**CZ3002 Assignment 2 Report**

Tan Jun Hong [b190013@e.ntu.edu.sg](mailto:b190013@e.ntu.edu.sg)

Tan Leng Hwee Gordon [GTAN070@e.ntu.edu.sg](mailto:GTAN070@e.ntu.edu.sg)

# 

[**Model View Controller Framework** 2](#_Toc69412126)

[**Architecture Diagram and Explanation in MVC Perspective** 2](#_Toc69412127)

[**Execution Flow with Sequence Diagram** 3](#_Toc69412128)

[**Dynamic Binding for Maintenance Change** 4](#_Toc69412129)

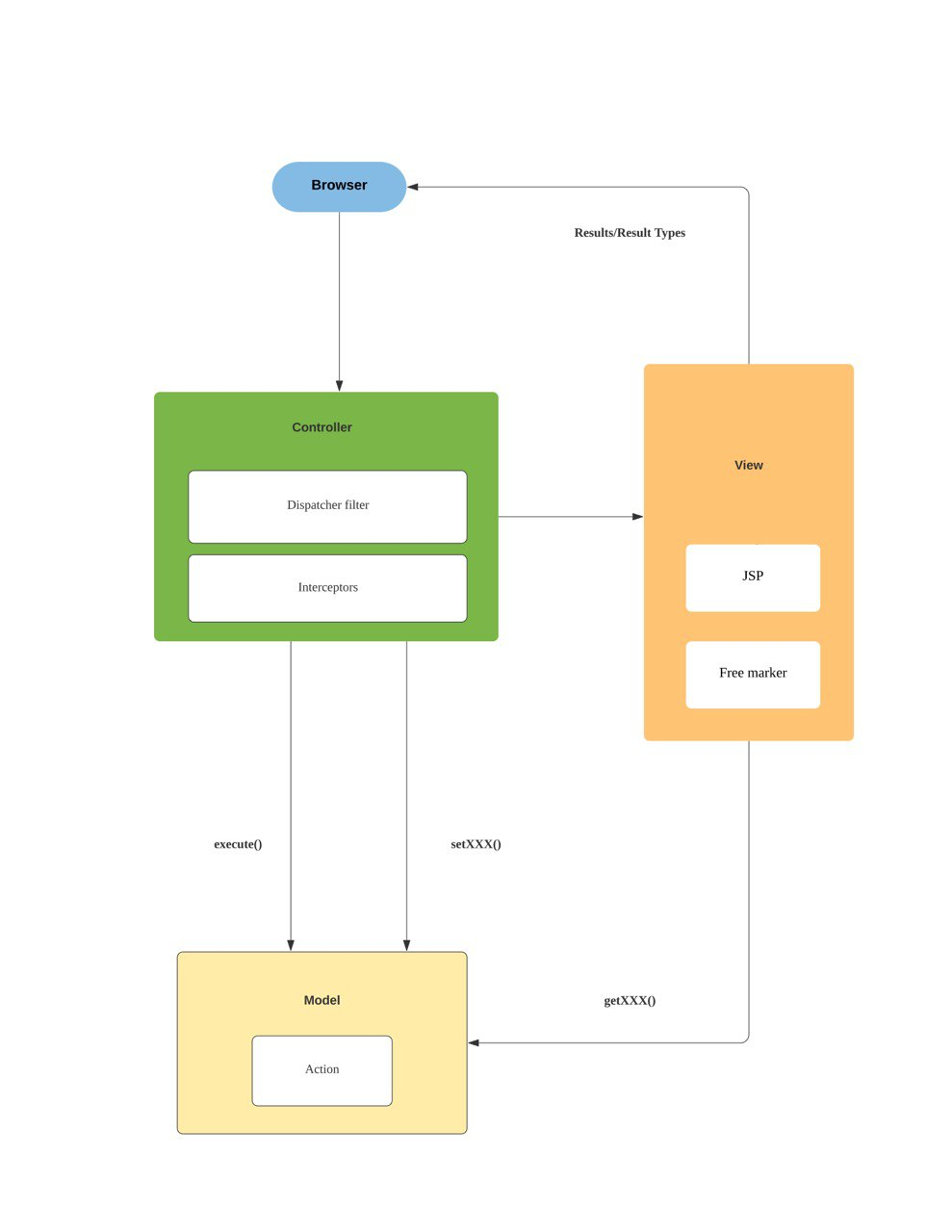
[**Organization of the Code, Installation Manual and User Manual** 4](#_Toc69412130)

## **Model View Controller Framework**

MVC framework consists of 3 parts: model, view, and controller.

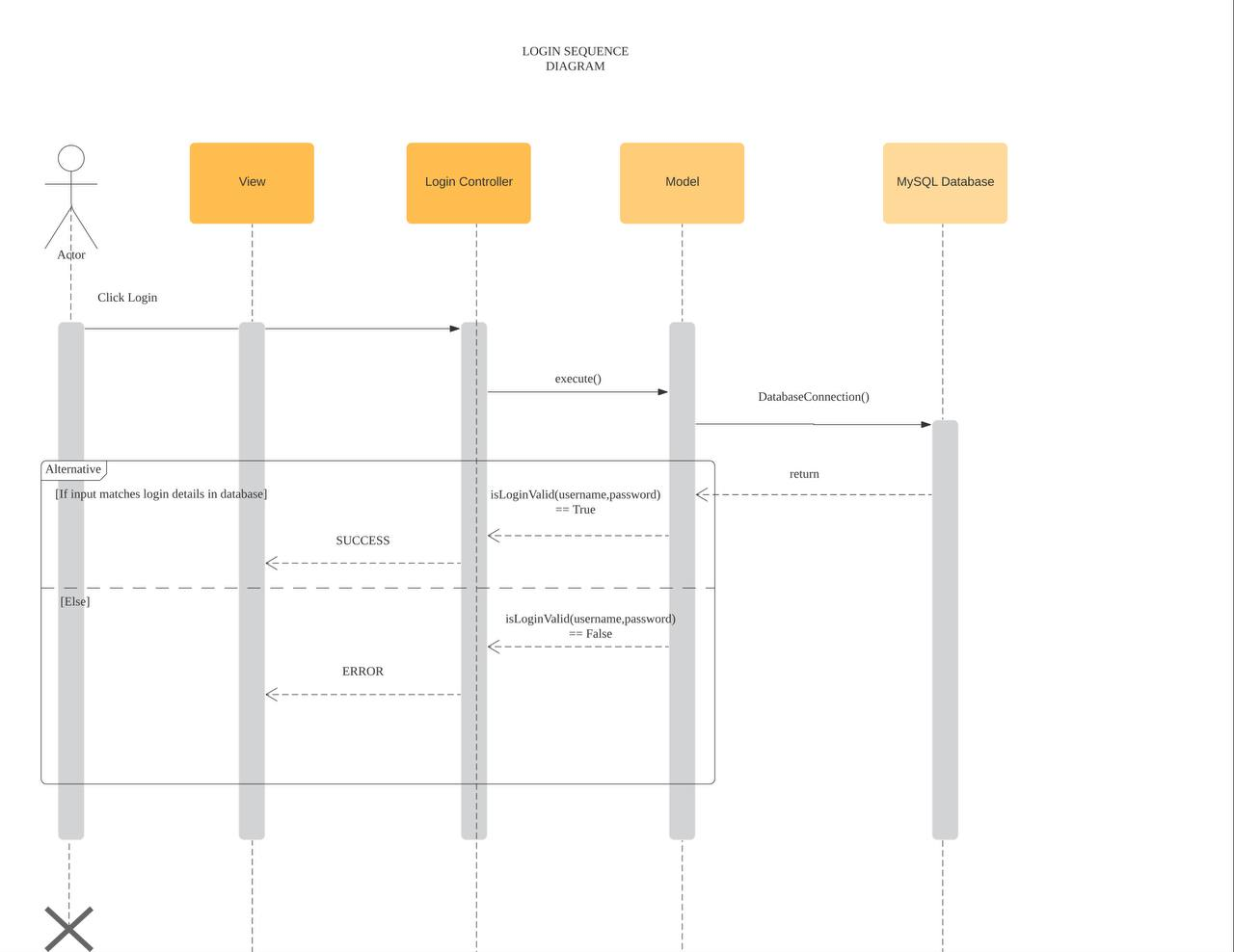
* The controller processes the requests’ parameters and decides which function should be called for processing or which view should be selected for the presentation of data.
* The model computes and processes the data to produce the desired output needed based on the user’s request.
* The view renders the presentation of what the model has computed based on the given request made by the user.

## **Architecture Diagram and Explanation in MVC Perspective**

****

* Model: Strut 2 has action classes that will process the requests and perform the required action to produce the required output.
* Controller: The dispatcher filter and interceptors are part of the controller for the strut 2 framework. The interceptors are responsible for handling user parameters while the dispatcher filter is responsible for routing the requests using xml configuration to the correct components i.e., view or controller.
* View: Strut 2 uses existing Java technologies such as JSP and FreeMarker that have tag libraries that are inbuilt libraries that assist in setting the webpage that the user will see and interact with.

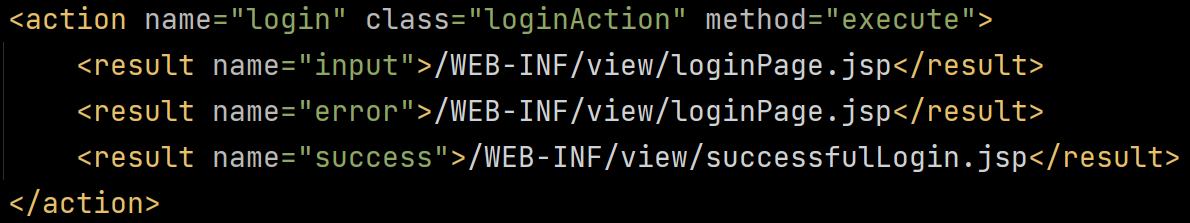
## **Execution Flow with Sequence Diagram**



1. After the user clicks on the login button, this action will initiate a request from the server that consequently calls execute() from the login controller.
2. After this, these parameters will be passed to the model. User stores the user’s information, while DatabaseConnection stores the database connection.
3. Afterwards, the LoginController filters and passes the parameters to the respective model(s). In this case, username and password are passed to both User (for creation of user class) and then DatabaseConnection (for verification with the database).
4. As discussed above, the model classes will be the ones that process the request that was initiated by the user before passing the updated information back to the controller. After which, the controller will route this result via xml back to the supposed view classes and reflect the updated changes back to the JSP file. The user will be able to see the status as a result (Successful/Unsuccessful).

## **Dynamic Binding for Maintenance Change**



****

It is easy to make changes. For example, we create a FingerprintLoginAction class that extends LoginAction and overrides the method execute. Then in the applicationContext.xml file, change the wiring to ase.group17.controller.FingerprintLoginAction instead of ase.group.controller.LoginAction. Since the method execute is overridden, there is dynamic binding, where the binding happens at runtime. This will ensure that the new login class/function will be used instead, and other parts of the code can be reused without changes, ensuring reusability, modularity, and easy extension of code.

## **Organization of the Code, Installation Manual and User Manual**

|  |  |
| --- | --- |
|  | The code is organised as follows:   * Controller package in src contains LoginAction; * Model package in src contains DatabaseConnection and User; * View package in webapp contains loginPage and successfulLogin   This allows easy visualisation of the MVC framework.  The mysql connector jar is required to connect to the database and is located at the resources package.  The applicationContext.xml contains the wiring, while the struts.xml contains the actions. |

Setup

1. To run the code, make sure maven is installed and the environment variables are set.
2. Follow the guide here to setup MySQL - [MySQL :: MySQL Installation Guide](https://dev.mysql.com/doc/mysql-installation-excerpt/8.0/en/)
3. Navigate to the DatabaseConnection class and edit the constants at the top of the page accordingly to connect to the database.
4. A file called ‘ScriptsToRun.sql’ has been included to populate the database.

Run

* Firstly, open terminal/command prompt and navigate to the project root.
* Then, type ‘mvn jetty:run’ in the command prompt.
* Open browser and go to <http://localhost:8080/CZ3002%20Assignment%202>