# CHAPTER 4
# DESIGN ANALYSIS

## 4.1 Server

### 4.1.1 Server Class

The server will have  three classes that have different tasks. Server can run on ip address and flexible port. Server open socket to receive connection from clients. This server uses a thread to handle the client connection. Each client is connected, the client gets a new thread controlled by the Server Class.

### 4.1.2 RequestHandler Class

After receiving a connection from the client, all of request is handled by the RequestHandler Class. This class uses a thread to handle the request. Every request from the client, get a new thread. Each message sent by the client, is handled by the server and separated by a whitespace. There are three data sent by the client that is access key, sensor identity and value. The protocol format used by the client to send data is :

**{[ACCESS_KEY] [SPACE] [SENSOR IDENTITY] [SPACE] [VALUE]}**

Access key is a unique identity of the channel. One user can have multiple channels. Each channel has different access keys. Access keys serve the final destination identity of data transmission, so the data sent does not exchange. The composition of the access keys is the result of character randomization consisting of a combination of email addresses, passwords, account creation dates, channel identity and random numbers from 10 to 1000000.

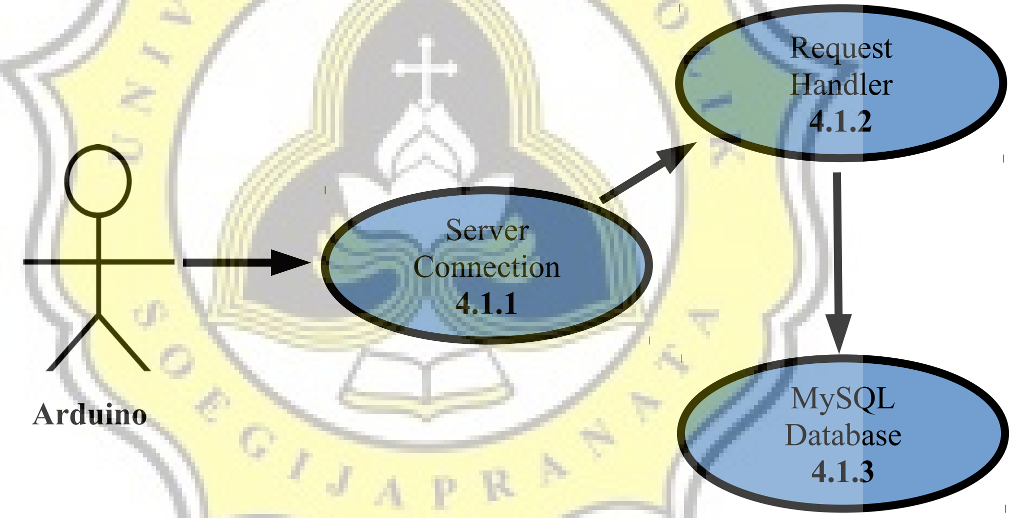
Sensor identities are used to differentiate sensors that are included in the channel. One channel can have multiple sensors. The sensor identity allows the sensor data trasmitted to the channel does not exchange.

The sensor value used is the decimal number.

### 4.1.3 MySQLConnection Class

After receiving message from the client and separated by the whitespace, data is forwarded to the database based on access keys. All of data entry proccesses to the database are handled by the MySQLConnection Class. The library used by the MySQLConnection Class is JDBC for MySQL database management system. All messages that have been separated, are stored to the database based on access keys.

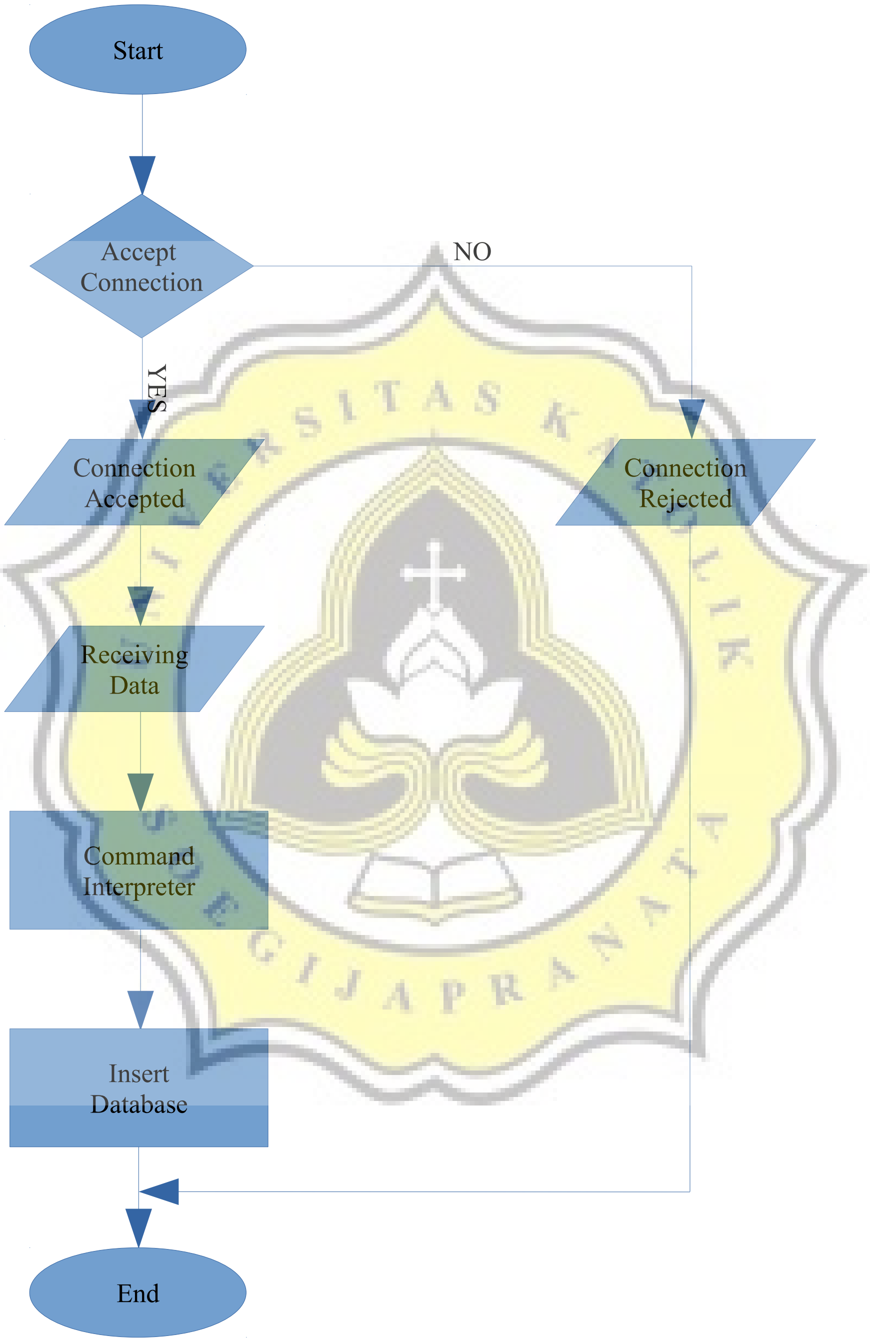
### 4.1.4 Use Case Diagram



Information :

Association →

### 4.1.5 Flow Chart



### 4.1.6 Class Diagram

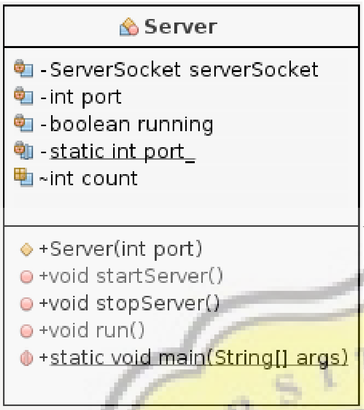


Illustration 1: 4.1.1

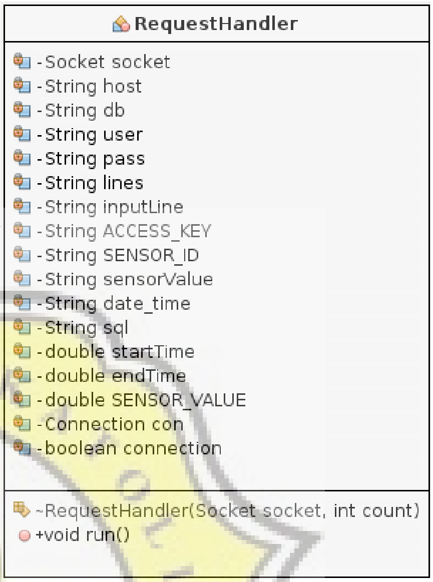


Illustration 2: 4.1.2

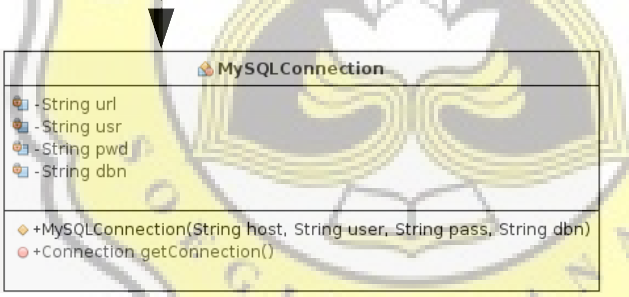

MySQLConnection
-String url
-String usr
-String pwd
-String dbn
+MySQLConnection(String host, String user, String pass, String dbn)
+Connection getConnection()

Illustration 3: 4.1.3

## 4.2 Front-End Application

### 4.2.1 Proses Class

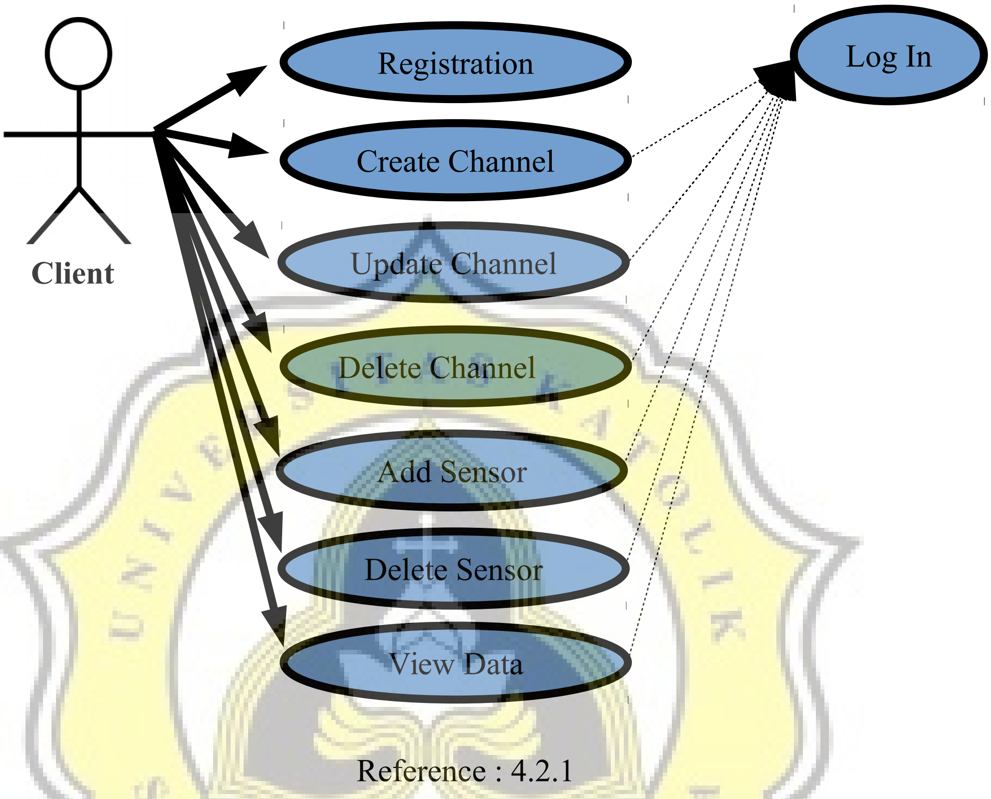
Front-end applications are required for users to use this platform. This front-end application is web based. This front-end application is created with object oriented PHP programming language. All of CRUD (Create Read Update Delete) processes are handled by Proses Class.

The CRUD process in this project are :

1. User registration
2. Create channel
3. Delete channel
4. Update channel information
5. Add sensor
6. Delete sensor
7. Update profile information
8. Show data in tables and graph

For connecting database in front-end application using PDO (PHP Data Object). Sensor data is displayed in the real time line graph. For graph libraries using chartist. All of sensors data is deleted every month from the date of registered users.

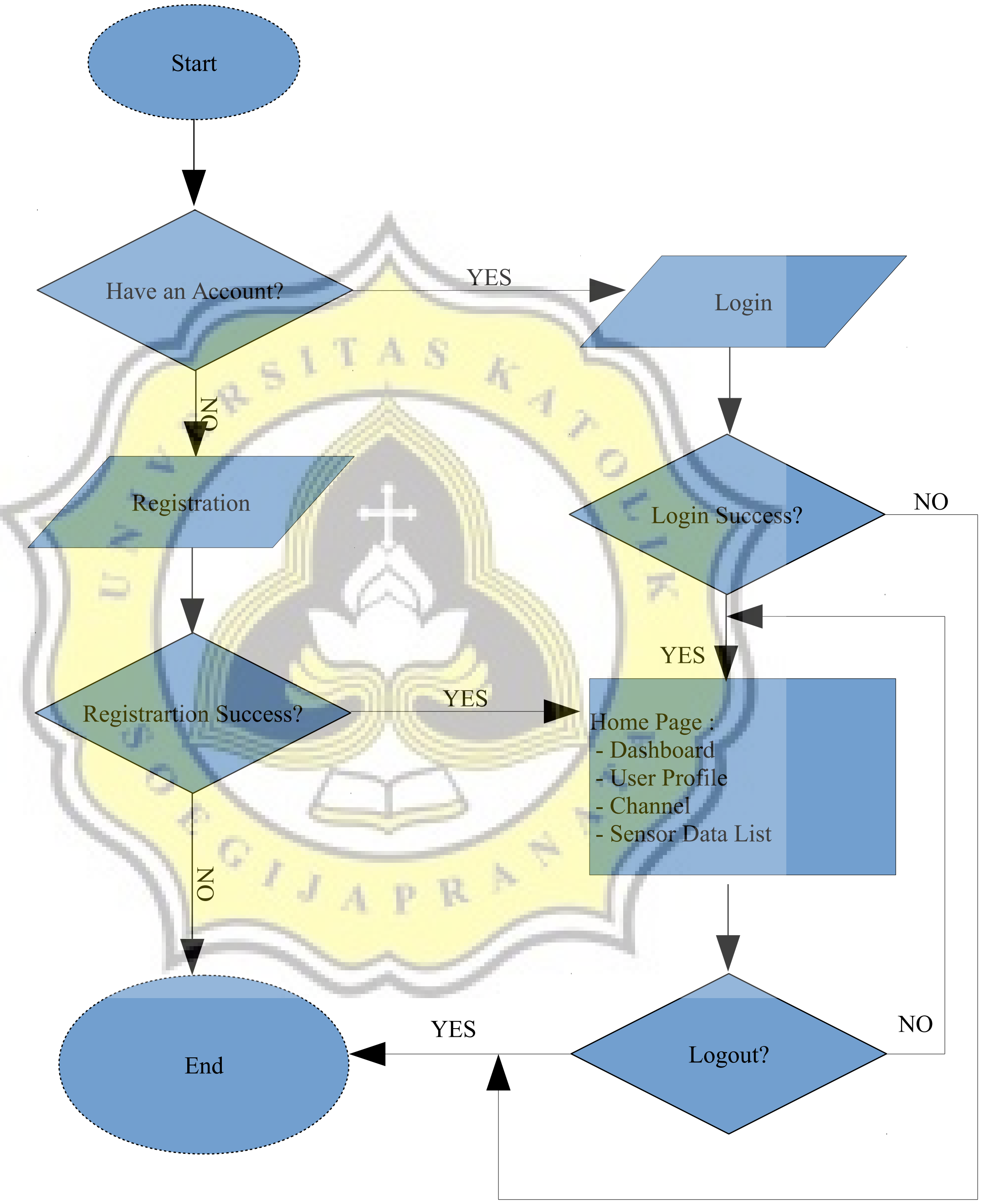
### 4.2.2 Use Case Diagram



Information :

Association
<<include>>

### 4.2.3 Flow Chart



## 4.3 Testing

There are two testing in this project. The testing are :

1. Server endurance

   To test endurance of the server, there are two artificial programs for simulating data attacks from the client.

   A) The first program, connection and data transmission in the same class

   B) The second program, connection and data transmission in the different classes.

   The data on this testing is attacked by a hundreds, thousands or billions of data. The tested server using three computers with different spesifications.

2. Packet data size

   To test the size of the data packets managed by the server in this testing, the test is done by performing manual calculations of characters sent from the device.