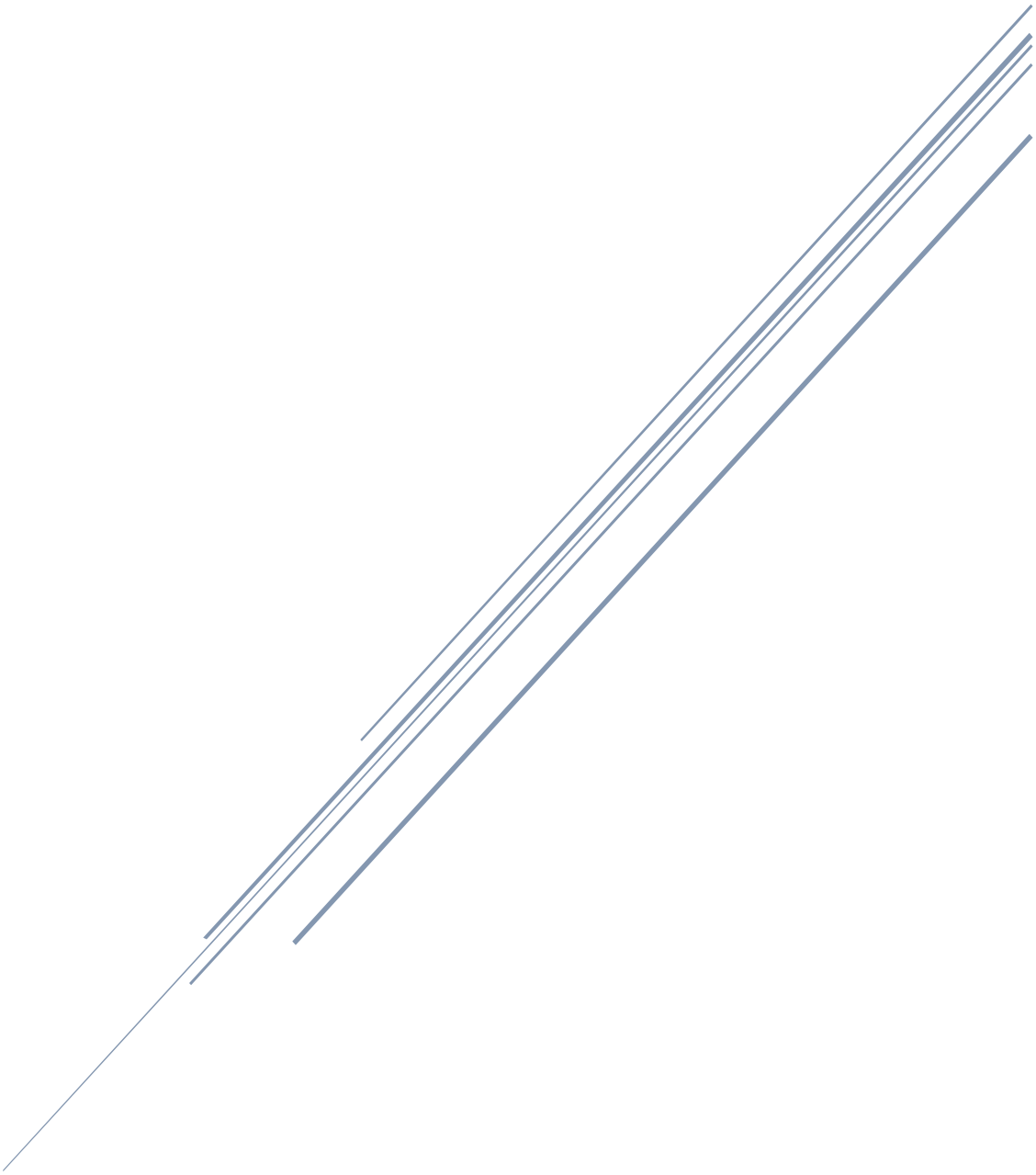
ST10084621

PROG7311 – POE Part 1 Resubmission



Course: BCAD Year 3 – Group 1

Due Date: 23 June 2023

Contents

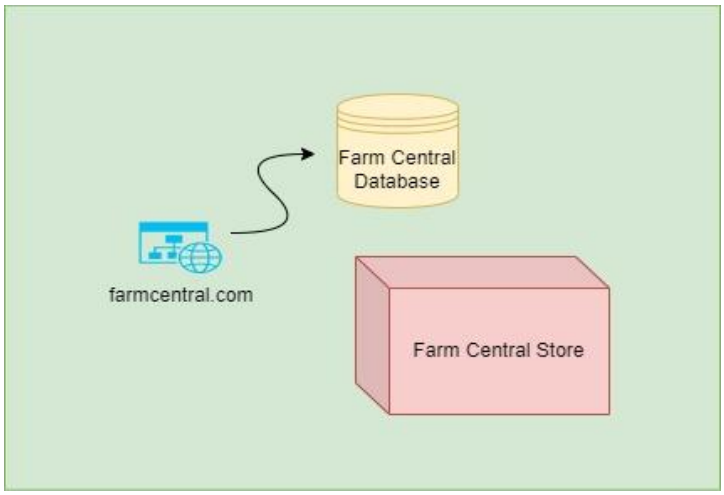
Part 1 – Requirements and Design Patterns

Intro:

A stock management website will be developed for Flesh Farming, in order to track incoming and outgoing stock of farmers, this report contains the contents of the proposed system.

Abstract

Flesh Farming is a store that sells meat products. They currently have a brick and mortar store and their systems are outdated. A website will be created in which the users of it will be able to perform management of stock. Employees can add Farmers to the database of Flesh Farming and Farmers can add products to their profiles. Only users that are registered with Flesh Farming may be granted access to this website.



1. Non-functional requirements of high-importance:

Data integrity

- The system will be used to securely store its data in a cloud-based database.
    - This will be done so that the integrity of all the data that needs to be stored by a user remains intact, accurate and unaltered.
    - Only a logged in user can add data to the database.
    - This requirement impacts how the system is developed by making ensuring that the data that needs to be captured by its users are stored in a secure environment. (Yesin, et al., 2021)

Usability

- The system will be easy to use for farmers.
    - The users of the system will be able to easily navigate through the system and understand the purpose of each webpage in order to make full use of the website.

- The user interface will be designed in a manner that is visually pleasant and easy to input data by users.
- This requirement impacts how the system is developed by ensuring that the system is easy to use and highly functional (Zowghi, 2010).

Scalability

- The system will be easily learnt how to be used by the many employees who are trained to interact with it.
  - This is done so that the employees that interact with the system can easily handle their workload.
  - A demo video on how to use the system will viewed by new employees to gain an understanding of how to interact with the system.
  - This impacts how the system is developed by ensuring that the system can be used by the many employees of Farm Central.

Accessibility

- The system will be designed in a manner that is user friendly to users with disabilities.
  - This is done so that users with disabilities are actually able to use the system without being negatively affected.
  - There will not be any flashing lights or high-contrast images throughout the website so that users with epilepsy can use the system.
  - This impacts how the system is developed by keeping the system accessible to users that have disabilities.

Security

- Users have to be logged into the system to access the contents of it.
  - This is done so that only authorized users have access to the system.
  - Login details of each user will be stored in a cloud-based database that which will be used to verify the login of a user on the system.
  - This impacts how the system is developed by ensuring that only users that are meant to be using the system have access to do so (Zowghi, 2010).

```
// seeding admin (SUPER USER)
using (var scope = app.ApplicationServices.CreateScope())
{
    // create userManager, scope is used to access services
    var userManager =
    scope.ServiceProvider.GetRequiredService<UserManager<IdentityUser>>();

    string email = "admin@admin.com";
    string password = "Test1234,";

    if (await userManager.FindByEmailAsync(email) == null)
    {
        var user = new IdentityUser();
        user.UserName = email;
        user.Email = email;

        await userManager.CreateAsync(user, password);

        // add user to role
        await userManager.AddToRoleAsync(user, "Admin");
    }
}
```

2. Design patterns and architecture patterns

Relevance:

Design and architecture patterns are relevant because they help to reduce the recurring problems associated with developing high quality software (Abufakher, 2020).

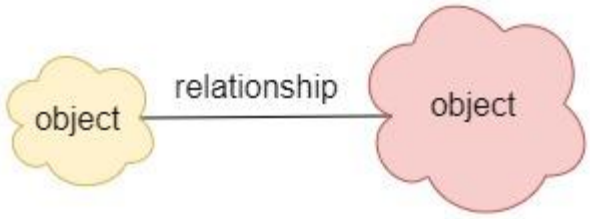
Design patterns

What are design patterns? These are solutions to problems that are encountered quite often when designing software. Design patterns show how classes should be structured and how those classes can interact with each other.(Abufakher, 2020).

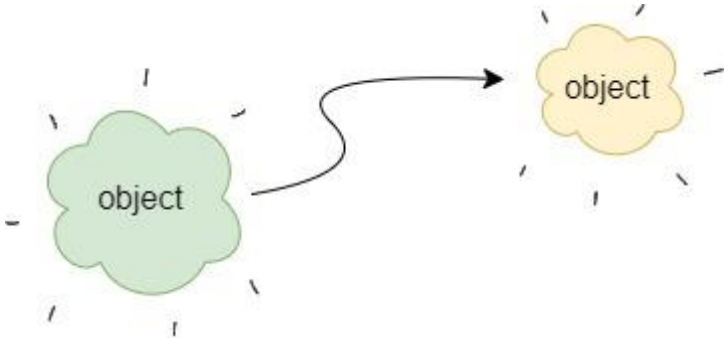
There 3 categories of design patterns:

- **Creational patterns:** Different ways to create objects.

- **Structural patterns:** The relationship between these objects.



- **Behavioural patterns:** The interaction between these objects.



Singleton Pattern

- A way of creating a single object that is shared amongst the resources throughout the application without having to recreate the object and provides a single access point to it. This object is accessible from anywhere in the application (Sharma, 2022). The Singleton pattern can be used in this project by including a class with methods and variables that can be used throughout the various classes of the program.

```java
// main class
public class FarmCentral {
    // Farmer f = new Farmer();
    // above code is impossible because of private constructor

    // instead create the instance of the class by using the method
    // created in the Singleton class
    Farmer f = Farmer.getInstance();
}

// Singleton class
5 usages
class Farmer {

    // create static variable of Singleton class
    1 usage
    static Farmer f = new Farmer();

    1 usage
    private Farmer()
    {
        // create private constructor of Singleton class
    }

    // create method that returns instance of the Singleton class
    1 usage
    public static Farmer getInstance()
    {
        return  f;
    }
}
```

e.g. The Farmer class is the Singleton class, everything in this class can be used throughout the application.

Architecture patterns

Model-View-Controller pattern:

This is the architectural pattern that will be used to create the website. This pattern separates the project into three parts:

1. Model – includes the data objects of the project (like a database).

2. View – this is the layout of the webpages that will be viewed by the user.
3. Controller – this part receives input from the user on the website and then interacts with the model to make a particular view appear to the user (Mallawaarachchi, 2017).

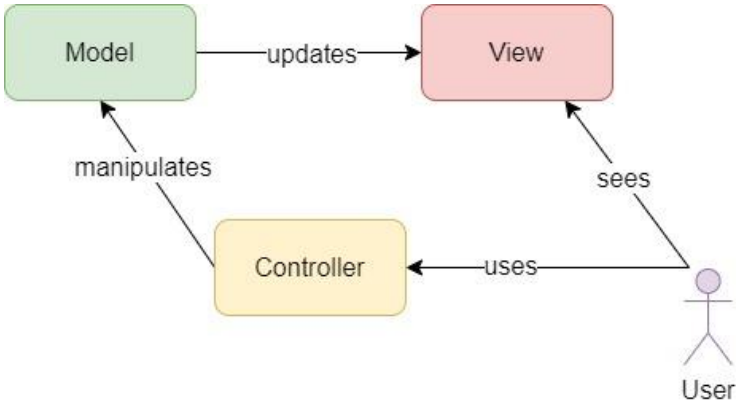


Diagram created on Draw.io

In the diagram above, the user will interact with the browser (e.g. click on a link) and the URL will be handled by the controller. The controller will then manipulate the model in order for a webpage to appear to the user using the view. In this project, the website will be designed using the MVC pattern.

Conclusion

The above document is a fully discussed and analysed report about requirements and design patterns.

Bibliography

Abufakher, S., 2020. Impact of design patterns on software quality: a systematic literature review. *Impact of design patterns on software quality: a systematic literature review,* 14(1), pp. 1-17.

Mallawaarachchi, V., 2017. *towardsdatascience.com.* [Online]
Available at: https://tinyurl.com/28wsk66a
[Accessed 16 April 2023].

Sharma, A., 2022. *javadevjournal.com.* [Online]
Available at: https://tinyurl.com/j8xw5e3u
[Accessed 16 April 2023].

Yesin, V. et al., 2021. Ensuring Data Integrity in Databases with the Universal Basis. *Ensuring Data Integrity in Databases with the Universal Basis,* 1(1), pp. 1-14.

Zowghi, D., 2010. An Investigation into the Notion of Non-Functional Requirements. *An Investigation into the Notion of Non-Functional Requirements,* pp. 311-317.