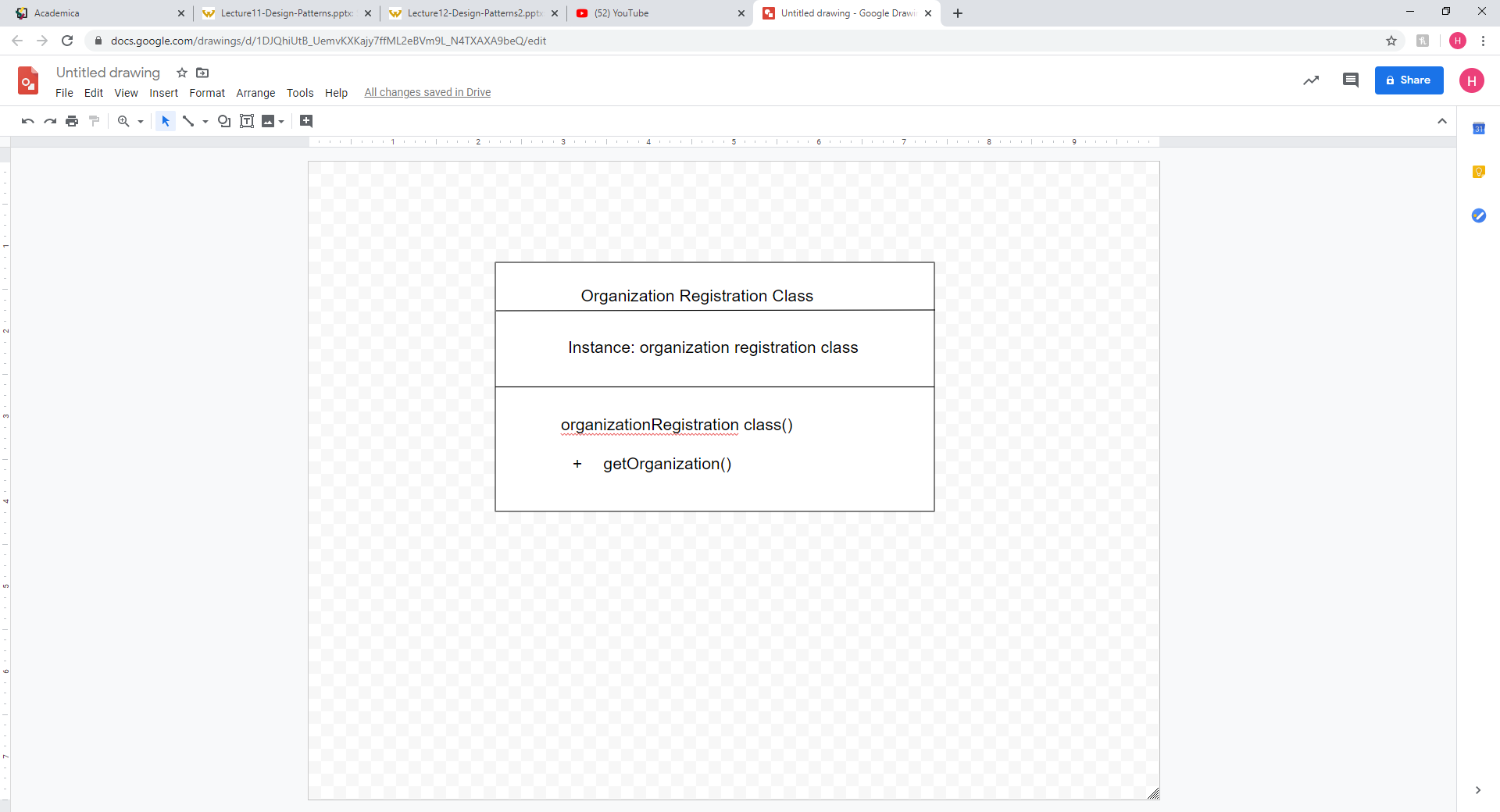
}catch(Exception e){}

}

else

System.out.println("Password and confirm password are not matching");

}



**Benefits:**

This pattern has helped us by keeping everything organized throughout all our code. Also, it ensures each class has only one instance, and provide a global point of access to it.

**Disadvantage:**

Application needs one, and only one, instance of an object. Additionally, lazy initialization and global access are necessary.