CITY COMPLAINT SYSTEM

Name: Chan Khai Shen

Coventry Student ID: 11323943

Supervisor Name: Mr. Light Lau

Bachelor of Science with Honours in Computer Science, 3+0 in Collaboration with Coventry University, UK

School of Engineering and Technology, INTI International College Penang

10th November 2023

6001CEM Declaration of Originality

I declare that this project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detecti.

Statement of copyright

I acknowledge that the copyright of this project report, and any product developed as part of the project, belong to Coventry University. Support, including funding, is available to commercialise products and services developed by staff and students. Any revenue that is generated is split with the inventor/s of the product or service. For further information please see www.coventry.ac.uk/ipr or contact ipr@coventry.ac.uk.

Statement of ethical engagement

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (https://ethics.coventry.ac.uk/) and that the application number is listed below (Note: Projects without an ethical application number will be rejected for marking)

Signed: Date:

Please complete all fields.

First Name:	Chan
Last Name:	Khai Shen
Student ID number	11323943
Supervisor Name	Mr. Light Lau

This form must be completed, scanned and included with your project submission to Turnitin. Failure to append these declarations may result in your project being rejected for marking.

Table of Contents

Table of Contents iii
List of Figures vi
List of Tables viii
List of Abbreviations ix
Abstract x
1. INTRODUCTION 1
1.1 Problem Statement 1
1.2 Detailed Research Question 2
1.3 Objective 2
1.4 Client, Audience and Motivation 2
1.5 Scope and Brief Description of the Proposed Work 3
2. LITERATURE REVIEW 4
2.1 E-Complaints MBPP (Penang Island, Penang, Malaysia) 4
2.1.1 Complainant registration 4
2.1.2 Lodge complaint 5
2.1.3 Check complaint status 10
2.2 Adu@KL (Kuala Lumpur, Malaysia) 11
2.2.1 Submit feedback 12
2.2.2 Feedback enquiry 14
2.3 Public Complaint System (Conceptual system) 14
2.3.1 Lodge complaint 15
2.3.2 Change complaint status 15
2.3.3 Security aspects of the application 16
2.4 Statistical reporting in complaint system 16
2.4.1 Electronic-Customer Complaint System (Reporting in chart) 16
2.4.2 Online Complaint Management System (Reporting in simple dashboard) 17
2.5 Critical analysis 17
2.5.1 Summary of literature review 17
2.5.2 Comparison of the proposed work and the existing and similar systems 20
3. RESEARCH METHODOLOGY 22
3.1 Waterfall Development Model 22
3.1.1 Requirement analysis 22
3.1.2 System design 23

3.1.3 Implementation 24
3.1.4 Testing 24
3.2 Gannt Chart 24
3.3 Use Case Diagram 25
3.3.1 Complainant 25
3.3.2 Complaint handler 28
3.3.3 Administrator 30
3.4 Entity Relationship Diagram 32
3.5 Libraries 32
3.5.1 Express 33
3.5.2 Axios 33
3.5.3 Body-Parser 33
3.5.4 JSON-Web-Token 33
3.5.5 Mongoose 33
3.5.6 Crypto 34
3.5.7 MD5 34
3.5.8 Node Mailer 34
3.5.9 React-Router-Dom 34
3.5.10 React-Chartjs-2 34
4. IMPLEMENTATION 35
4.1 Brief Explanation on the Structure of the System 35
4.2 Registration 35
4.2.1 Register complainant 35
4.2.2 Add or delete complaint handler 39
4.2.3 Set / reset password 42
4.2.4 Create administrator 44
4.2.5 Password hashing and salting 45
4.3 Login 45
4.3.1 Access control before login 46
4.3.2 Login page and landing page 47
4.4 Lodge Complaint (Available to complainant only) 49
4.4.1 Lodge complaint page 50
4.4.2 Administrator add / delete the area and category options 52
4.5 Check Complaint 53
4.5.1 Check complaint page 53
4.5.2 Search, filter and sort 54

4.5.3 Complaint details page 55
4.5.4 Histories section 56
4.5.5 Add message 57
4.5.6 Change status 59
4.5.7 Assign complaint handler to a case (Complaint handler / Administrator) 61
4.6 Statistics Reporting 62
4.6.1 Specify date section and summary of complaint count section 62
4.6.2 Pie chart section 63
4.6.3 Stacked bar chart section 63
4.6.4 Case count section 64
4.7 Code Snippet 65
4.7.1 Email verification through one-time password (OTP) 65
4.7.2 Statistics reporting 66
5. Testing 70
5.1 Unit Testing 70
5.2 Integration testing 70
5.3 Security testing 71
REFERENCES 72
6. Appendix A 75

List of Figures

Figure 2.1 E-Complaints MBPP Complainant registration page (MBPP, n.d.) 5
Figure 2.2 E-Complaints MBPP New complaint page (MBPP, n.d.) 6
Figure 2.3 E-Complaints MBPP Problem description page (MBPP, n.d.) 8
Figure 2.4 E-Complaints MBPP Confirm complaint details page (MBPP, n.d.) 9
Figure 2.5 E-Complaints MBPP Add attachment page (MBPP, n.d.) 10
Figure 2.6 E-Complaints MBPP View complaint page (MBPP, n.d.) 11
Figure 2.7 Adu@KL Fill in feedback details page (DBKL, n.d.) 12
Figure 2.8 Adu@KL Fill in personal details page (DBKL, n.d.) 13
Figure 2.9 Adu@KL Feedback enquiry page (DBKL, n.d.) 14
Figure 3.1 Waterfall development model (Tutorials Point, n.d.) 22
Figure 3.2 Work load break down (final) 23
Figure 3.3 GitHub Repository 24
Figure 3.4 Gannt Chart (Apr 2023) 24
Figure 3.5 Gannt Chart (Aug 2023) 25
Figure 3.6 Use case diagram (Complainant) 27
Figure 3.7 Use case diagram (Complaint handler) 29
Figure 3.8 Use case diagram (Administrator) 31
Figure 3.9 Entity relationship diagram 32
Figure 4.1 Brief illustration of project structure 35
Figure 4.2 Complainant registration flow chart 36
Figure 4.3 Register complainant 37
Figure 4.4 Register complainant (after OTP sent) 37
Figure 4.5 Verification email 38
Figure 4.6 Registration done 38
Figure 4.7 General flow of complaint handler registration 39
Figure 4.8 Add complaint handler flow chart (step 1) 40
Figure 4.9 Delete complaint handler flow chart 41
Figure 4.10 Add / delete complaint handler 42
Figure 4.11 Set / Reset password flow chart 43
Figure 4.12 Set password page 43
Figure 4.13 Set password page (after OTP sent) 44
Figure 4.14 Create administrator 45
Figure 4.15 Front-end access control 46
Figure 4.16 "/verifytoken" get method 47
Figure 4.17 verifyToken() function (server-end) 47
Figure 4.18 Login page 48
Figure 4.19 Access control of different user 48
Figure 4.20 Landing page after login 49
Figure 4.21 Lodge complaint flow chart 50
Figure 4.22 Lodge complaint page 51
Figure 4.23 Complaint summary page 51
Figure 4.24 Complaint summary (email) 52
Figure 4.25 Add category / area section 53
Figure 4.26 Check complaint page 54

Figure 4.27 Search, filter and sort options 55
Figure 4.28 Administrator search option 55
Figure 4.29 Complaint details page 56
Figure 4.30 Histories section 57
Figure 4.31 Add message flow chart 58
Figure 4.32 Add message bar 58
Figure 4.33 Notification email (New message from complaint handling side) 58
Figure 4.34 Change status flow chart 60
Figure 4.35 Change status button 60
Figure 4.36 Order of change of case status 60
Figure 4.37 Notification email (Change status) 61
Figure 4.38 Choose complaint handler (Administrator view) 62
Figure 4.39 Statistics reporting page 62
Figure 4.40 Specify range of date and display summary of complaint count section 63
Figure 4.41 Pie chart section 63
Figure 4.42 Stacked bar chart section 64
Figure 4.43 Case counts section 64
Figure 4.44 OTP Schema 65
Figure 4.45 Generate OTP 66
Figure 4.46 Verify OTP 66
Figure 4.47 Generate grouped data 67
Figure 4.48 Front end handing of grouped data 68
Figure 4.49 Rendering chart 69
Figure 5.1 Integration testing 71
Figure 5.2 Security testing 71

List of Tables

Table 2.1 Complaint description fields 18
Table 2.2 Personal information fields 19
Table 2.3 Feature comparison 20
Table 5.1 Test case 70

List of Abbreviations

Abbreviation	Full form
DBKL	Kuala Lumpur City Hall (*Dewan Bandaraya Kuala Lumpur*)
E-CCMS	Electronic-Customer Complaint System (System reviewed)
IC	Identity Card
JWT	JavaScript object notation web token
MBPP	City Council of Penang Island (*Majlis Bandaraya Pulau Pinang*)
MD5	Message Digest algorithm
OCMS	Online Complaint Management System (System reviewed)
OTP	One-Time Password
PBKDF2	Password-Based Key Derivation Function
PCS	Public Complaint System (System reviewed)
SHA-256	Secure Hash Algorithm – 256 bits

Abstract

City council is the government unit that is closest to the citizen, thus having an effective complaint system is important. Statistical reporting in city complaint system can help the person-in-charge of city complaint system to have the overall information of the system, which can help him to do fact-based decisions. To cater these problems, this project will be conducted to develop a conceptual web-based city complaint system based on Penang's landscape and implement statistical reporting in the system. Based on the literature review, this project aims to add a few improvements to the city complaint system, namely email verification through one-time password (OTP), dynamic rendering of options for category and area, email notification on status update, and systematic statistics reporting for administrator. This project is implemented based on the MongoDB – Express.js – React.js – Node.js (MERN) stack.

1. INTRODUCTION

1.1 Problem Statement

City council is the government unit that is closest to citizens' living. The service provided by the city council, such as cleanliness, public facility and basic utility are very much related to citizens. So, citizens can be said as the major group of customers of the city council. So, it is important for the city council to manage citizens' complaint well, so that the city council's image can be retained.

To manage citizen complaint effectively, it is important for the city council to have a city complaint system. A city complaint system can provide a gateway for the citizen to lodge a complaint case when they feel dissatisfied about any services provided the city council. Complaint can also be lodged if there is any misconduct of the city council officer. If the citizen finds that there is any activity that is suspected to be against of the city council's rule and regulation, they can also lodge a complaint. Based on statistics calculated by Statista (2023), the population in Penang (or *Pulau Pinang*) state as of July 2023 is approximately 1.77 million. So, it is important to have a computerised city complaint system to cope with the citizens' complaint cases because the citizen population is considerably large.

A computerised city complaint system is can also produce statistical reporting of the complaint cases in the system. The statistical report, in means of charts and dashboard, depicts the key statistics of the city complaint system. This helps the person-in-charge of the complaint system to get some overall information of the system, such as number of complaints cases going on and break down of the complaint cases by area. By knowing this key information from time to time, the person-in-charge can make a more effective arrangement for complaint resolving. For example, the person-in-charge can arrange more complaint handlers in areas where the number of complaints is higher.

As a solution to the problems above, this project will be conducted to develop a conceptual city complaint system that allows citizens to lodge complaint and check the complaint status. In addition, this project will also implement statistical reporting at the back end of the city complaint system.

1.2 Detailed Research Question

1. How can a city complaint system be developed?

2. How can statistical reporting be implemented in a city complaint system?

1.3 Objective

1. To develop a web-based city complaint system that allows citizen to lodge complaint and check the complaint status

2. To implement statistical reporting for a city complaint system in means of charts and dashboard

1.4 Client, Audience and Motivation

The clients of the proposed work are:

- Citizen
- Complaint handler
- Person-in-charge of complaint handling team

By having a web-based city complaint system, citizens can send a complaint to the city council easily. They do not have to go to the city council office at working hours. Instead, they can lodge the complaint at their free time. On the other hand, the complaint handler can also handle the complaint case more easily. Instead of the hassle of paper and pen, the complaint handlers can read and reply to complaint case more easily through the web.

The person-in-charge of the complaint handling team is the client for the statistical reporting feature. By having the key statistics of the city complaint system, he can have enough information to make fact-based decisions.

1.5 Scope and Brief Description of the Proposed Work

The proposed work will have 3 types of users, namely complainant, complaint handler and administrator.

Complainant is a citizen who lodges a complaint case. Complainant can only access from the complainant side of the system. Complainant is allowed to lodge a complaint and check the complaint status from the complainant side. Complainant needs to register and login to be able to access the system. Upon registration, the complainant has to provide his email address which will be verified through OTP. This email address is used to receive notification email upon status change of the case. Complainant is also allowed to reply to the reply from the complaint handler.

Complaint handler is a city council officer whose role is handling complaint cases. Complaint handler can only access from the complaint handler side, Complaint handlers must login to gain access. Complaint handler is allowed to read the details of complaint case, reply to complainant and change the status of complaint. Complaint handlers are allowed to change password, but not register and remove account because this are regulated by the administrator.

Administrator is the person-in-charge of the complaint handling team. From technical point of view, administrator is a complaint handler who has higher privilege because the administrator has access to all features available to complaint handlers. On top of that, administrator has administrative roles, such as register new complaint handler and remove complaint handler, as well as managing the list of categories and areas in the system. Administrator is allowed to see the key statistics of the complaint cases in the system. The number of administrator is not restricted but the creation of administrator is restricted by only can be done by accessing the code or the database.

2. LITERATURE REVIEW

Under this chapter, two existing city complaint systems in Malaysia and two research paper related to complaint system will be reviewed, compared and analysed.

2.1 E-Complaints MBPP[1] (Penang Island, Penang, Malaysia)

E-Complaints MBPP (also called *E-Aduan MBPP*) is city complaint system in Penang Island, a local municipal in Penang, Malaysia. E-Complaints MBPP is operated by City Council of Penang Island (*Majlis Bandaraya Pulau Pinang*, MBPP). E-Complaints MPPP is a web-based system and can be accessed from aduan.mbpp.gov.my.

2.1.1 Complainant registration

In E-Complaints MBPP, complainant can register an account in the system. In E-Complaints MBPP, a registered complainant is called a member (MBPP, n.d.). A member only needs to fill in his personal information one time, which is during registration. Once registered, every time when a member lodges a complaint, he does not need to fill in his personal information again (MBPP, n.d.). Once login, a member can check his complaints according to chronology and implementation status, thus he does not need to remember the complaint reference number (MBPP, n.d.).

Figure 2.1 is the complainant registration page. Upon registration, the complainant will be requested to give a username and password, as well as filling in personal details, such as name, email address, mobile phone number, fax number, telephone number, address, gender, race and user type (MBPP, n.d.). Besides username and password, only name and email address are mandatory fields. As for optional fields, mobile phone number is only meant for people who want to use E-Complaints MBPP SMS service (to check complaint status) (MBPP, n.d.), whereas telephone number, fax number and address are for better communications. Moreover, race, gender and user type are also optional as these fields are meant to know the complainant background and for statistics record purposes (MBPP, n.d.). There are several

[1] aduan.mbpp.gov.my

user types for the complainant to choose. They are individual, association, company, government agency and other.

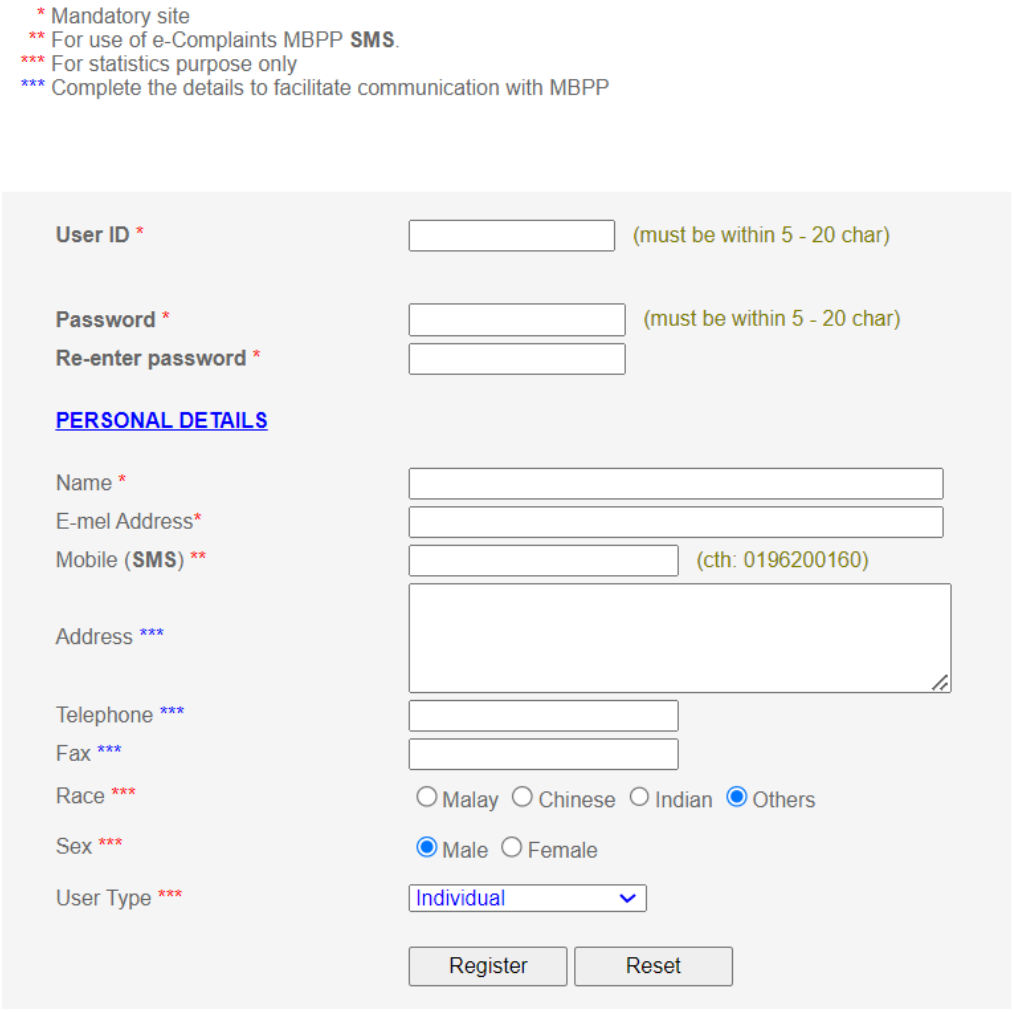


Figure 2.1 E-Complaints MBPP Complainant registration page (MBPP, n.d.)

2.1.2 Lodge complaint

In E-Complaints MBPP, registration is not mandatory for a complainant to lodge a complaint. The difference between registered and non-registered complainants is non-registered complainant needs to fill in his personal details when lodging a complaint, whereas registered complainant does not need to do this (MBPP, n.d.).

2.1.2(a) Fill in personal details

Figure 2.2 is the personal details form for a non-registered complainant to fill in. The fields are similar to the fields for member registration. The information that a non-registered complainant needs to fill in are name, address, identity card (IC) number, email address, mobile phone number, phone number, fax number, race, gender and customer category (MBPP, n.d.). IC number is an additional field for non-registered complainant. Customer category for non-registered complainant is similar to user type for registered complainant. There are several customer categories, which are individual, company, association, government agency and others (MBPP, n.d.). Among all fields, only name, address and mobile phone number are mandatory (MBPP, n.d.).

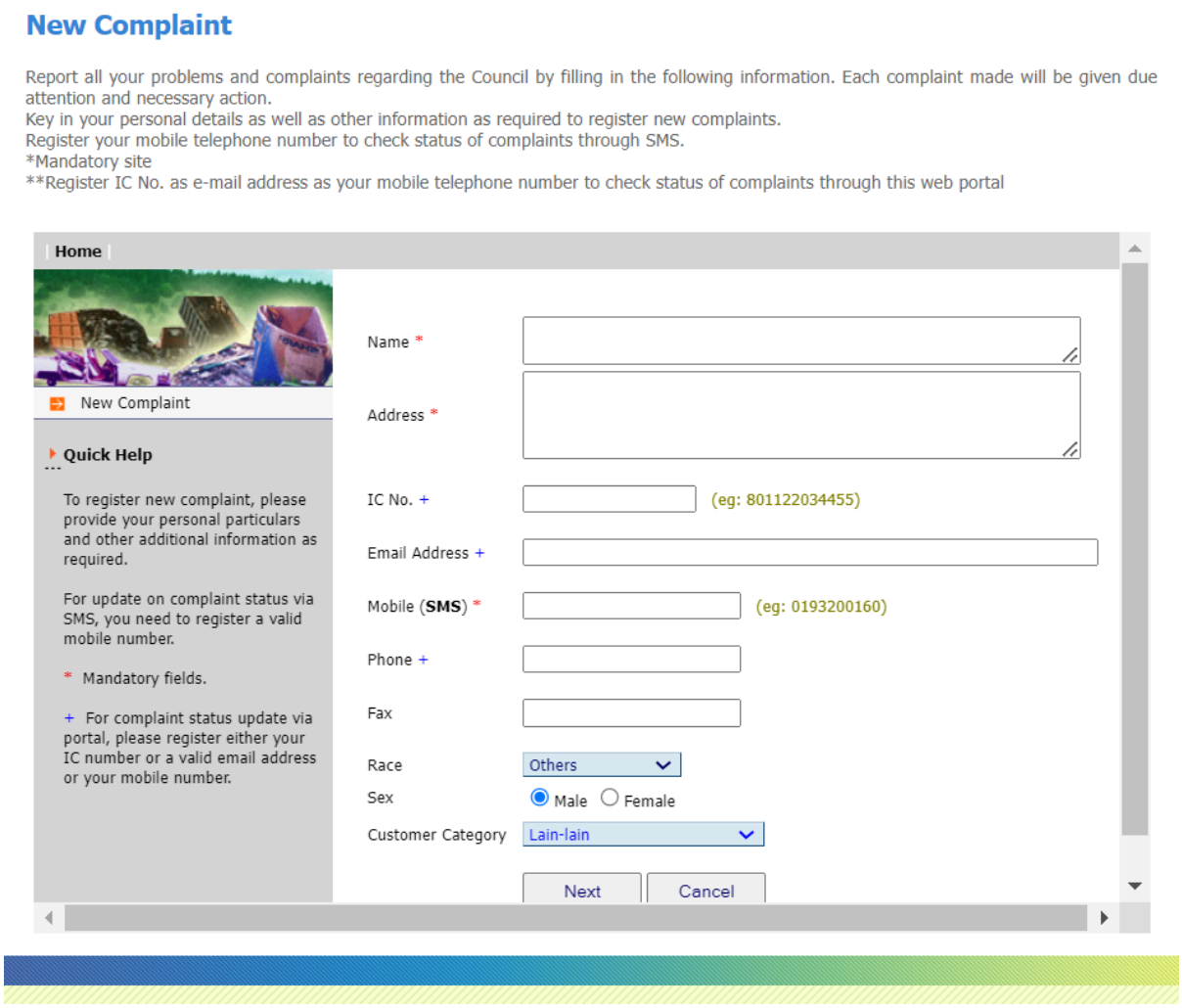


Figure 2.2 E-Complaints MBPP New complaint page (MBPP, n.d.)

2.1.2(b) Problem description

After filling in personal details, a complainant needs to explains his problem. Figure 2.3 is the problem description page for all complainants to fill in. First, the complainant has to select problem category, then select problem type (or sub-category) (MBPP, n.d.). The complainant also needs to select a service area (location) and sub-service area (MBPP, n.d.). Lastly, the complainant has to write a problem title, problem location and problem description (MBPP, n.d.). There is a long list of problem categories for the complainant to select, such as animal, building, cleanliness, environment, obstruction, parking, public facilities, road, etc (MBPP, n.d.). Problem type is sub-categories under the selected problem category (MBPP, n.d.). For example, under road category, the selectable problem types are road damage / hole, traffic sign board damage, overgrown grass at the road pavement, etc (MBPP, n.d.). Service area is selectable and refers to the locality of the problem, such as Bayan Lepas, Batu Lanchang, Pantai Jerejak, etc., whereas problem location is for the complainant to write the address of the location of the problem (MBPP, n.d.). In this page, all fields are mandatory (MBPP, n.d.). The length limit for problem title, problem location and problem description are 150, 250 and 500 characters respectively (MBPP, n.d.).

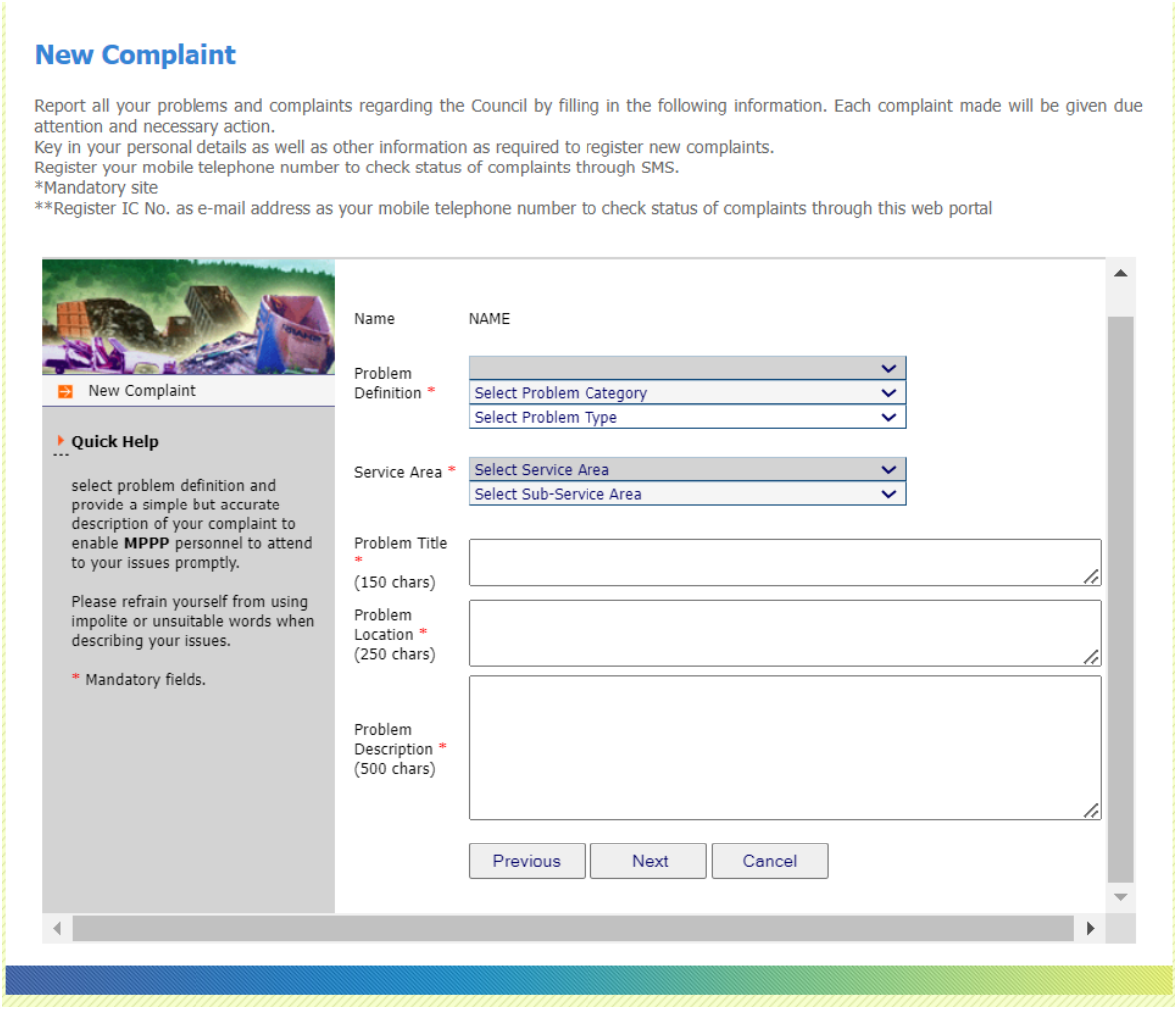
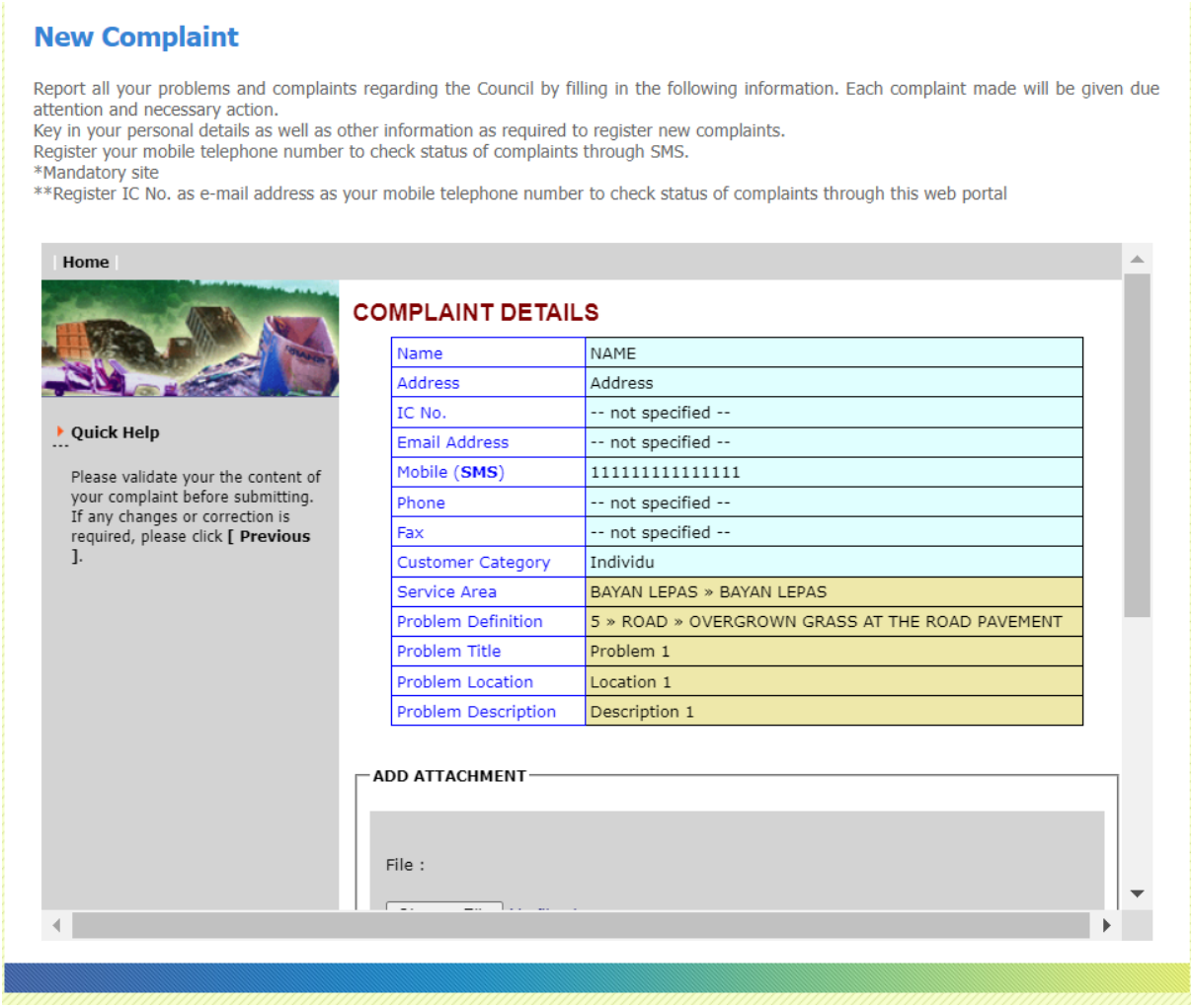


Figure 2.3 E-Complaints MBPP Problem description page (MBPP, n.d.)

2.1.2(c) File attachment and confirm complaint details

As the last step of lodging complaint, the complainant needs to confirm the complaint information and attach any supplementary file or document if necessary. Figure 2.4 is the confirm complaint details page. Before submission, particulars of the complaint case, such as name, address, IC number, email address, mobile phone number, phone number, fax number, customer category, service area, problem definition, problem title and problem location are listed for the complainant to check and confirm (MBPP, n.d.). Problem definition is a combination of problem category and problem type as described earlier. Figure 2.5 is the add attachment file page. The complainant is allowed to upload files but there is a size limit of 2 megabyte (MBPP, n.d.).



New Complaint

Report all your problems and complaints regarding the Council by filling in the following information. Each complaint made will be given due attention and necessary action.
Key in your personal details as well as other information as required to register new complaints.
Register your mobile telephone number to check status of complaints through SMS.
*Mandatory site
**Register IC No. as e-mail address as your mobile telephone number to check status of complaints through this web portal

Home

Quick Help

Please validate your the content of your complaint before submitting. If any changes or correction is required, please click [**Previous**].

COMPLAINT DETAILS

Name	NAME
Address	Address
IC No.	-- not specified --
Email Address	-- not specified --
Mobile (**SMS**)	111111111111111
Phone	-- not specified --
Fax	-- not specified --
Customer Category	Individu
Service Area	BAYAN LEPAS » BAYAN LEPAS
Problem Definition	5 » ROAD » OVERGROWN GRASS AT THE ROAD PAVEMENT
Problem Title	Problem 1
Problem Location	Location 1
Problem Description	Description 1

ADD ATTACHMENT

File :

Figure 2.4 E-Complaints MBPP Confirm complaint details page (MBPP, n.d.)

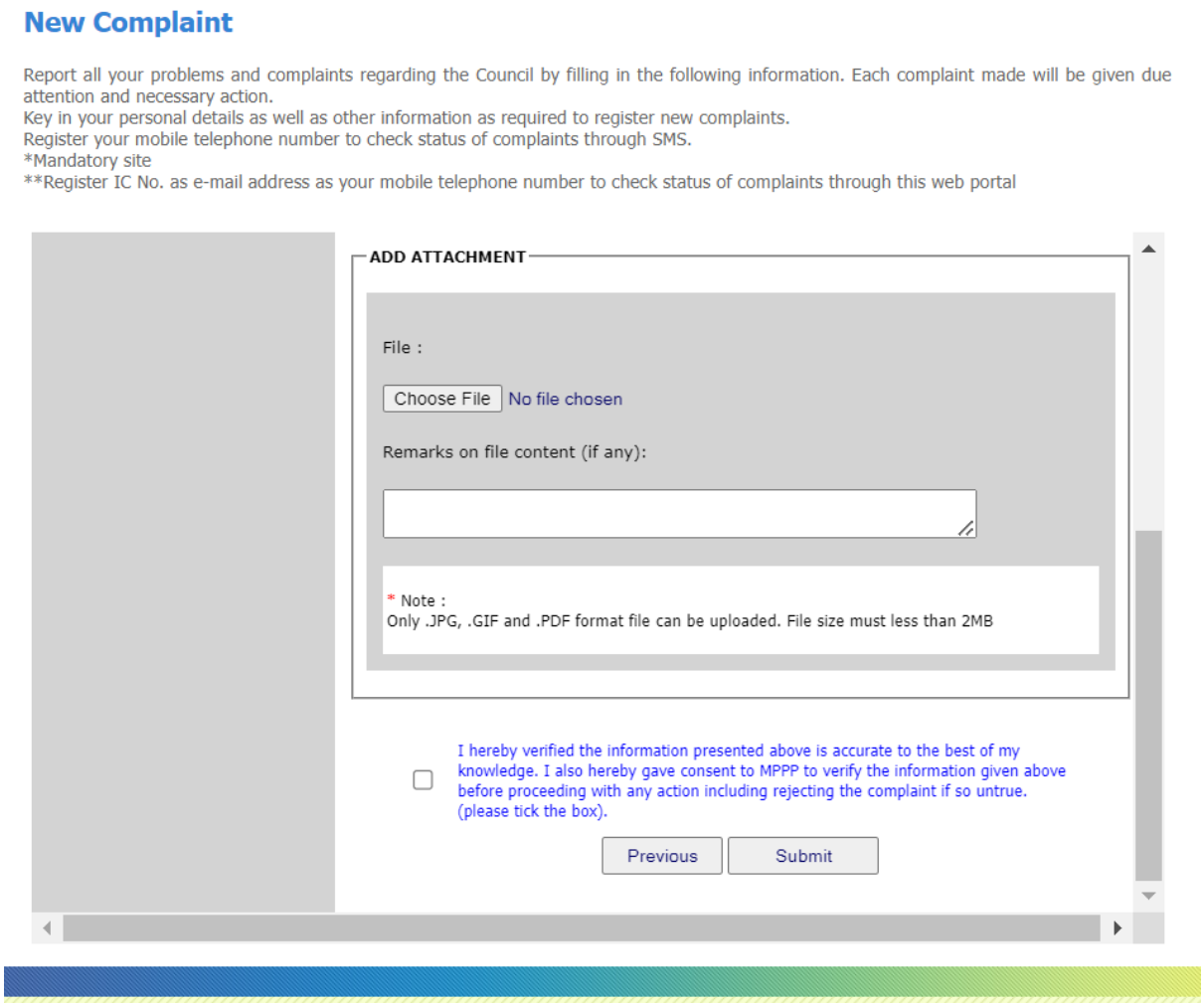


Figure 2.5 E-Complaints MBPP Add attachment page (MBPP, n.d.)

2.1.3 Check complaint status

In E-Complaints MBPP, the complainant has several ways to check the complaint status. For registered complainants, they can login and check the status (MBPP, n.d.). Figure 2.6 is the view complaint page for non-registered complainant. For non-registered complainants, they can access the web portal and key in the complaint reference number and any one of the following: IC number, mobile phone number, email address (MBPP, n.d.). The possible statuses are accepted, inquiry in action, have done, closed and have been cancelled (MBPP, n.d.).

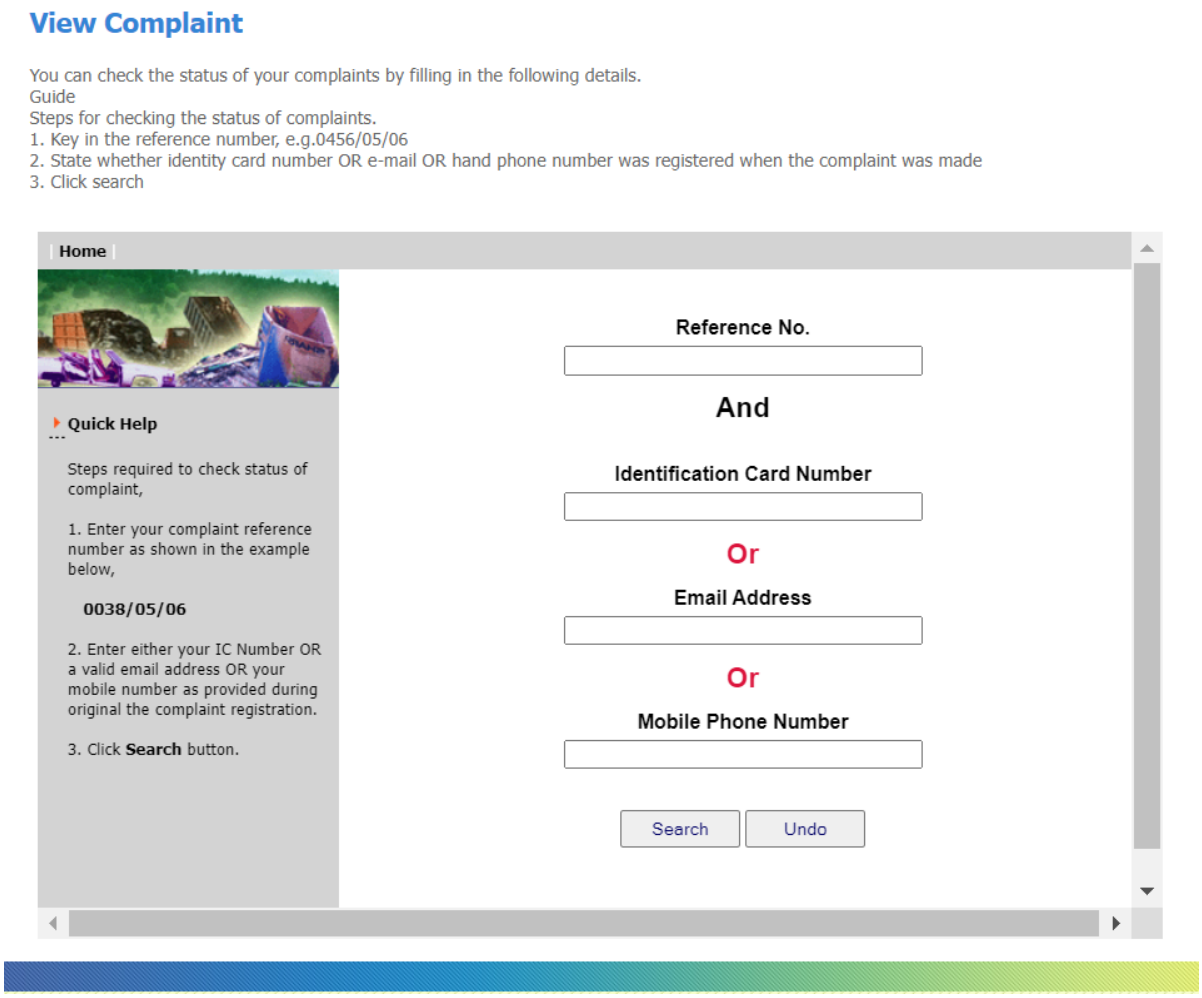


Figure 2.6 E-Complaints MBPP View complaint page (MBPP, n.d.)

2.2 Adu@KL[2] (Kuala Lumpur, Malaysia)

Adu@KL is the complaint system in Kuala Lumpur, Malaysia. Adu@KL is a web-based system operated by Kuala Lumpur City Hall (*Dewan Bandaraya Kuala Lumpur*, DBKL) and can be accessed from adukl.dbkl.gov.my. In Adu@KL, the term “feedback” is used instead of “complaint”, so under this subtopic these two terms will be used interchangeably.

[2] adukl.dbkl.gov.my

2.2.1 Submit feedback

In Adu@KL, a complainant has to fill in the feedback form to submit his feedback. The feedback form has three parts. The first part is for filling in feedback details, second is personal details, the last one is a legal disclaimer, which is not the interested part of this reseach.

2.2.1(a) Fill in feedback details

Figure 2.7 is the fill in feedback details page. The complainant needs to write the feedback subject, detail and expected result (DBKL, n.d.). The complainant can specify the location of the problem on an interactive map, as well as attach a file if necessary (DBKL, n.d.). Subject and detail are mandatory fields (DBKL, n.d.).

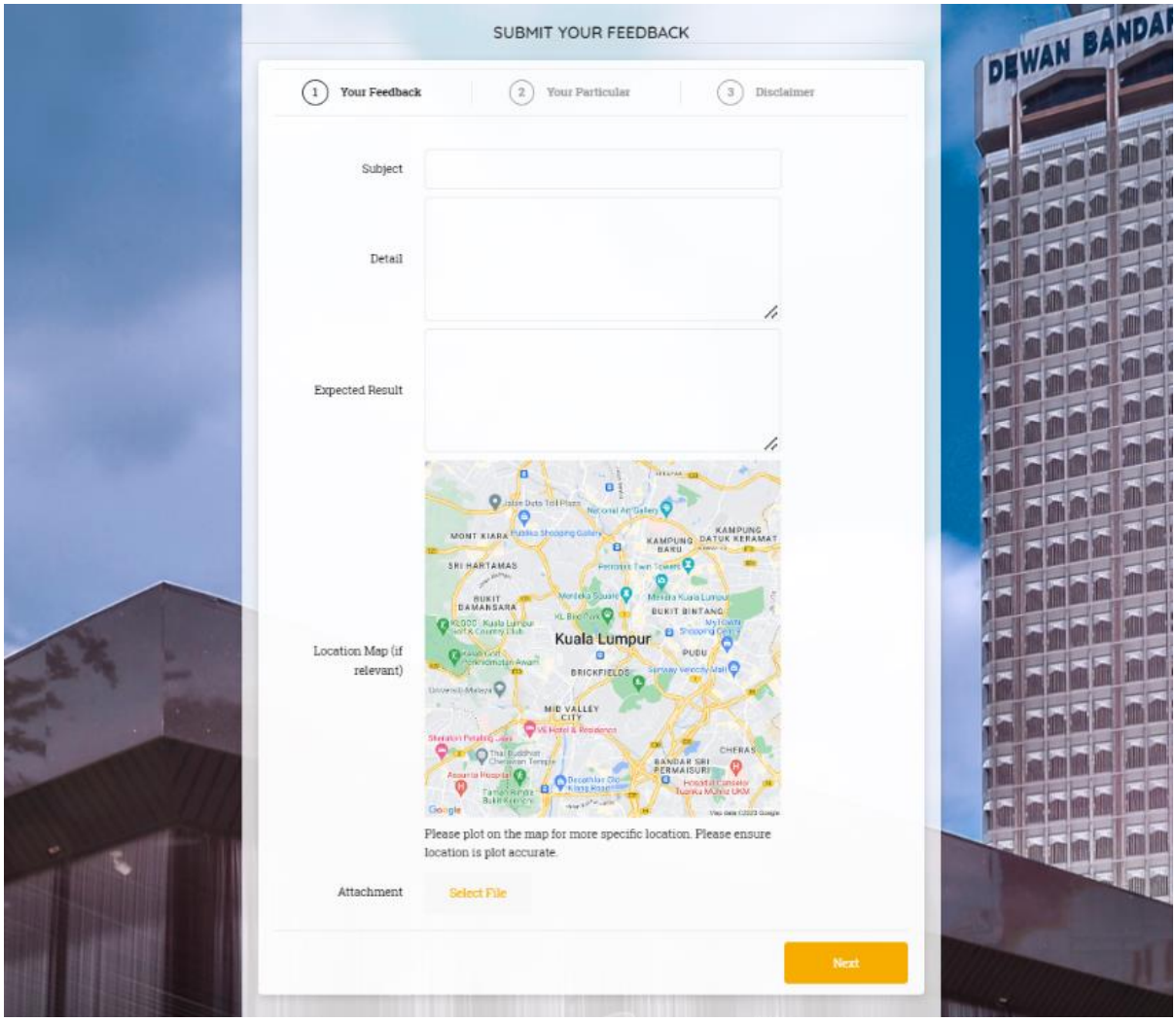


Figure 2.7 Adu@KL Fill in feedback details page (DBKL, n.d.)

2.2.1(b) Fill in personal particulars

At the second step of feedback submission, the complainant needs to provide his personal particulars. Figure 2.8 is the fill in personal particulars page. The complainant has to provide his name, ID number, company name (optional), mobile phone number and email (DBKL, n.d.). If the complainant wants to protect his complaint case, he can set a password for his feedback. If the complainant is individual, the ID number can the complainant's MyKad number or passport number (DBKL, n.d.). If the complainant is a company, then the ID number is the company's Registry of Companies (RoC) number (DBKL, n.d.). The mandatory fields are name, ID number and mobile phone number (DBKL, n.d.). The third step, disclaimer, is a legal statement that states the terms and conditions of the complaint (DBKL, n.d.). The third step is not interested in this research.

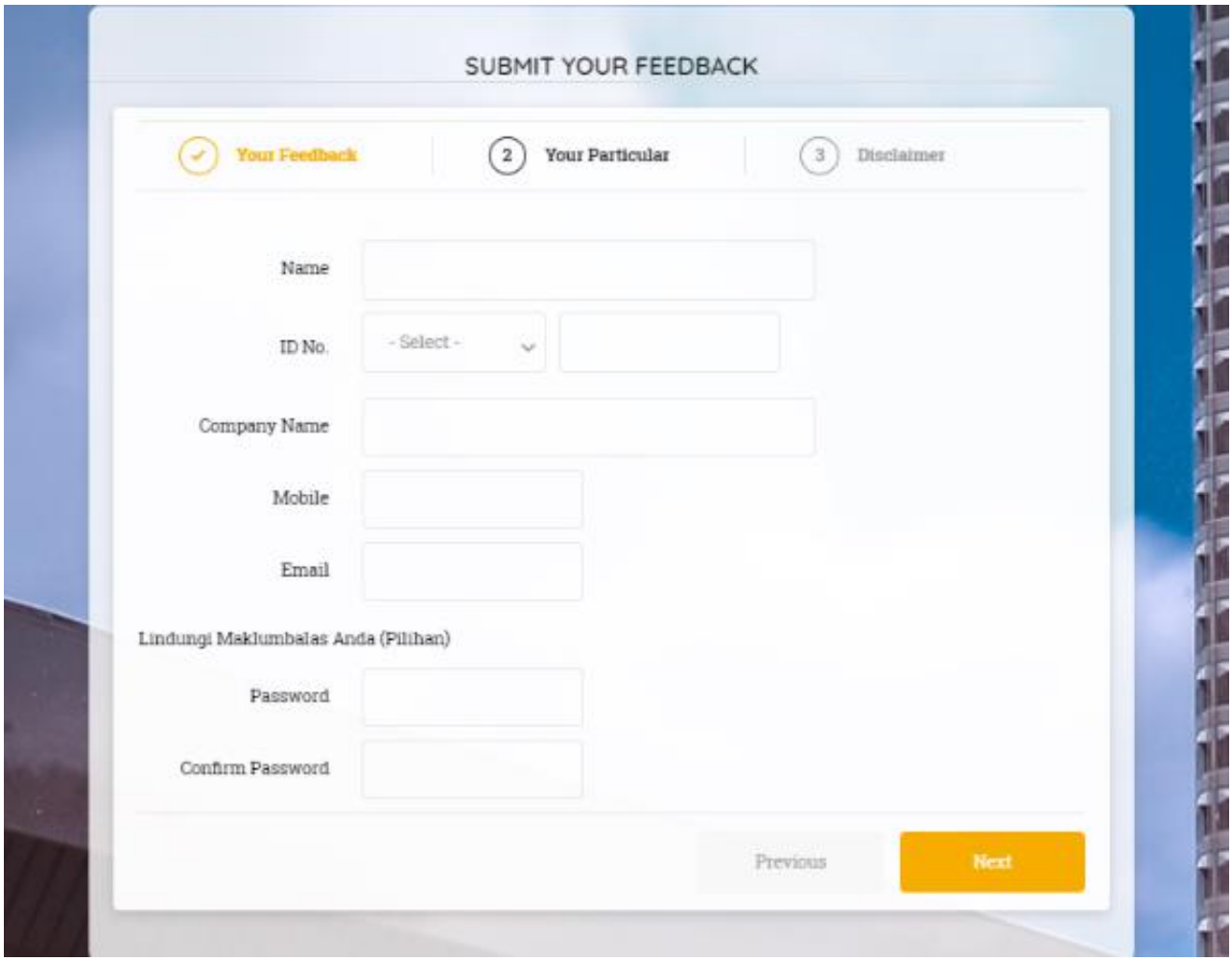


Figure 2.8 Adu@KL Fill in personal details page (DBKL, n.d.)

2.2.2 Feedback enquiry

In Adu@KL, the complainant can enquire the feedback status from the feedback enquiry page. Figure 2.9 is the feedback enquiry page. The complainant can choose to key in the ticket ID of the feedback and password (if they have specified password during submission (DBKL, n.d.). Alternatively, the complainant can also find his complaint case using his ID number (refer to 2.2.1(b)) (DBKL, n.d.).

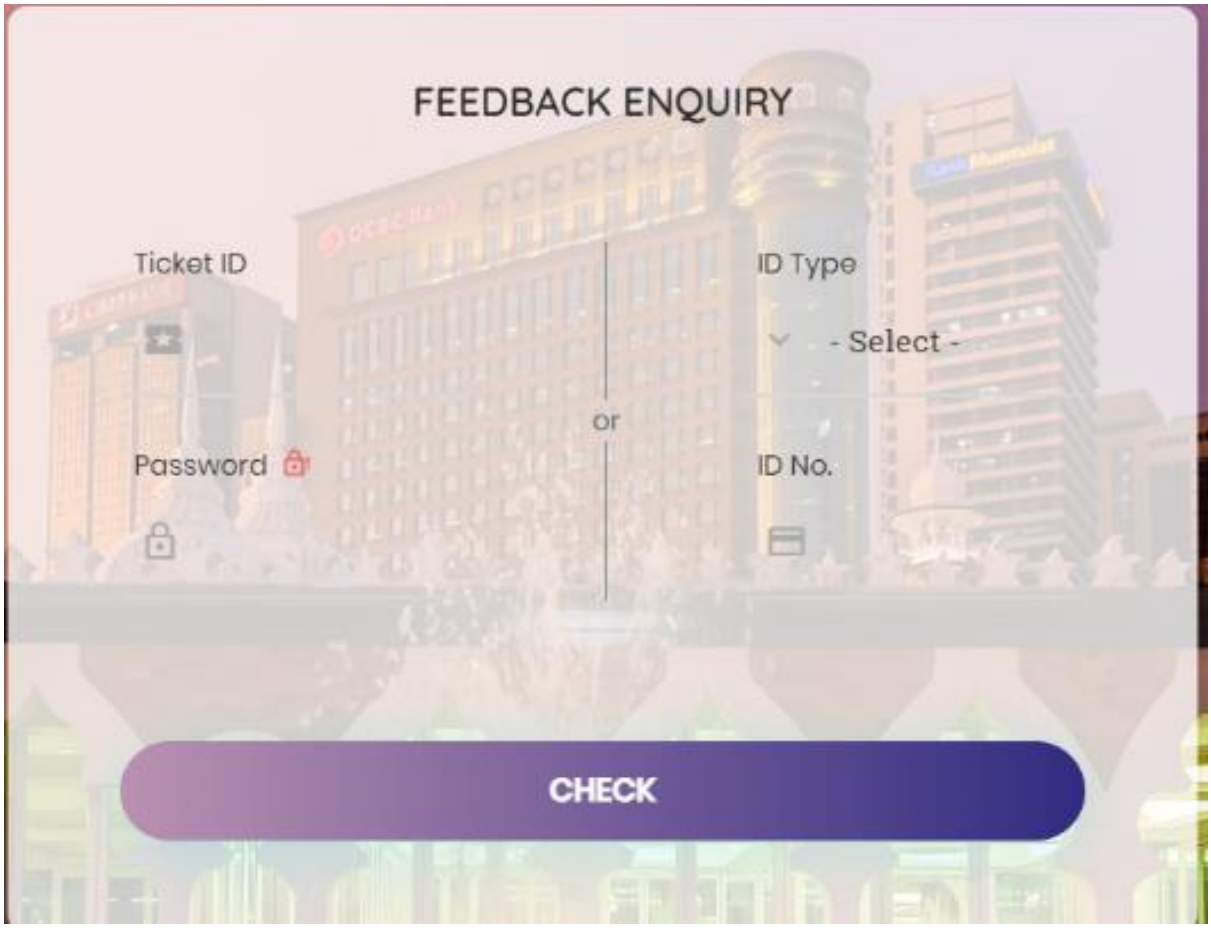


Figure 2.9 Adu@KL Feedback enquiry page (DBKL, n.d.)

2.3 Public Complaint System (Conceptual system)

Suryotrisongko, et al. (2017) designed and developed a conceptual public complaint system (PCS). PCS is a web application that is deployed in cloud environment and accessible through browser (Suryotrisongko, et al., 2017). PCS is designed based on Software-as-a-Service (SaaS) concept and serves as a microservice that provides complaint management service for the bigger e-government system (Suryotrisongko, et al., 2017). The idea of PCS is such that different local government units rent and pay the vendor to use PCS (Suryotrisongko,

et al., 2017). Based on the documentation, PCS has four actors, namely citizen, government work unit, administrator and vendor (Suryotrisongko, et al., 2017). The local government unit assigns an administrator whose role is to upload citizen ID data and manage the list of categories (Suryotrisongko, et al., 2017). The vendor actor's role is to view customer information (Suryotrisongko, et al., 2017). Based on the process, the vendor is the operator of the system and the customer is local government unit that rents the system.

2.3.1 Lodge complaint

A citizen can use PCS to lodge complaint to local government unit, provided if the unit which he wants to complaint to is renting PCS to handle complaint cases (Suryotrisongko, et al., 2017). When a citizen lodges a complaint, the citizen's ID will be verified based on the citizen ID uploaded by the administrator assigned by the local government unit (Suryotrisongko, et al., 2017). The complaint case will be stored in the cloud database at the vendor side, which can be accessed by the government work unit through browser (Suryotrisongko, et al., 2017). Based on the documentation, the actor, government work unit, is a person who works for the local government unit and whose role is to handles the complaint case (Suryotrisongko, et al., 2017). In the following text under this subtopic, government work unit refers to the person, whereas local government unit refers to the department.

2.3.2 Change complaint status

In PCS, the government work unit will read the complaint and decide whether the complaint content is valid (Suryotrisongko, et al., 2017). If the complaint content is valid, then only the government work unit will accept the complaint case (Suryotrisongko, et al., 2017). The government work unit will classify the complaint based on the list of categories set by the administrator of the respective local government unit (Suryotrisongko, et al., 2017). The government work unit will send complaint case to the related government unit out there (Suryotrisongko, et al., 2017). The government unit that receives the complaint will try to solve the complaint through external process (Suryotrisongko, et al., 2017). When the complaint is resolved, the citizen who lodged the complaint can change the complaint status to completion (Suryotrisongko, et al., 2017).

2.3.3 Security aspects of the application

In PCS, login is not required for citizen, but is required for all other types of users, such as government work unit, administrator and vendor (Suryotrisongko, et al., 2017). Once login is successful, PCS identifies the user identity by sending a javascript object notation web token (JWT) that contains the user information (Suryotrisongko, et al., 2017). PCS protects the user password by using Blowfish crypt (BCrypt) algorithm to hash the password (Suryotrisongko, et al., 2017).

2.4 Statistical reporting in complaint system

Under this subtopic, two conceptual complaint management system are analysed for their approach of statistical reporting.

2.4.1 Electronic-Customer Complaint System (Reporting in chart)

Afify & Kadry (2019) designed and developed a conceptual Electronic-Customer Complaint System (E-CCMS). In E-CCMS, the complaint statistics are reported to the supervisor actor in charts (Afify & Kadry, 2019).

2.4.1(a) Pie chart

According to the documentation, E-CCMS presents the following data using pie charts (Afify & Kadry, 2019):

1. Percentage of complaints in each status
2. Percentage of complaints in each priority group
3. Percentage of complainants in each gender
4. Percentage of complainants in each marital status

2.4.1(b) Bar chart

Based on the description in the documentation, E-CCMS presents the following data using bar charts (Afify & Kadry, 2019):

1. Number of complaints of each day in a month
2. Number of complaints of each quarter in a year
3. Number of complaints from each region
4. Number of complaints based on classification (category)
5. Number of complaints based on sub-classification (sub-category)

2.4.2 Online Complaint Management System (Reporting in simple dashboard)

Bhadouria, et al. (2021) designed and developed a conceptual online complaint management system (OCMS). In OCMS, dashboard is used to present the complaint statistics (Bhadouria, et al., 2021). In the dashboard in OCMS, the following information are displayed (Bhadouria, et al., 2021):

1. Number of complaints received on today
2. Number of complaints received by the system in this month
3. Number of users signed up in the system

2.5 Critical analysis

Under this subtopic, the findings from the previous subtopics will be discussed and summarised. These findings will bring to the proposed work.

2.5.1 Summary of literature review

Under this subtopic, the findings from the literature review of three existing and similar systems (subtopic 2.1, 2.2 and 2.3), as well as statistical reporting (subtopic 2.4), will be discussed and summarised.

2.5.1(a) Main summary

After reviewing the three systems, it can be concluded that conventionally in public complaint systems, authentication of complainant is not required for lodging complaint. In Adu@KL and PCS, there is no complainant registration and login process. The complainant can straight away go to lodge complaint page without any login. In E-Complaints MBPP, registration and login features are present, but it is not mandatory, just that the registered

complainant can save their personal information and do not need to fill in the personal information form. So, it is clear that in some public complaint system, for the lodge complaint page might not require the user to login. However, the propose work will require the complainant to login in order to lodge complaint to enhance the security of the system.

However, on the other hand, authentication of complainant is required for checking complaint status. It is not clear on how PCS handles this, but both E-Complaints MBPP and Adu@KL requires the complainant to authenticate himself. The fields used for authentication is complaint ticket number plus password (set by the complainant when lodging complaint) or complaint ticket number plus one personal unique identifier (MyKad number, company registration number, mobile phone number or email address). This is to ensure that only the complainant himself can access to the complaint status. The proposed work will also require login for check complaint page.

In E-Complaints MBPP and Adu@KL, when the complainant status is updated by the city council and the complainant can only check the status. Oppositely, in PCS, the complainant updates the complaint status and closes the complaint case after he is satisfied with the solution. So, in conclusion, whether the complaint status is updated by the complainant or the complaint handler actually does not matter. The proposed work will only allow the complaint handler to update the status because the complaint handler is the person who follows the case solving process, so the complaint handler might be in a more suitable position to update the status.

2.5.1(b) Complaint fields

Table 2.1 Complaint description fields

Field name	Is mandatory	Additional remarks
Problem title	Yes	
Problem details	Yes	
Expected result	No	
Area	Yes	Choice given
Incident address	No	
Category	Yes	Choice given

In Table 2.1, the consolidated complaint description fields are listed. After reviewing E-Complaints MBPP and Adu@KL, it can be concluded that problem title, details and location

(either area and full address or precise location) are the fields that both systems asked for. On the other hand, category and expected result are asked in only one of the systems. In the proposed work, only the fields that are asked by both systems will be listed as mandatory.

Table 2.2 Personal information fields

Field name	Is mandatory	Additional remarks
Complainant name	Yes	
Complainant IC number	Yes	
Email address	Yes	
Mobile phone number	Yes	
Home address	No	
Fax number	No	
Gender	No	Choice given

In Table 2.2, the consolidated personal information fields are listed. After reviewing E-Complaints MBPP and Adu@KL, the personal information that both systems asking from the complainant are complainant name or organisation name, unique identifier (MyKad number, IC number, passport number or company registration number), email address and mobile phone number are asked in both systems. On the other hand, address, fax number, gender and race are asked in one of the systems. In the proposed work, only the fields that both systems asked for will be set as mandatory because this information is the minimal personal information needed for a city complaint system.

2.5.1(c) Statistical reporting

As for statistical reporting of complaint cases, commonly statistical reporting in complaint system will report the break down of complaints based on the category, region and status, the break down of complainants based on gender and race, and key information such as total number of complaints and total number of complainants.

As seen in E-CCMS, both pie chart and bar chart can accommodate detailed grouped information such as the number of complaints of each category or area. So, in the proposed work, both pie chart and bar chart will be used. As an enhancement, the proposed work will also used stacked bar chart, which can further analyse the data in each category, by grouping based on status.

2.5.2 Comparison of the proposed work and the existing and similar systems

Table 2.3 Feature comparison

Feature	E-Complaints MBPP	Adu@KL	PCS	The proposed work
Fill in complaint form	Yes	Yes	Yes	Yes
Allow file attachment	Yes	Yes	No	Yes
Complainant email verification through OTP	No	No	No	Yes
Dynamic rendering of options for category and area in complaint form	No	No	No	Yes
Auto-send receipt to complainant email	No	No	No	Yes
Check complaint status	Yes	Yes	Yes	Yes
Add message to complaint case	No	No	No	Yes
Auto-send email to notify complainant when status changed	No	No	No	Yes
Statistical reporting	No	No	No	Yes

The comparison of the features in the proposed work and the existing and similar systems reviewed are listed in Table 2.3. The systems reviewed in subtopic 2.4, i.e. E-CCMS and OCMS, are not being compared because only the statistical reporting features of those systems are reviewed. In addition, those systems are commercial complaint system, which differ from those public complaint system and city complaint systems reviewed in subtopics 2.1, 2.2 and 2.3.

As compared to the existing and similar public complaint systems, the proposed work will have following additional features.

1. **Email address verification**

Email address verification is an essential part of this project because this is a feature which the existing city complaint systems reviewed do not implement. OTP verification adds another layer of security to the registration process by ensuring the email address is valid before allowing user to register (Sandeep, 2023). This prevents the creation of account using a fake email address (Sandeep, 2023). This is important in the proposed work because it ensures that there is at least one valid and effective communication channel. The validation of communication information, such as email address and mobile phone number, through checking the pattern of the input does not necessarily confirm that the email address or mobile phone number is a real one.

2. **Dynamic rendering of options of category and area**

   Dynamic rendering of options of category and area will provide the administrator with more flexibility in terms of categorizing the complaint case.

3. **Sending receipt and notification of change of status of the complaint case to complainant email**

   This feature will remind the complainant about the new changes of the case solving process, as well as giving the complainant a record of the progress of the case.

4. **Add message to complaint case**

   This feature can be used as an optional communication channel between the complainant and the complaint handler.

5. **Statistical reporting of complaint cases in the system**

   This feature gives the administrator an analytical view point which the administrator can analyse the complaint case based on the case count of each category, area or status in means of chart. This will give the administrator an overview of the complaint cases, which facilitate him to analyse the cases.

3. RESEARCH METHODOLOGY

3.1 Waterfall Development Model

This project will follow the waterfall development model because the development work in this project will be implemented sequentially. Figure 3.1 is an illustration of the steps in waterfall development model. This project will follow all the steps in this model, except for maintenance step because the proposed work is just a conceptual system.

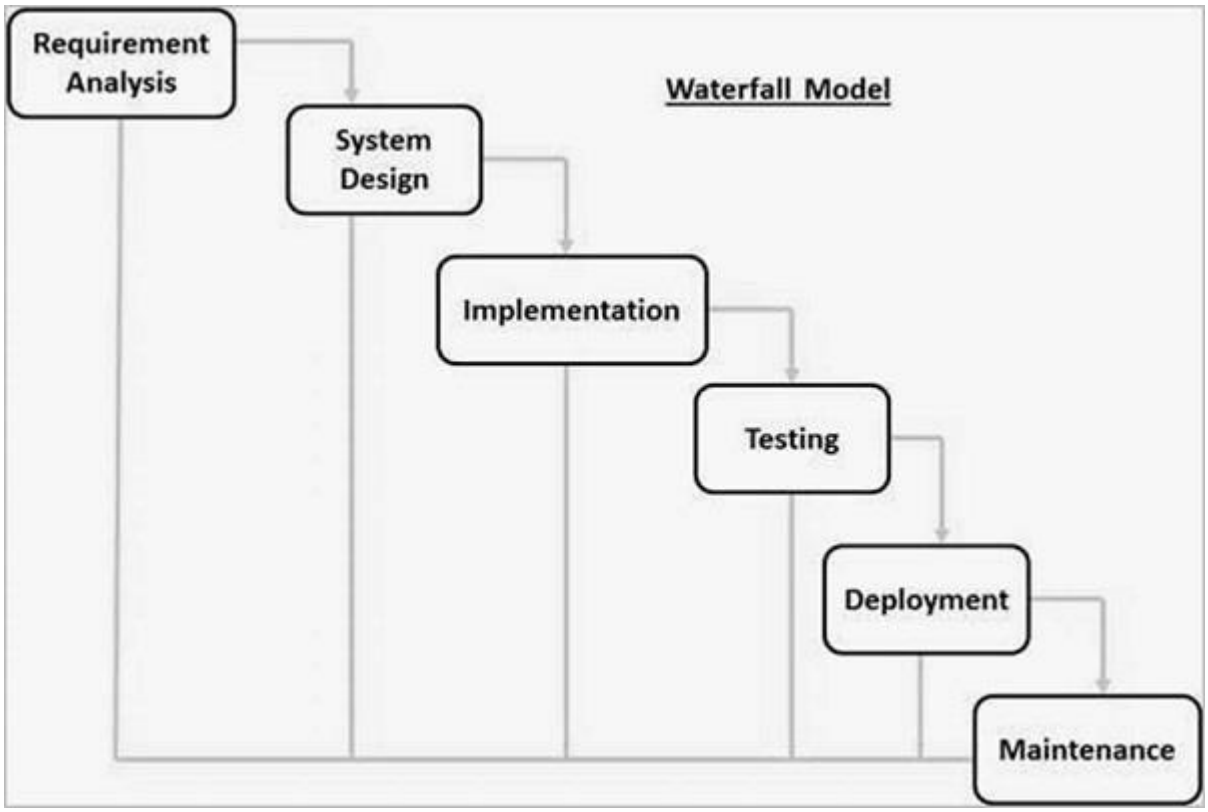


Figure 3.1 Waterfall development model (Tutorials Point, n.d.)

3.1.1 Requirement analysis

The requirements of this project are gathered during this phase. The existing and similar systems, namely E-Complaints MBPP, Adu@KL and PCS (subtopics 2.1, 2.2 and 2.3), are reviewed and analysed. The important features of a city complaint system should have, such as lodge complaint, attach file, check complaint status, are listed and recorded. The fields that normally public complaint systems will ask from the complainant are consolidated by comparing across the existing systems.

In addition, the statistical reporting features in two commercial complaint systems are reviewed (subtopic 2.4). The information that commonly reported in complaint systems are combined across the two systems, listed and recorded.

3.1.2 System design

During system design phase, the use case diagrams of three actors in the proposed work, namely complainant, complaint handler and administrator, are designed and drawn (subtopics 3.3.1, 3.3.2 and 3.3.3). Entity relationship diagram is also drawn. The flow charts of each process are also drafted (displayed in Chapter 4). The break down of the workload is also drafted. Along the process of implementation, in fact this planning and diagrams are slightly changes along the way. The final break down of the workload is displayed in Figure 3.2.

Workload breakdown

Objective	Feature	Component	Period of time			
			30/9-17/10	18/10-24/10	25/10-31/10	1/11-7/11
1	Complainant registration	Fill in personal details				
		Verify email address				
		Display complete registration message				
	Login					
	Lodge complaint	Load categories and areas				
		Fill in complaint description				
		Auto-send summary of complaint to complainant email				
		Display summary of complaint on screen				
	Manage system (administrator)	Display list of categories				
		Add / delete category				
		Display list of areas				
		Add / delete area				
		Display list of complaint handlers				
		Add / delete complaint handler				
		Auto-send set password link to new complaint handler email				
2	Statistics reporting	Specify start date and end date				
		Display total number of complaints				
		Display bar chart of number of complaints of each area				
		Display bar chart of number of complaints of each category				
		Display pie chart of complaint based on area				
		Display pie chart of complaint based on category				
		Display pie chart of complaints based on status				
		Display number of complaints in each status				
1	Complaint handler registration (and also Forget password)	Verify email address				
		Set password				
	View complaint	Auto-decide list of complaints based on role and user identity				
		Display list of complaints				
		Display complaint details and status based on selection				
		Search for complaint by title				
		Filter complaint by area / category / status				
		Sort complaint by title / date / area / category / status				
	Change status	Change complaint status (administrator / complaint handler)				
		Auto-send email to complainant when there is change in status				
		Assign complaint handler (administrator)				
	Send message	Add new message				
		Auto-send email to complainant when there is new message from complaint handling side				
	Manage account	Display email address, name and role				
2	Statistics reporting (continued from above)	Display table of complaints case counts of each status based on area and category				
		Change bar chart (referred to above) to stacked bar chart				

Figure 3.2 Work load break down (final)

3.1.3 Implementation

Based on the work load break down, the development work is carried out stage by stage. Every week, the development work is committed to the GitHub. Figure 3.3 is the screenshot of the GitHub repository.

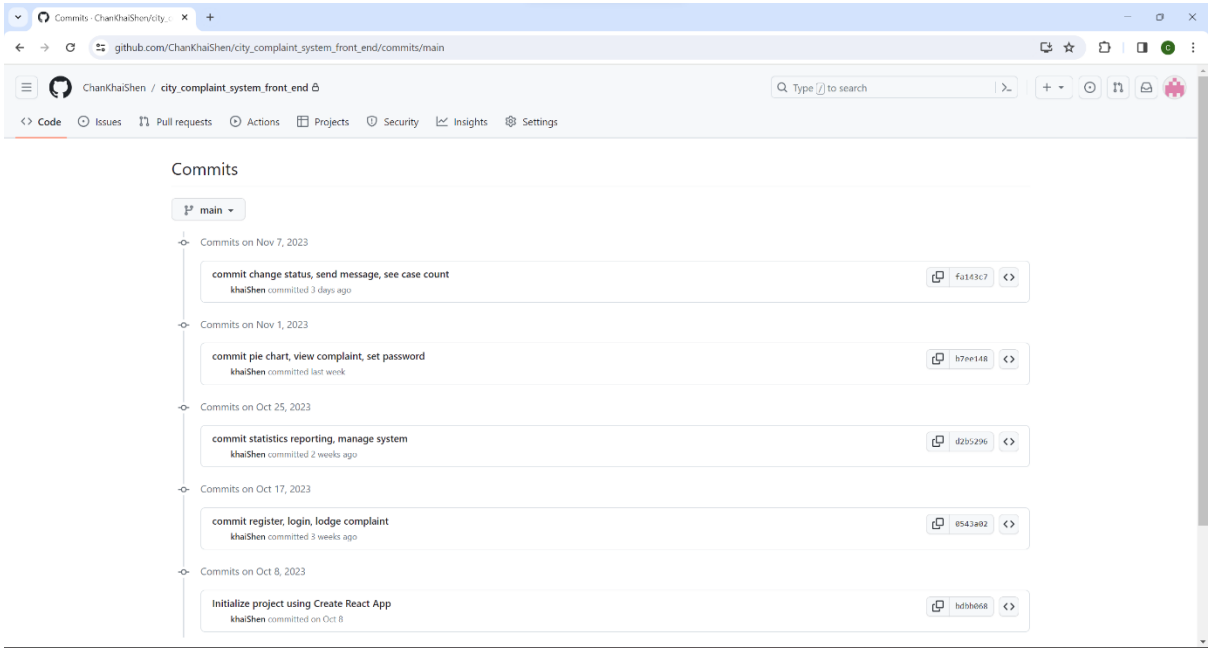


Figure 3.3 GitHub Repository

3.1.4 Testing

In the last stage, testing is done and is documented in Chapter 5.

3.2 Gannt Chart

Figure 3.4 is the Gannt chart for Apr 2023 semester, whereas Figure 3.5 is the Gannt chart for Aug 2023 semester.

Week	**6/4**	**13/4**	**20/4**	**27/4**	**4/5**	**11/5**	**18/5**	**25/5**	**1/6**	**8/6**	**15/6**	**22/6**
Requirement analysis												
System design												

Figure 3.4 Gannt Chart (Apr 2023)

Week	24/8	31/8	7/9	14/9	21/9	28/9	5/9	12/9	19/9	26/9	2/9	7/9
System design												
Implementation												
Testing												

Figure 3.5 Gannt Chart (Aug 2023)

3.3 Use Case Diagram

Under this subtopic, the use case diagrams of the three actors in the proposed system will be presented and discussed. These three actors are complainant, complaint handler and administrator.

3.3.1 Complainant

Figure 3.6 is the use case diagram for complainant actor. A complainant needs to register before he is able to use the proposed work. During registration, the complainant has to provide email address, password and personal information, namely name, IC number, mobile phone number, home address (optional) and fax number (optional). After that, the server of the proposed work will send an email which contains a one-time password (OTP) to the email address provided. Then the complainant has to enter the correct OTP on the registration page to complete the registration process.

After register, the complainant can login to the proposed work. Upon successful login, the first page that the complainant will see is the manage account page, which will display the complainant's email address, name and role.

After login, the complainant will be able to lodge complaint. When lodging complaint, the complainant should give the complaint description, which includes title, description, expected result (optional), area and incident address (optional). Upon confirming receive the complaint case, the server of the proposed work will send an email which contains the summary of complaint case to the complainant. The same summary of complaint case will also be displayed on screen.

After login, the complainant will be able to check the complaint. When checking the complaint, the complainant will see a list of complaints which are lodged by him. On the check complaint page, the complainant can search a complaint case by specifying the title, specify

the range of date of the displayed complaint cases, filter complaint case by specifying title, area, category and status, and sort the complaint cases by title, date, area, category and status.

The complainant can click on any complaint case to see the details. On the complaint details page, the complainant can see the details of the complaint case, see the histories of the complaint case (change of status and histories of messages) and add new message to the complaint handler (or replying to the complaint handler).

If the complainant forgets his password, he can reset the password by clicking on the forget password link on the login page. On the set password page, the complainant will need to provide his email address. Upon confirming the email address is a valid user who exists in the database, the server in the proposed work will send an OTP to the complainant's email. After that, the complainant will need to enter the correct OTP on the set password page to be able to set the new password.

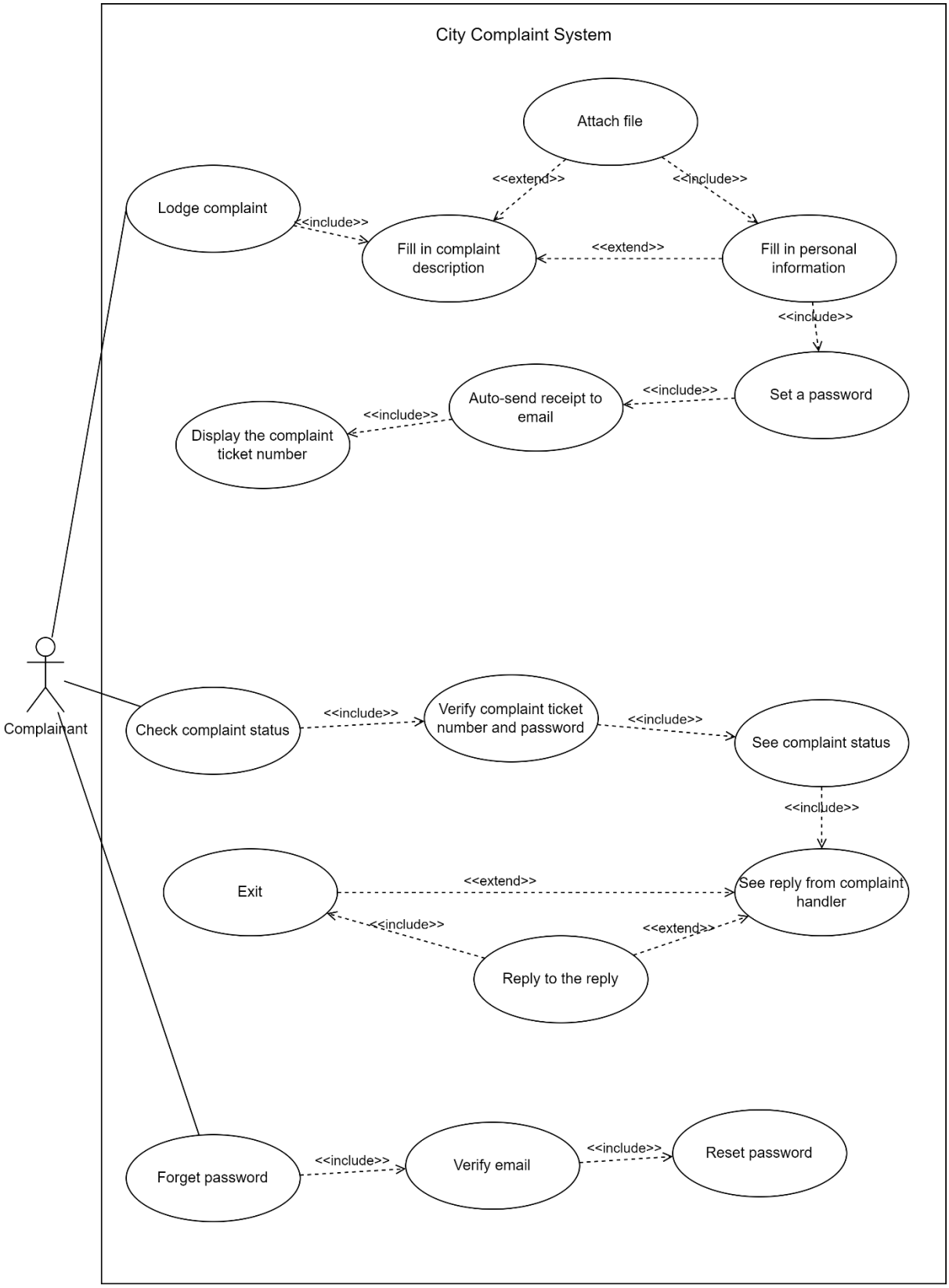


Figure 3.6 Use case diagram (Complainant)

3.3.2 Complaint handler

Figure 3.7 is the use case diagram for complaint handler actor. The complaint handler is registered by the administrator (for details, see subtopic 3.3.3). To complete the registration, the complaint handler needs to click on the link sent to his email. The link will bring him to the set password page. In fact, this is the same process to reset password. The complaint handler will need to enter his email address. Upon the confirming the identity of the complaint handler, the server in the proposed work will send an email which contains an OTP to the complaint handler. After entering the correct OTP, the complaint handler will be bale to set his password.

After login to the proposed work, the complaint handler will see the manage account page, which has his email address, name and role.

After the complaint handler login into the system, the complaint handler can check complaint cases. A list of complaint cases, which includes the complaint cases which its complaint handler is the current user or does not have complaint handler, will be displayed on the check complaint page. As for search, filter and sort features, it exactly same to the complainant. The complaint handler can search a complaint case by specifying the title, set the range of date of the displayed complaint cases, filter complaint case by specifying title, area, category and status, and sort the complaint cases by title, date, area, category and status.

On the check complaint page, the complainant can click on any complaint case to see the details. On the complaint details page, the complainant can see the details of the complaint case. Different from the complainant who can only see complaint details, the complaint handler will be able to see the complaint details, complainant personal details and complaint handler details (which is himself). The complaint handler also can see the histories of the complaint case (change of complaint handler, change of status and histories of messages) and add new message to the complainant (or replying to the complainant). On the complaint details page, the complaint handler can change the status of the complaint with the following restrictions, if the current status is ‘Received’, then the new status can only be ‘Progressing’, else if the current status is ‘Progressing’ then the new status can only be ‘Solved’.

If the complaint handler forgets his password, he can reset the password by clicking the forget password link on the login page. The complaint handler will need to go through the same process as the set password process during registration as mentioned above, which will include email verification through OTP.

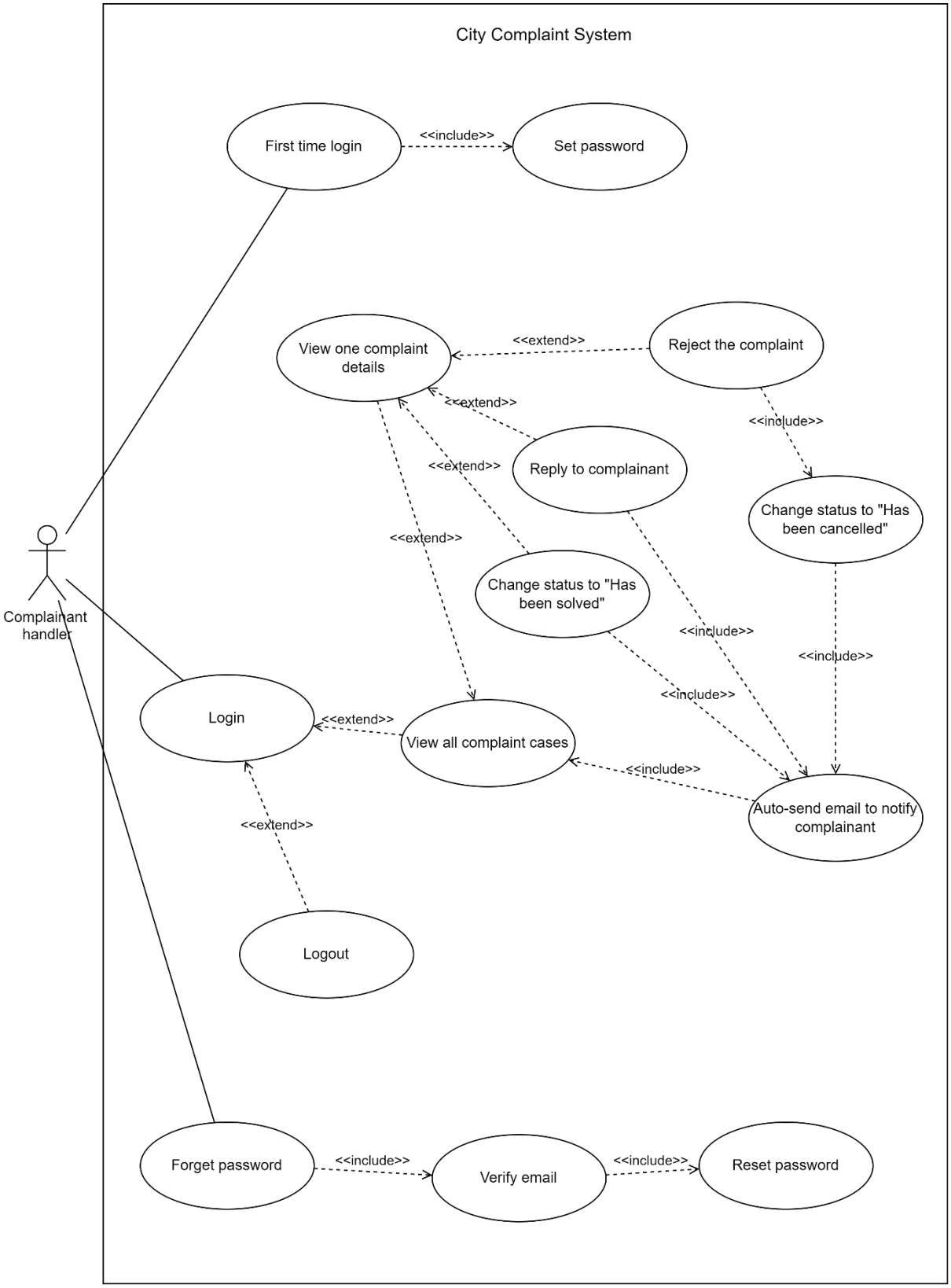


Figure 3.7 Use case diagram (Complaint handler)

3.3.3 Administrator

Figure 3.8 is the use case diagram for administrator actor. The administrator can The administrator can add and remove category, area and complaint handler. When adding complaint handler, the administrator should enter name and email address.

Besides that, the administrator can view the simple dashboard with the total number of complaints and break down of complaints by status. The administrator can also view the key statistics by specifying the period of time and choose between bar chart of number of complaints by category and bar chart of number of complaints by area.

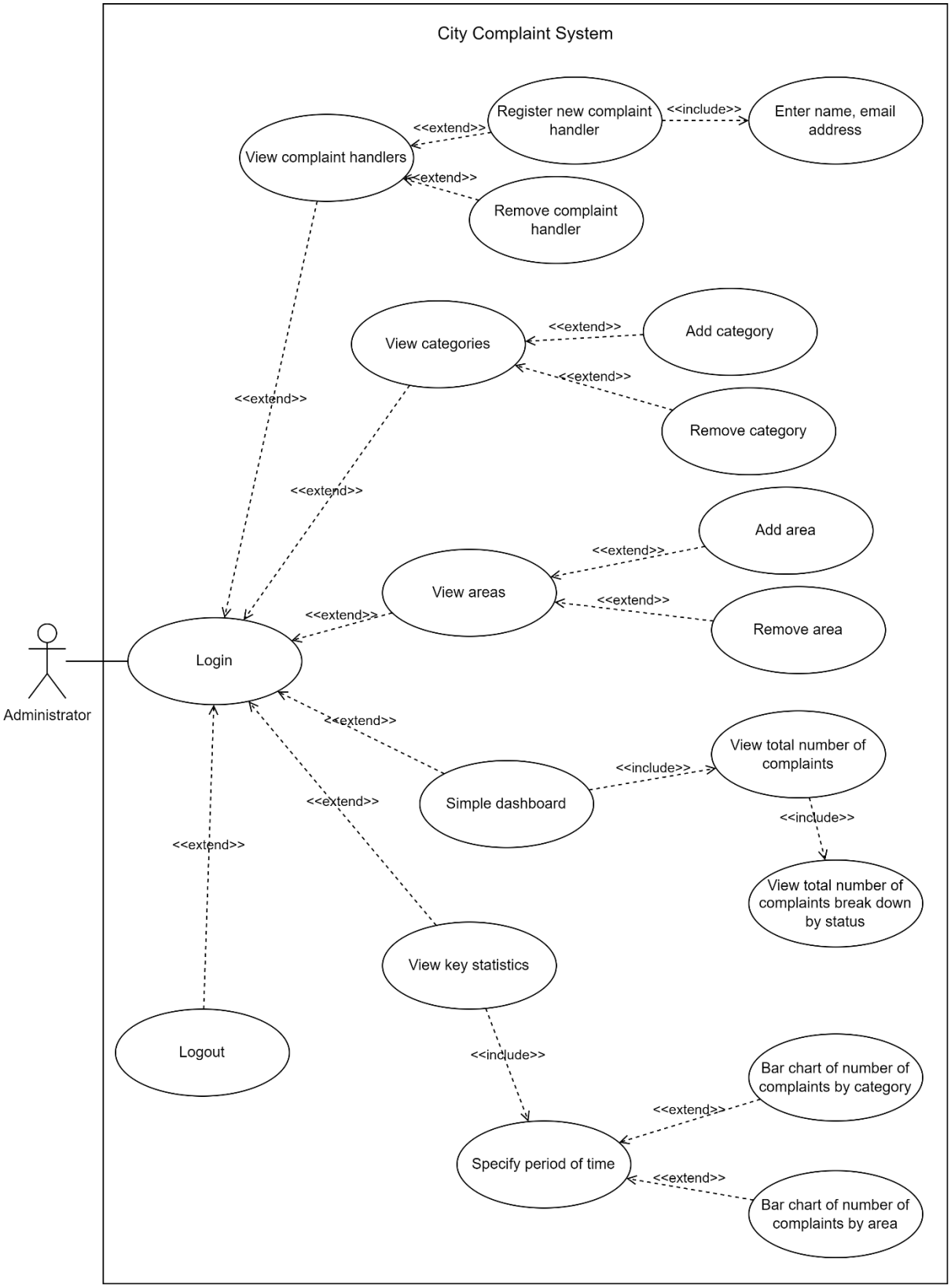


Figure 3.8 Use case diagram (Administrator)

3.4 Entity Relationship Diagram

The database used in this project is MongoDB. Figure 3.9 is the entity relationship diagram. One complainant can lodge zero or many complaints, but one complaint must be lodged by one complainant. One complaint handler can handle zero or many complaints, and one complaint can be handled by zero or one complaint handler. Complaint handler can also be administrator, depending on the “role” field. One complaint handler (only for role is “administrator”) can create zero or many areas and categories, but one area or one category can only be created by one complaint handler. One complainant can request for zero or many OTPs. Each complaint has an array called histories, which has a few fields itself.

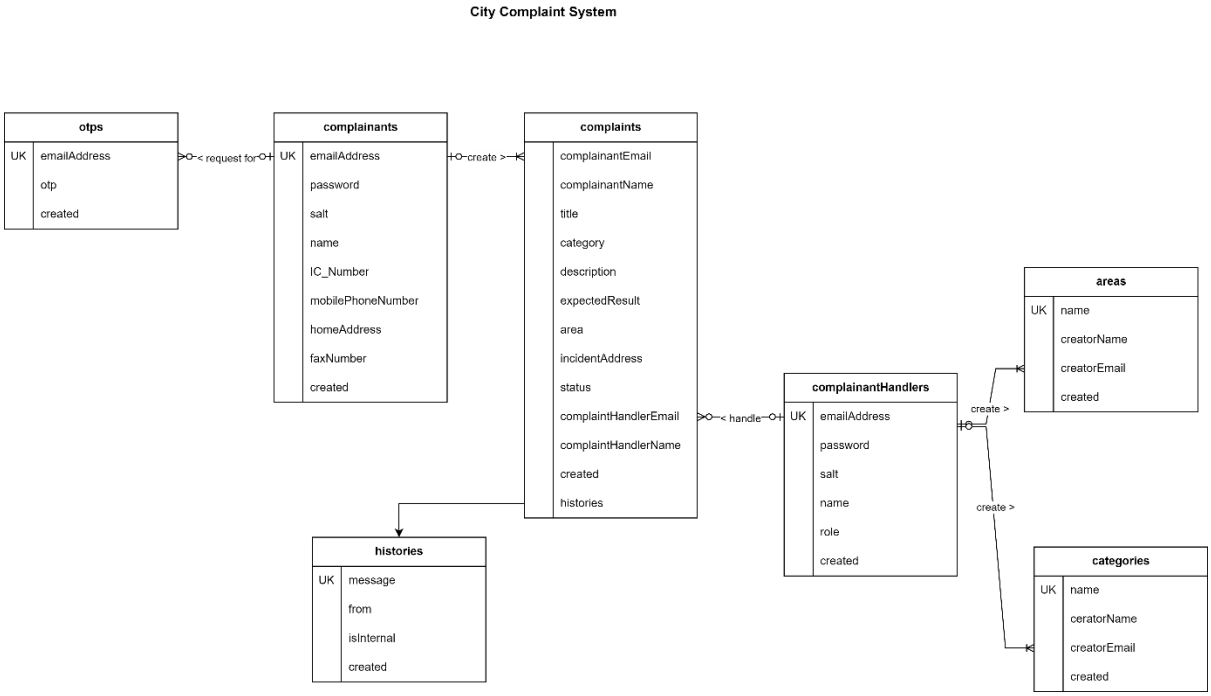


Figure 3.9 Entity relationship diagram

3.5 Libraries

The following libraries are used in the proposed work.

3.5.1 Express

Express is a JavaScript library used to create a server that can handles hyper text transfer protocol (HTTP) request and provide response. In the proposed work, Express is used to set up a server in the project at the server end (refer subtopic 4.1).

3.5.2 Axios

Axios is a JavaScript library that can initiate a HTTP request and handles the response from server. In the proposed work, Axios is used to perform GET, POST, PUT and DELETE request from the front end (refer subtopic 4.1)

3.5.3 Body-Parser

Body-Parser is a Node.js body parsing library which can parse the body of a HTTP request (Express Js, n.d.). In the proposed work, this is used at server end for parsing body of POST request sent from the front end.

3.5.4 JSON-Web-Token

JSON-Web-Token library is used to handle JSON web token (JWT) verification and creation. In the proposed work, JWT is used as for session management, so this library is used at the server end to handle the JWT token (Morgan, 2022).

3.5.5 Mongoose

Mongoose is a library to handle connection to MongoDB (Mongoose Js, n.d.). In the proposed work, the library is used at server end to connect to MongoDB.

3.5.6 Crypto

Crypto library is used for password encryption. The Password Based Key Derivation Function (pbkdf2) function is used to perform slow hash of the password with a salt (Geeks for geeks, 2021). It is used at the server end.

3.5.7 MD5

MD5 library used for performing Message Digest hashing. is used at front end during registration to hash the password before sent to the server.

3.5.8 Node Mailer

Node Mailer library is used to send email in Node.js (W3 Schools, n.d.). This library is used to implement the send email features at the server side, which is an essential part of sending OTP verification and sending notification email to complainant.

3.5.9 React-Router-Dom

React-Router-Dom is a library used for implementing page routing in React (W3 Schools, n.d.). In the propose work, this library is used to implement the navigation bar at the top of the page for the user interface.

3.5.10 React-Chartjs-2

React-Chartjs-2 is a library used for implement Chart.js feature in React, and is used to implement chart rendering feature (Maduabuchi, 2022) (React Chart Js 2, n.d.). In this proposed work, it is used to render the pie chart and bar chart at the statistics reporting page.

4. IMPLEMENTATION

4.1 Brief Explanation on the Structure of the System

The proposed work is built based on MongoDB - Express.js - React.js - Node.js (MERN) stack. As illustrated in Figure 4.1, the proposed work has three major components, namely React.js, Express.js and MongoDB. The main purpose of React.js is to render the webpage to all types of users, namely complainant, complaint handler and administrator. The main purpose of MongoDB is to store all data needed for the system. The main purpose of Express.js is to handle the flow of data and controls the logic process of the system between the React.js front-end and MongoDB database. In the following subtopics, the phrase "front-end" will refer to the React.js part, whereas "server-end" will refer to the Express.js part.

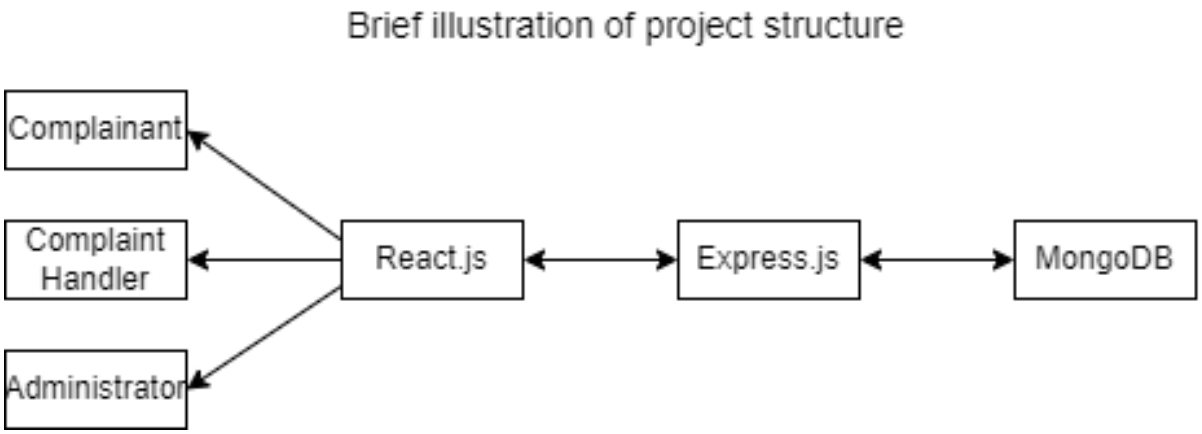


Figure 4.1 Brief illustration of project structure

4.2 Registration

There are three types of users in system of the proposed work, namely complainant, complaint handler and administrator. This subtopic will go through the registration for every type of user.

4.2.1 Register complainant

This subtopic discusses about the registration of complainant. Figure 4.2 is the complainant registration flow chart. First, the complainant will be presented with a registration

form which requires the complainant to provide personal details. After the complainant filled the form click get OTP button, his email address and name will be sent to the server. At the server side, after confirming that the user does not exist, an OTP will be sent to the complainant's email (will be further discussed in code snippet subtopic).

After that, the complainant will fill in the OTP and submit the registration form. After validation, if all the fields are valid, the form will be sent to the server. At the server side, after cross checking with the OTP sent, if the OTP provided is correct, the complainant information will be saved to the database. Finally, upon completion of registration, the complainant will be navigated to the login page.

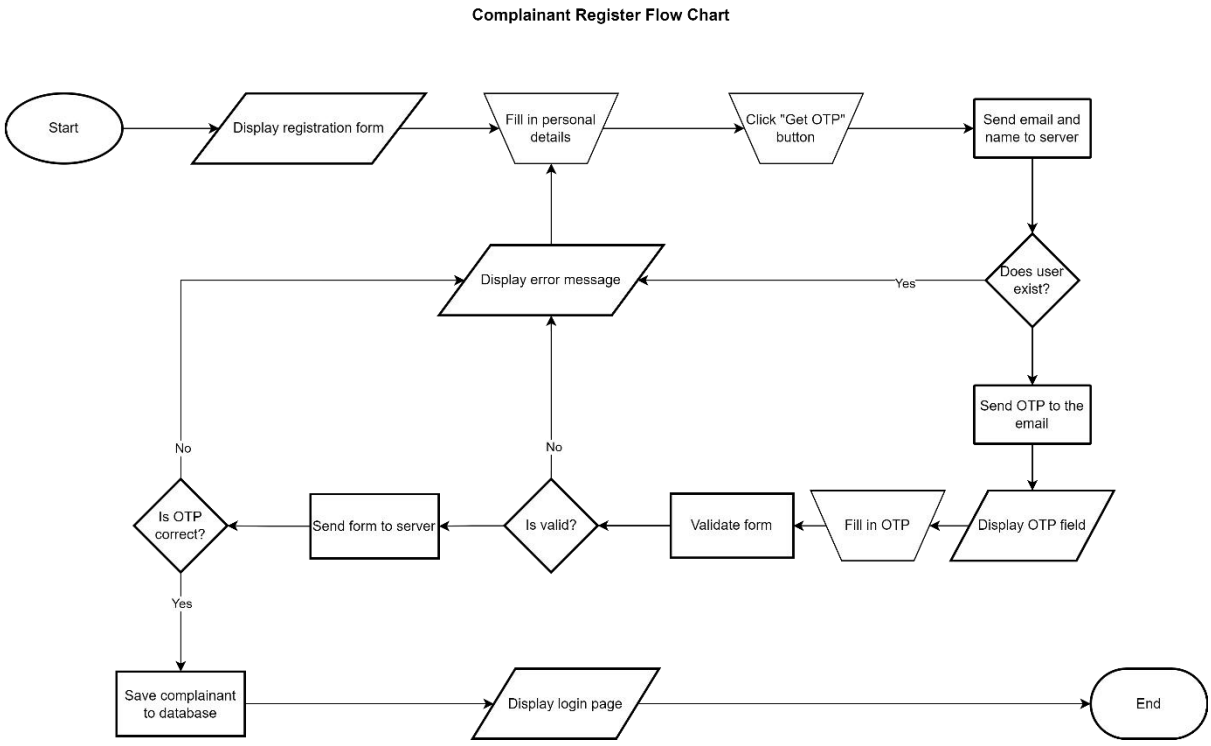


Figure 4.2 Complainant registration flow chart

Figure 4.3 is the screenshot of the registration page. To register a complainant account, the mandatory fields are a valid email address, password, name, IC number and mobile phone number, whereas the optional fields are home address and fax number. At this point of time, as shown in Figure 4.3, there is no register button, so complainant is unable to submit yet.

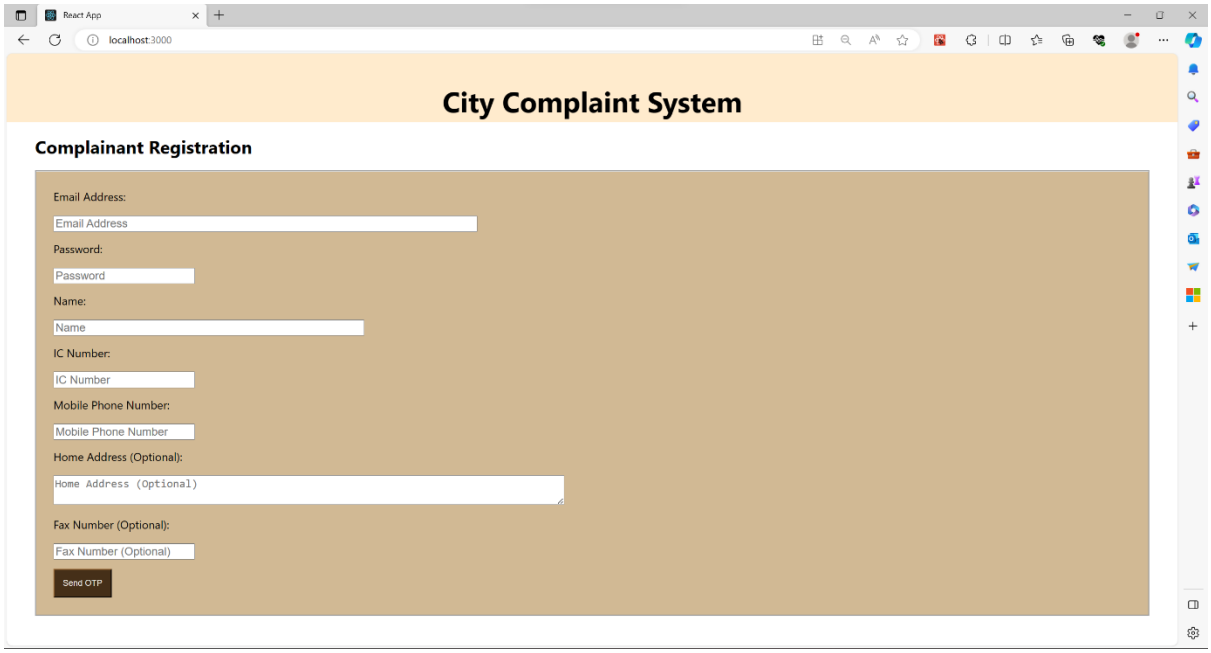


Figure 4.3 Register complainant

After filling in all fields, the complainant can click the send OTP button to request for OTP. As shown in Figure 4.4, after the OTP is sent to the complainant's email, the OTP field will appear. Only then, the register button will appear, so the complainant can click register button after he enters the OTP sent to his email.

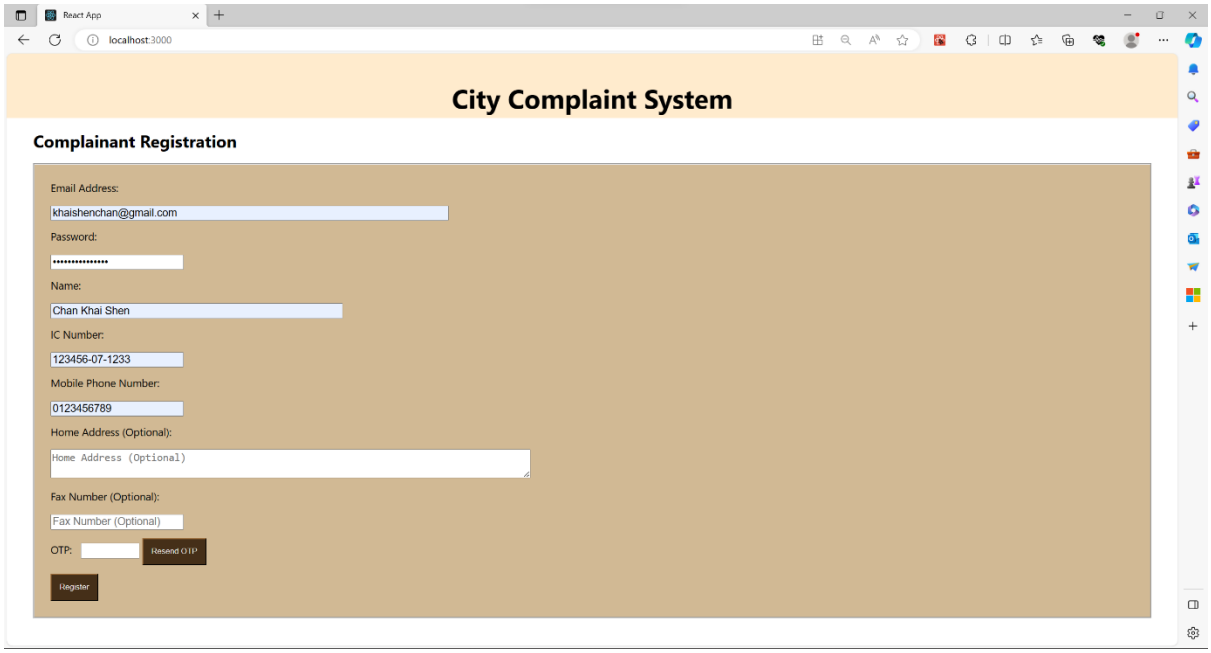


Figure 4.4 Register complainant (after OTP sent)

Figure 4.5 shows the verification email containing the OTP which is automatically sent to the complainant using node mailer. The detailed explanation will be placed in code snippet subtopic.

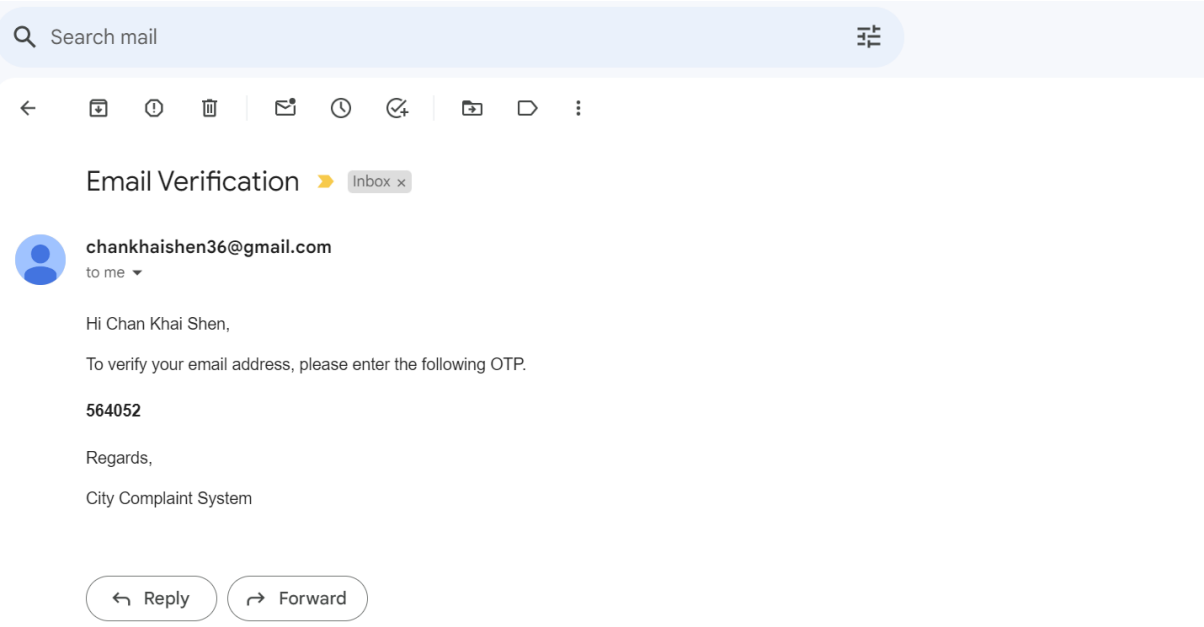


Figure 4.5 Verification email

Upon successful registration, as shown in Figure 4.6 the registration done page will appear, which will tell the complainant to go back to the login page.

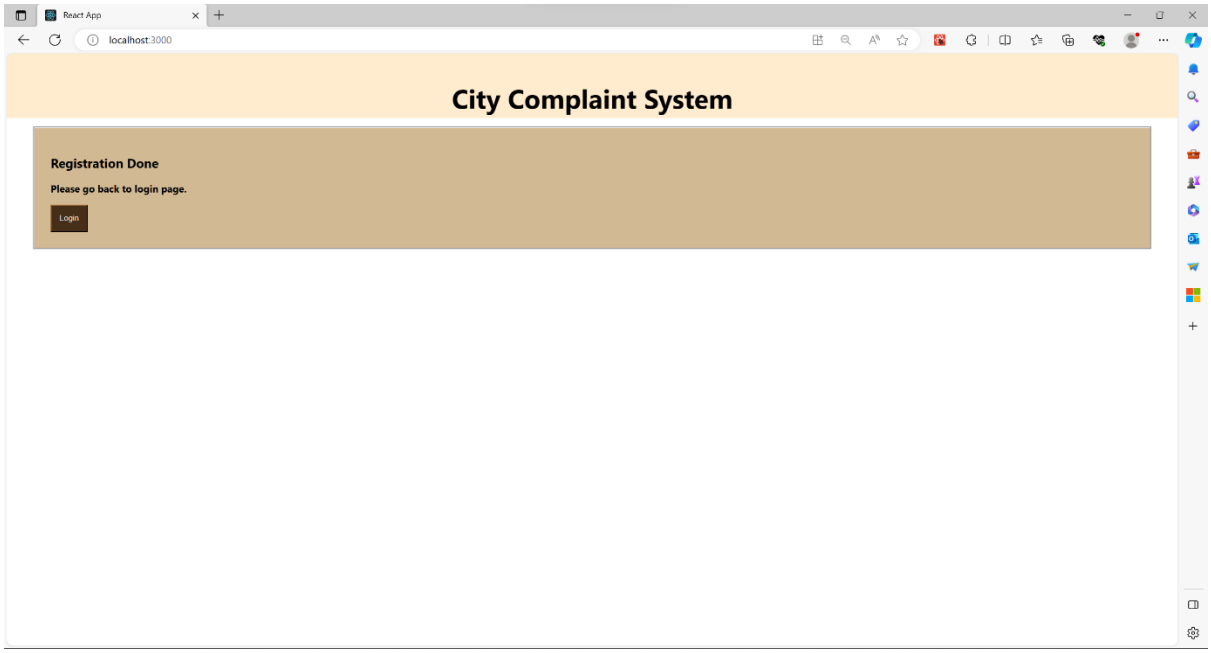


Figure 4.6 Registration done

4.2.2 Add or delete complaint handler

The complaint handler registration has two steps. These two steps are illustrated in Figure 4.7. First, the administrator will create the complaint handler account, then the complaint handler will set the password before his first login. This design is because the complaint handler is a person who can change the status of complaint case and view most of the information in the complaint cases that he handles, so the creation of complaint handler account should be controlled by the administrator. This will be further explained in subtopic 4.2.2(a) below.

Second, after the account has been created, the complaint handler will need to set the password of his account. This can be done by clicking on the forget password link on the login page before the complaint handler's first login. There will be more explanation on subtopic 4.2.3.

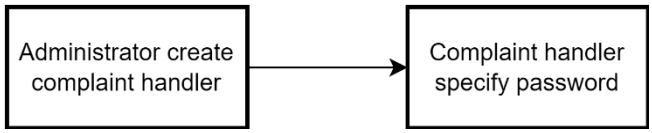


Figure 4.7 General flow of complaint handler registration

4.2.2(a) Administrator create complaint handler

Figure 4.8 is the flow chart for administrator create complaint handler, which is the step 1 as described in Figure 4.7. The administrator fills in email address and name of the new complaint handler. After the server received the information and cross checked the email address with all existing users stored in database, if the email address does not exist, the complaint handler information will be saved to the database. Then, an email will be sent to the complaint handler's email address to notify that an account has been created.

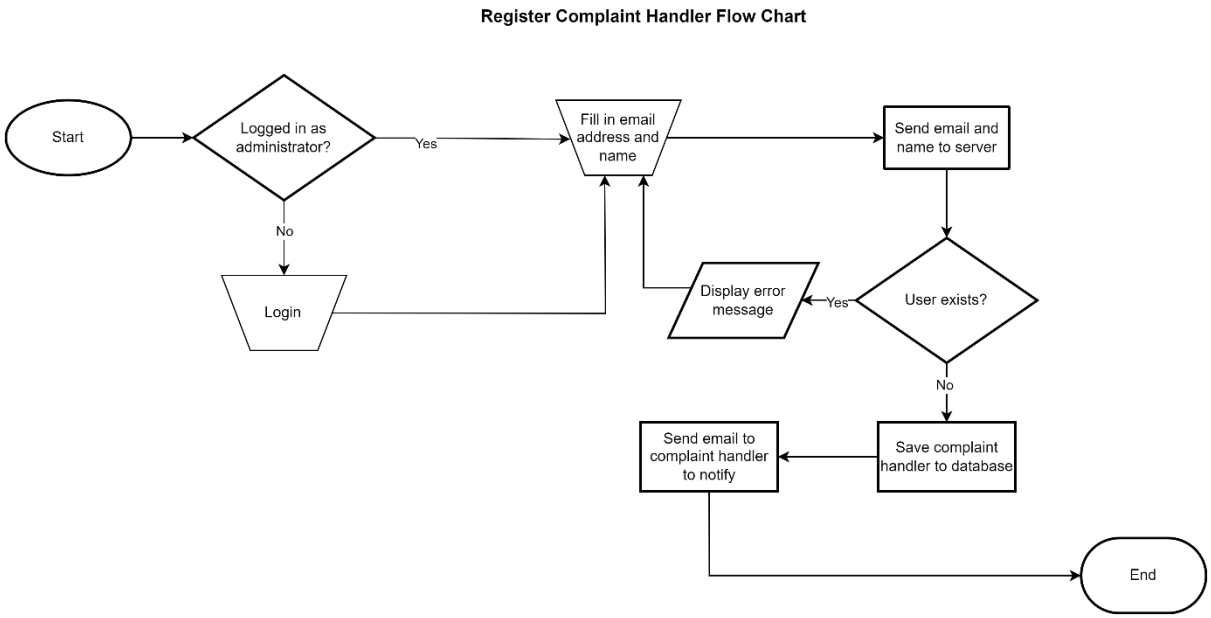


Figure 4.8 Add complaint handler flow chart (step 1)

4.2.2(b) Administrator delete complaint handler

Figure 4.9 is flow chart for administrator delete complaint handler. From a list of complaint handlers, the complaint handler selects one of the complaint handlers and clicks delete button. At the server side, the server will check the complaint case records to confirm whether the complaint handler has a case in status “Progressing”.

If the result is no, then the complaint handler information will be deleted from the database. The complaint case which is handled by the complaint handler, including the complaint handler’s name and email address, will not be deleted.

If the result is yes, then the deletion process will be stopped and an error message will be displayed. The administrator will have to find the complaint case handled by the complaint handler (details in subtopic 4.5.2(b)) and re-assign the case to other complaint handler.

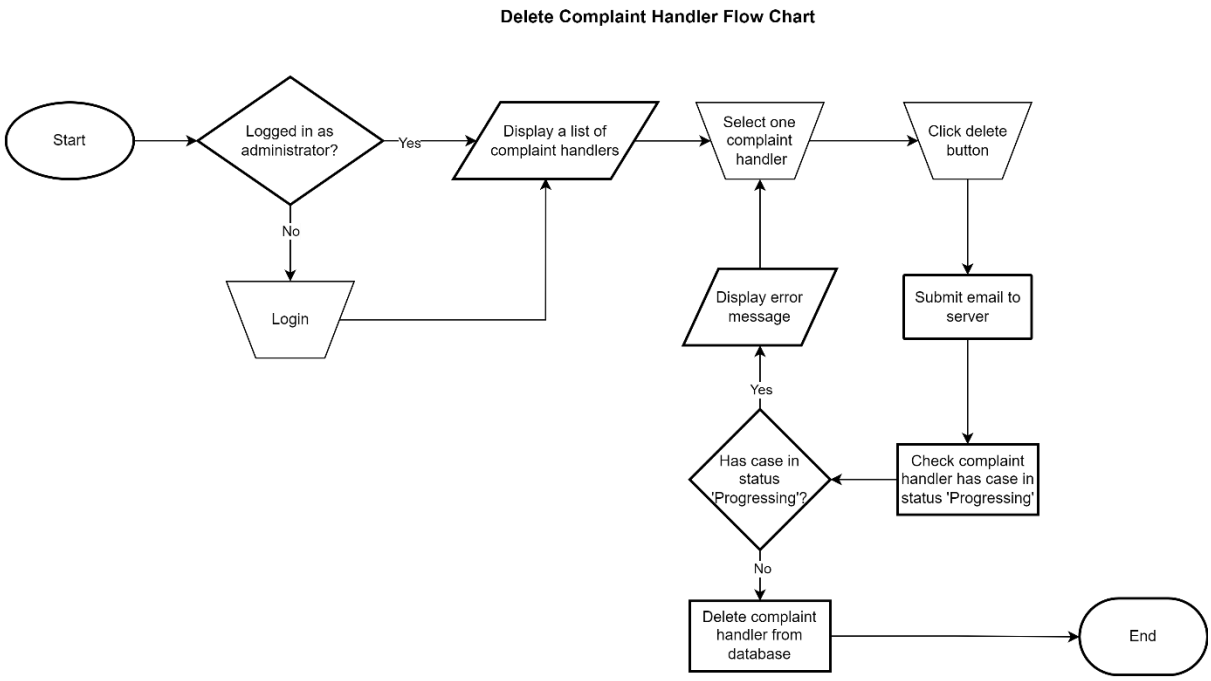


Figure 4.9 Delete complaint handler flow chart

4.2.2(c) Display / add/ delete complaint handler section (UI)

Figure 4.10 is the display / add / delete complaint handler section, which is part of manage system page. This page is only available to the administrator. The list of all complaint handlers will be displayed on this section. The administrator can find the delete button at each row of complaint handler. The interface for adding new complaint handler, including the name and email address field, as well as the add button, can be found beneath the list of complaint handlers.

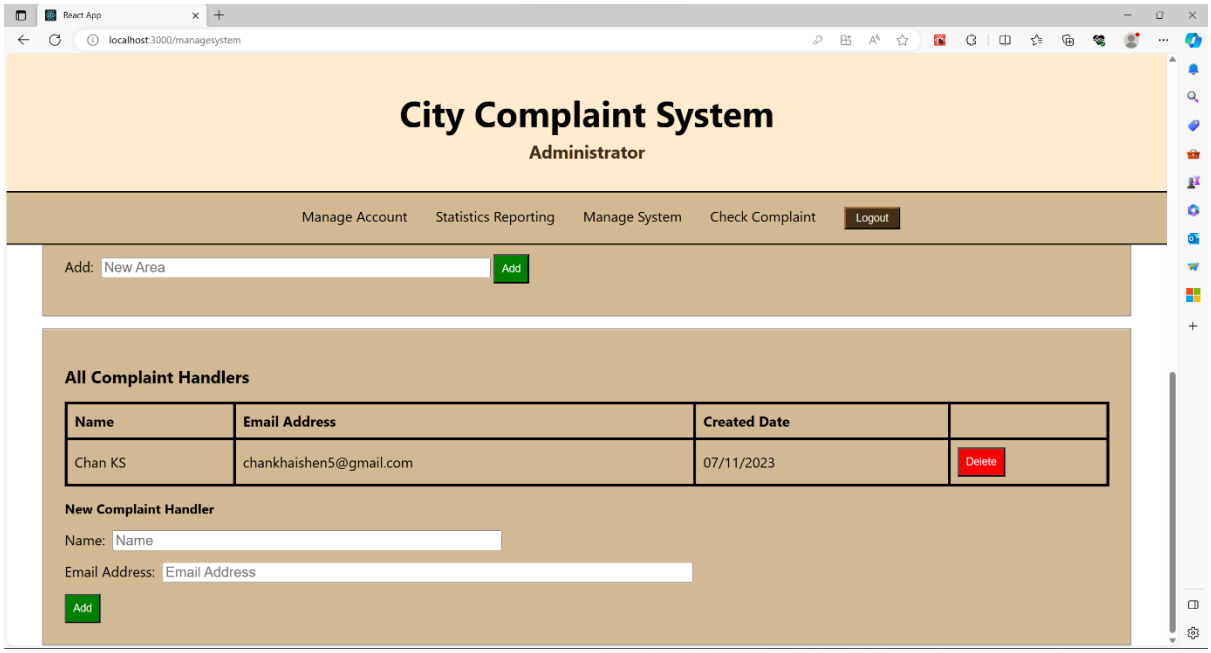


Figure 4.10 Add / delete complaint handler

4.2.3 Set / reset password

This subtopic discusses about set / reset password functionality, which is accessible from the forget password link on the login page. The functionality is used for all users to reset password when they have forgotten it, as well as used for complaint handler to set their password before their first login, which is the step 2 of complaint handler registration as described in the previous subtopic 4.2.2.

As depicted in the flow chart in Figure 4.11, the user will first enter his email address that is used to register his account. Then, after submitting to the server, the server will confirm whether the user is a valid user by cross-checking with the records in database. If the user exists, then an email containing the OTP will be sent to the user's email address (will be explained in code snippet subtopic).

After that, the OTP field and new password field will appear. After the user enters the OTP and the new password, if the OTP is correct, the new password will be saved.

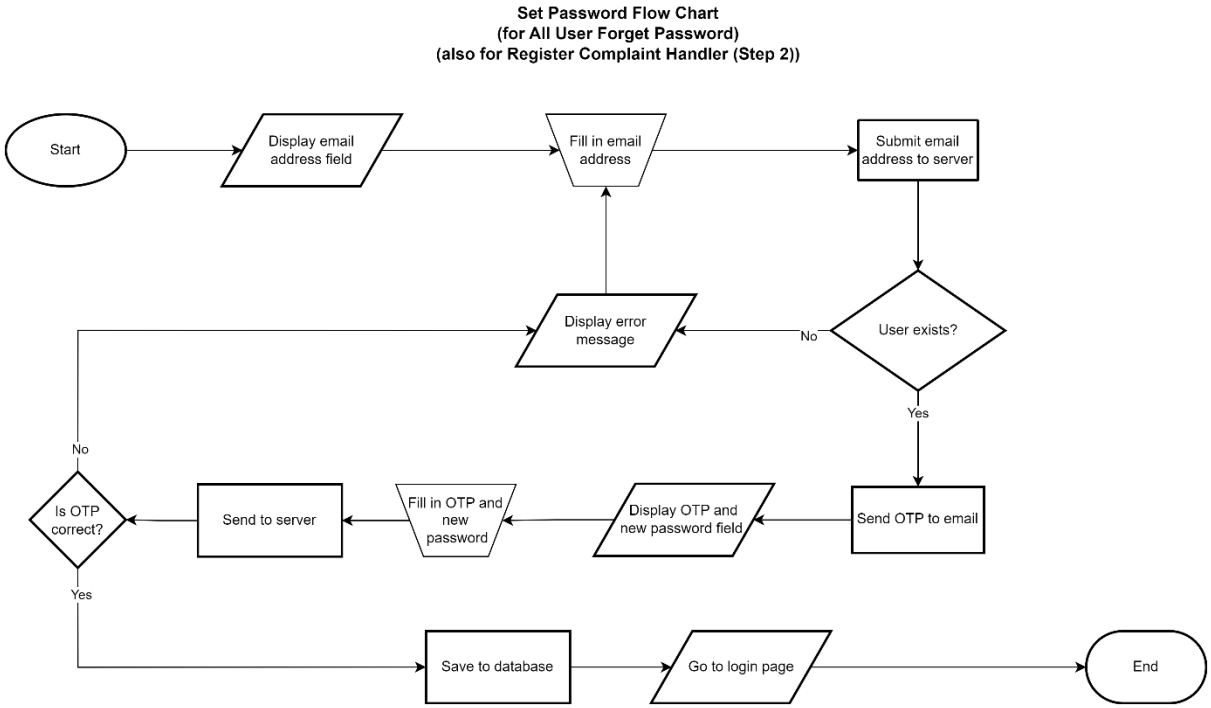


Figure 4.11 Set / Reset password flow chart

Figure 4.12 is the screenshot of set password page, whereas Figure 4.13 is the screenshot of set password page after the OTP is sent.

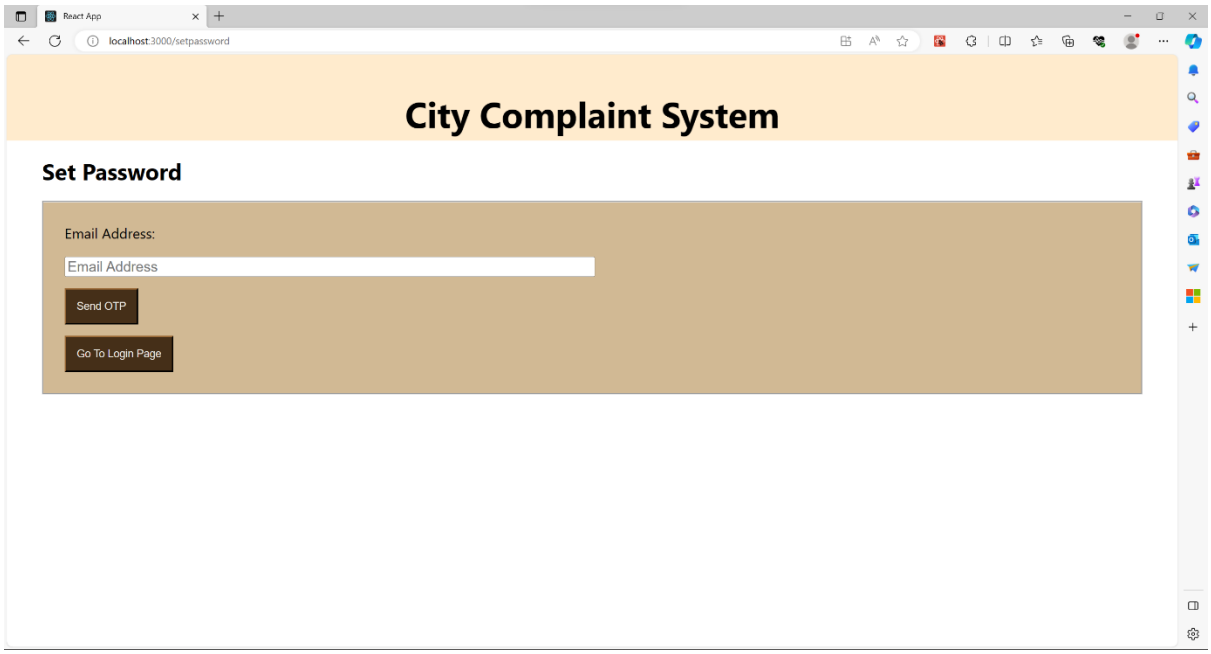


Figure 4.12 Set password page

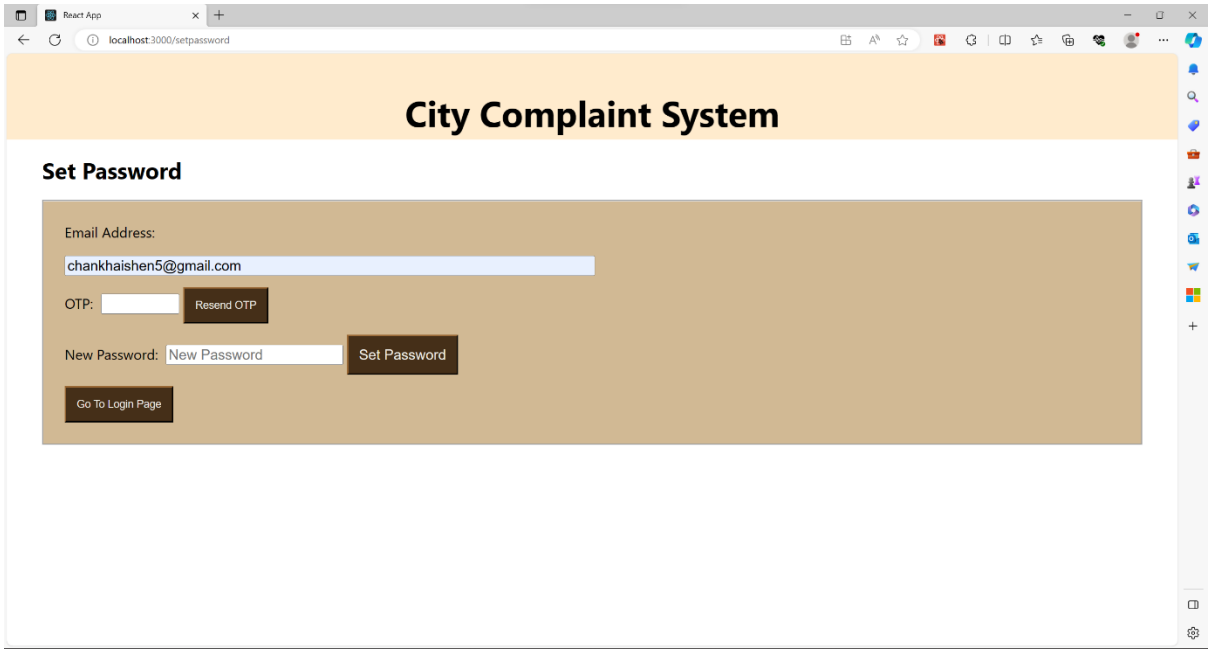


Figure 4.13 Set password page (after OTP sent)

4.2.4 Create administrator

The creation of administrator is restricted and can only be done be executing the code at the server. Figure 4.14 is the block of code to create administrator. This design is because administrator has access to crucial functionalities, such as create /remove complaint handler, which is possibly destructive, so its creation has to be restricted.

```
// The following block of code is used for creating new administrator
/*
const md5 = require('md5');
const crypto = require('crypto');

const password = 'My$Password123?';
const hashedPassword = md5(password);
const salt = crypto.randomBytes(16).toString('base64');

crypto.pbkdf2(hashedPassword, salt, 100, 256, 'sha-256', (err, derivedKey)=>{
    if (err != null) {
        console.log(err);
    }
    else {
        const securedPassword = derivedKey.toString('base64');
        const administrator = new complaintHandlerModel({
            emailAddress: 'cks123@mail.com',
            name: 'cks',
            password: securedPassword,
            salt: salt,
            role: 'administrator'
        });
        administrator.save().then(response=>{
            console.log('saved', response);
        }).catch(error=>{
            console.log('not saved', error);
        });
    }
});
*/
```

Figure 4.14 Create administrator

4.2.5 Password hashing and salting

From Figure 4.14 above, it also can be seen that the password is hashed twice. The first time is hashed by message digest 5 (MD5), and the second time is hashed using password-based key derivation function (PBKDF2), with a random salt generated by the crypto library, by using secured hash algorithm (SHA-256) for 100 iterations. According to Crack Station, password hashing with salt by using a slow hash function like PBKDF2 will make password cracking slower (Crack station, n.d.). Thus, this is one of the security aspects which is incorporated into the proposed work.

4.3 Login

This subtopic talks about access control, user login and the landing page, which generally applies to each user that uses the system of the proposed work.

4.3.1 Access control before login

As part of access control, the system requires every user to login to be able to gain access. Figure 4.15 shows the code for implementing access control at front-end. The system evaluates whether there is a token stored in the local storage of the browser and sends the token to server-end through a get method to see whether the token is a valid token. The token is placed in the “Authorization” header If there is no token or the token has expired, then login page will be rendered. The conditional rendering of login page based on the availability of token is one way which can be used to implement in React, which is token-based authentication (Morgan, 2022). This is to render login page first if the user who has not yet login tries to go to a page which requires login, as well as saving the user’s authentication token for future use (Morgan, 2022).

```
const [hasToken, setHasToken] = useState(localStorage.getItem('token') != null);

console.log(env.SERVER_URL);

useEffect(()=>{
  if (hasToken) {
    axios.get(`${env.SERVER_URL}/verifytoken`, {
      headers: {
        'Authorization': `Bearer ${localStorage.getItem('token')}`
      }
    }).then(result=>{
      console.log('token is valid', result);
      setHasToken(true);
    }).catch(error=>{
      console.log('token is invalid', error);
      localStorage.removeItem('token');
      localStorage.removeItem('role');
      localStorage.removeItem('name');
      localStorage.removeItem('emailAddress');
      setHasToken(false);
    })
  }
});

if (window.location.href === `http://localhost:${env.PORT}/setpassword`)
  return (
    <SetPassword />
  );

if (!hasToken)
  return (
    <Login setHasToken={setHasToken}/>
  );
```

Figure 4.15 Front-end access control

Figure 4.16 is the code for “/verifytoken” get method at server-end, which is the section of code which responses to the request to check the validity of token made from the code in Figure 4.15 above. The code will call a function named verifyToken() to verify the validity of the token, which will be explained below.

```
app.get('/verifytoken', verifyToken, (req, res)=>{
    if (req.user != null) {
        console.log('verify token: Verified');
        res.status(200).send();
    }
    else {
        console.log('verify token: Not verified');
        res.status(401).send();
    }
});
```

Figure 4.16 "/verifytoken" get method

Figure 4.17 is the code for verifyToken() function. In this project, JavaScript Object Notation (JSON) Web Token (JWT) is used, so the jwt library is used to handle the verification.

```
const verifyToken = (req, res, next) => {
    var authorization = req.headers['authorization'];
    console.log('verifyToken', authorization);

    if (authorization === undefined) {
        console.log('verifyToken: Authorization header is empty')
        res.status(401).json({
            message: "Authorization header is empty"
        });
        return;
    }

    token = authorization.split(' ')[1];
    console.log('verifyToken', token);

    if (token === undefined) {
        console.log('verifyToken: No token')
        res.status(401).json({
            message: "No token provided"
        });
        return;
    }

    jwt.verify(token, process.env.JWT_SECRET, (error, decoded) => {
        if (error != null) {
            console.log('verifyToken: Fail to authenticate', error);
            res.status(401).json({
                message: "Fail to authenticate token"
            });
            return;
        }

        req.user = decoded;
        next();
    });
}
```

Figure 4.17 verifyToken() function (server-end)

4.3.2 Login page and landing page

Figure 4.18 is the screenshot of login page. No matter which type of user, the user has to provide email address and password for login. From the login page, the user can also go to the registration page (complainant only) and forget password page (all users). The forget password page is also the set password page for complaint handler registration.

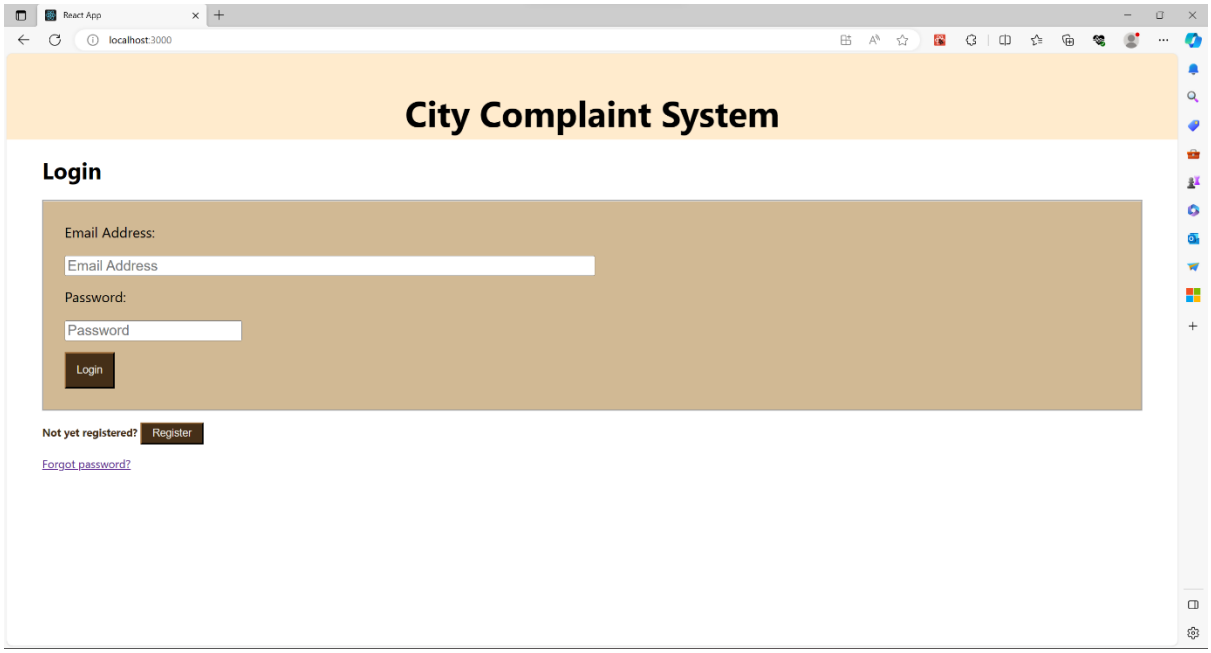


Figure 4.18 Login page

Upon successful login, the JWT token, as well as the user's email address, name and role, will be returned. This information is stored in local storage. As shown in Figure 4.19, based on the role stored in local storage, different layout will be rendered for different users, depending on their role. This is done by using conditional rendering by applying logic operator (React, n.d.).

```
if (localStorage.getItem('role') === 'complainant')
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element = {<ComplainantLayout/>} >
          <Route index element = {<ManageAccount/>} />
          <Route path="lodgecomplaint" element = {<LodgeComplaint/>} />
          <Route path="checkcomplaint" element = {<CheckComplaint/>} />
          <Route path="*" element = {<NoPage/>} />
        </Route>
        <Route path="*" element = {<NoPage/>} />
      </Routes>
    </BrowserRouter>
  );
else if (localStorage.getItem('role') === 'complaint handler')
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element = {<ComplaintHandlerLayout/>} >
          <Route index element = {<ManageAccount/>} />
          <Route path="checkcomplaint" element = {<CheckComplaint/>} />
          <Route path="*" element = {<NoPage/>} />
        </Route>
        <Route path="*" element = {<NoPage/>} />
      </Routes>
    </BrowserRouter>
  );
else if (localStorage.getItem('role') === 'administrator')
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element = {<AdministratorLayout/>} >
          <Route index element = {<ManageAccount/>} />
          <Route path="statisticsreporting" element = {<StatisticsReporting/>} />
          <Route path="managesystem" element = {<ManageSystem/>} />
          <Route path="checkcomplaint" element = {<CheckComplaint/>} />
          <Route path="*" element = {<NoPage/>} />
        </Route>
        <Route path="*" element = {<NoPage/>} />
      </Routes>
    </BrowserRouter>
  );
```

Figure 4.19 Access control of different user

By default, all users will be navigated to manage account page, which is the default landing page after login. As shown in Figure 4.19, on manage account page, the user's email address, name and role will be displayed. The user can then navigate to different page from the top navigation bar.

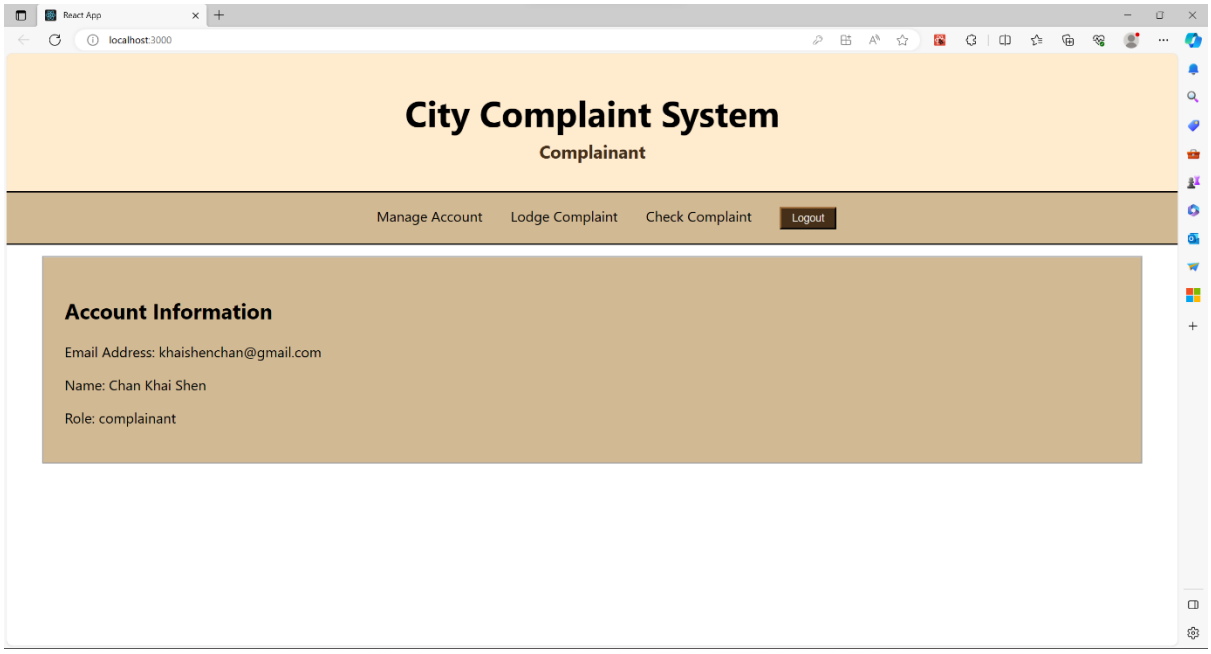


Figure 4.20 Landing page after login

4.4 Lodge Complaint (Available to complainant only)

Figure 4.21 is the lodge complaint flow chart. This feature is available to complainant only. The complaint form is displayed and the complainant fills in the complaint description field. The form fields will go through validation, if all fields are valid, then the form is sent to the server. At the server, the initial status of the case will be set as "Received". After the complaint case is saved to database, the complaint summary will be sent to the complainant's email address. After that, the same summary will also be displayed on screen.

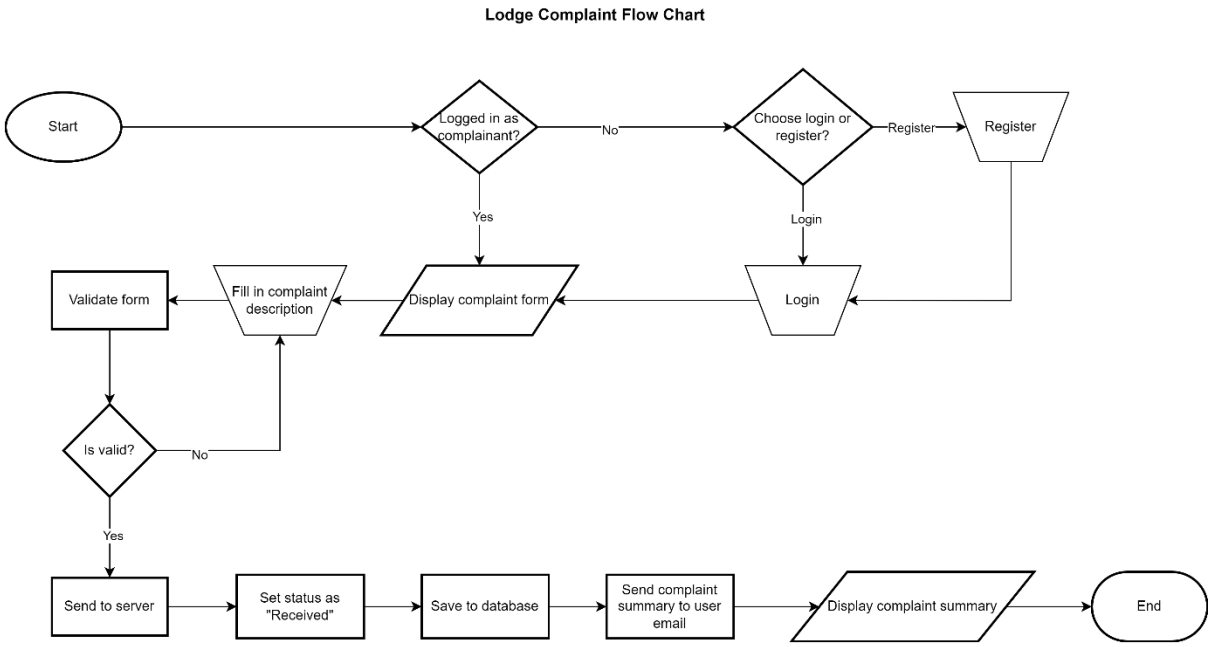


Figure 4.21 Lodge complaint flow chart

4.4.1 Lodge complaint page

Figure 4.22 is the lodge complaint page. The mandatory fields are title, category, description and area, whereas the optional fields are expected result and incident address. The category and area fields are selectable. The options of category and area are loaded dynamically (detailed explanation on subtopic 4.4.2). In React, each input field in a form is kept track by using the use state hook, in order to ensure a single source of truth in the form data (W3 Schools, n.d.). After the user clicks submit, the form will be submitted.

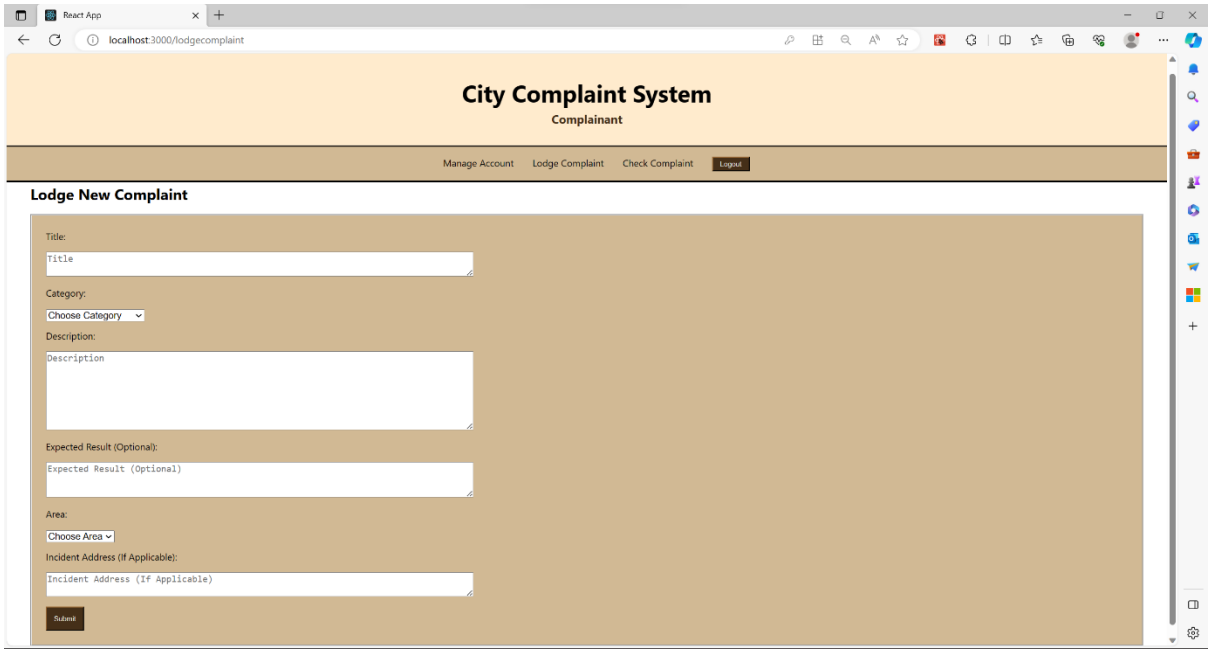


Figure 4.22 Lodge complaint page

Upon successful submission, the complaint details will be displayed on screen, as shown in Figure 4.23. By default the status is “Received”. The unique object ID generated by database, is used as the case ID and is displayed to the complainant.

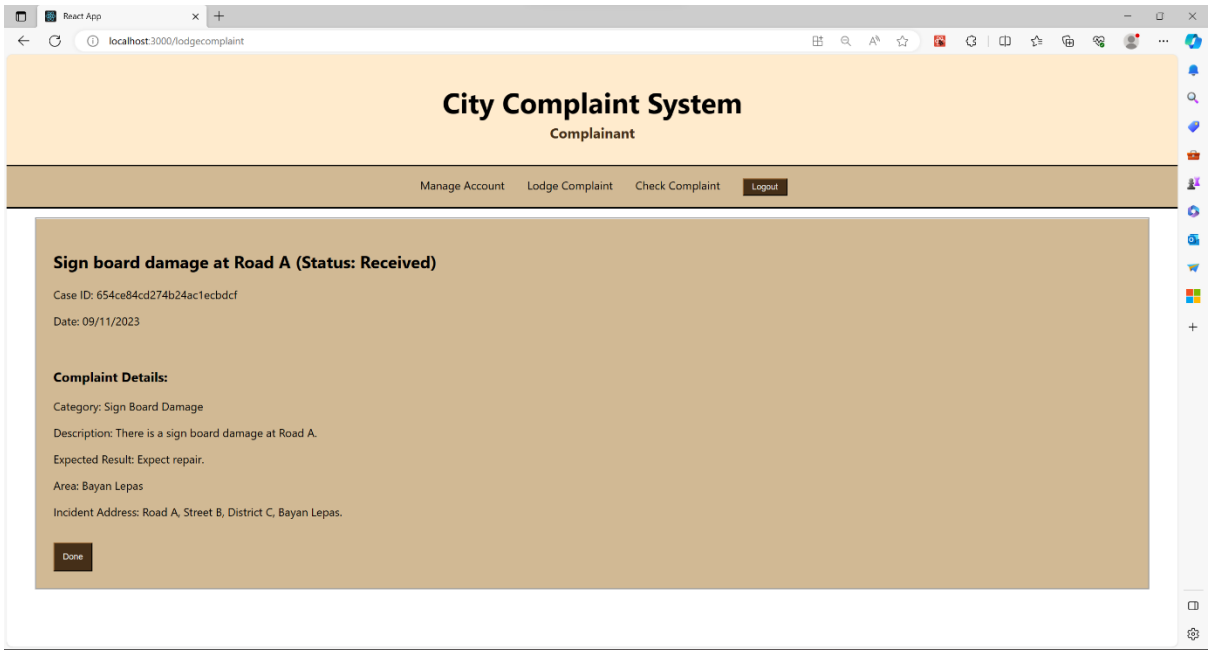
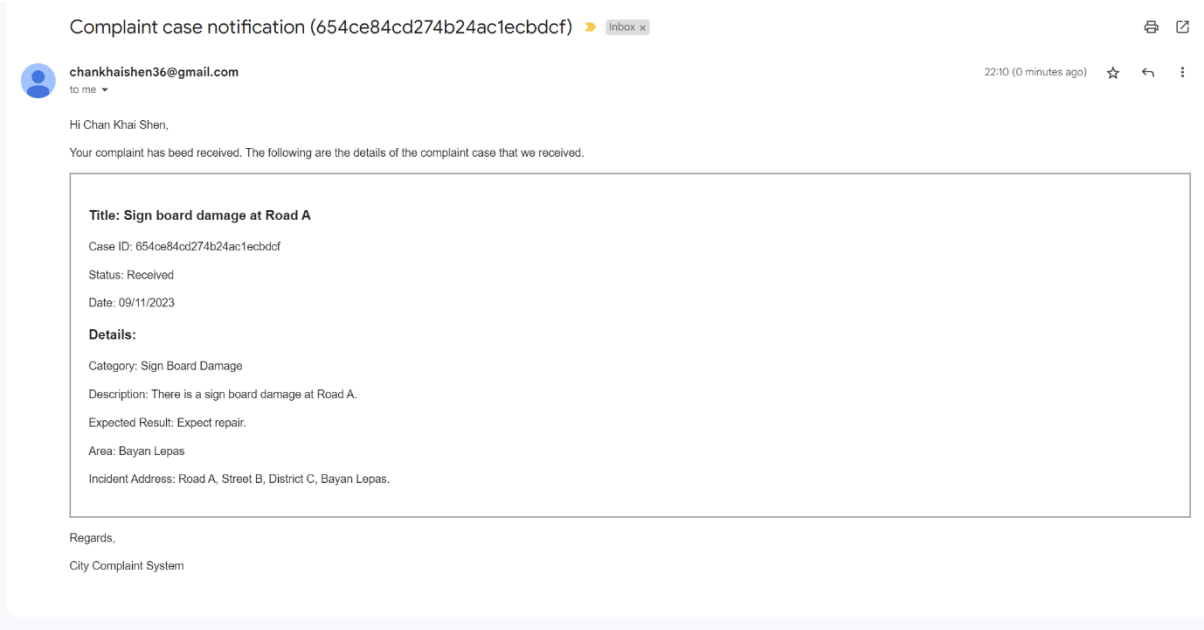


Figure 4.23 Complaint summary page

The similar information is sent to the complainant email, as shown in Figure 4.24. This is to give the complainant a record of the complaint case for future reference.



Complaint case notification (654ce84cd274b24ac1ecbdcf) Inbox ×

chankhaishen36@gmail.com
to me

22:10 (0 minutes ago)

Hi Chan Khai Shen,

Your complaint has beed received. The following are the details of the complaint case that we received.

Title: Sign board damage at Road A

Case ID: 654ce84cd274b24ac1ecbdcf

Status: Received

Date: 09/11/2023

Details:

Category: Sign Board Damage

Description: There is a sign board damage at Road A.

Expected Result: Expect repair.

Area: Bayan Lepas

Incident Address: Road A, Street B, District C, Bayan Lepas.

Regards,

City Complaint System

Figure 4.24 Complaint summary (email)

4.4.2 Administrator add / delete the area and category options

To provide flexibility to the system, the options of area and category is not hardcoded, but will be rendered dynamically. Instead, it can be set by the administrator and fetched during the rendering of lodge complaint form. Figure 4.25 is the add / delete category and area section on manage system page, which is available to administrator only.

Both are them are similar, so will be explained together. As shown in Figure 4.25, the list of category / area is displayed to the administrator, which the administrator can view the information of the creation date and creator information of each category / area. The administrator can delete any category / area by clicking on the delete button at the rightmost column on that row. As shown in Figure 4.25, the administrator can add new category / are by entering the name of the new category / area into the new category field beneath the list of category / area and clicking on the add button. If there is no collision with the existing category / area, then the new category / area will be sent to the server and saved to database. Finally, the change will reflect on the list on-screen.

Any changes in the category / area will only affect the future complaint form. So, the previous and existing complaint cases which had already used the deleted category / area, will still be using that category / area. This also applies to statistics reporting since statistics reporting is generated based the existing complaint cases.

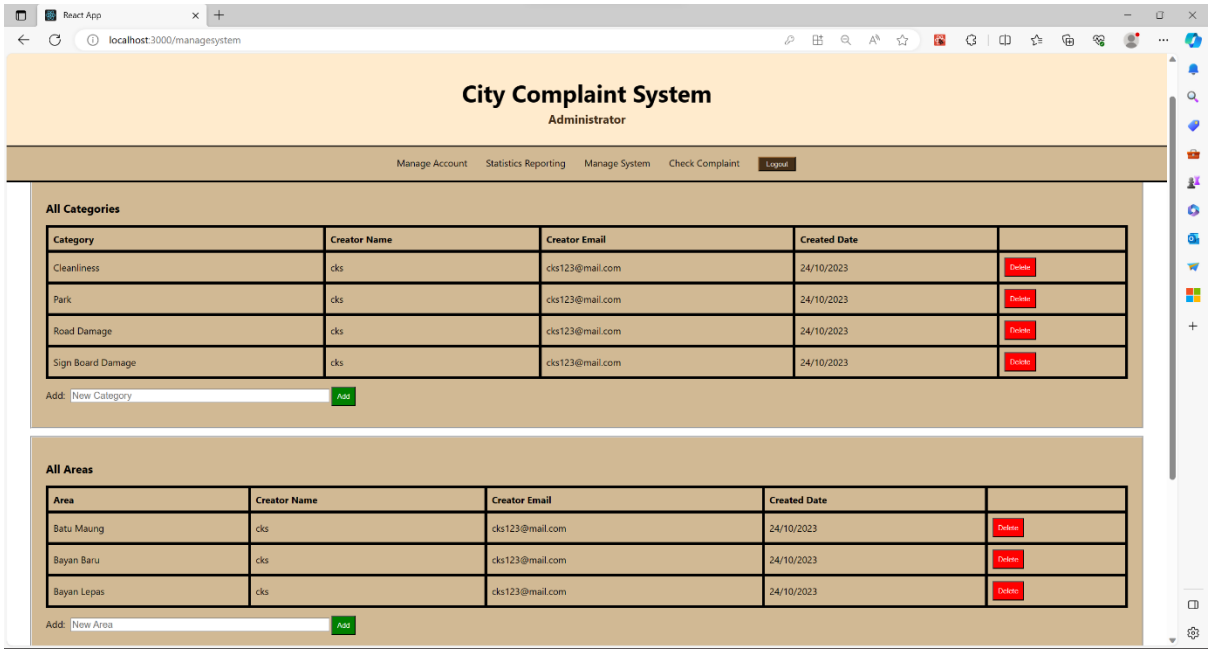


Figure 4.25 Add category / area section

4.5 Check Complaint

This subtopic discusses about check complaint feature, which includes check complaint page, search, filter and sort feature, complaint details page, histories section, add message feature, change status feature and assign complaint handler to a case.

4.5.1 Check complaint page

The complainant can check complaint from the check complaint page, as shown in Figure 4.26. The list of complaints is listed dynamically in tabular form. In React, multiple similar components can be rendered dynamically from an array by using map() function and a unique key for each item (React, n.d.). In this case, the case ID, which is the unique object identifier generated by the database is used as the unique key. By default, all complaints are listed. The complainant can only see his own complaint, whereas complaint handler can only see complaints which he is the complaint handler of the case or does no have complaint handler, whereas administrator can see all complaint cases. Figure 4.26 is the complaint handler view.

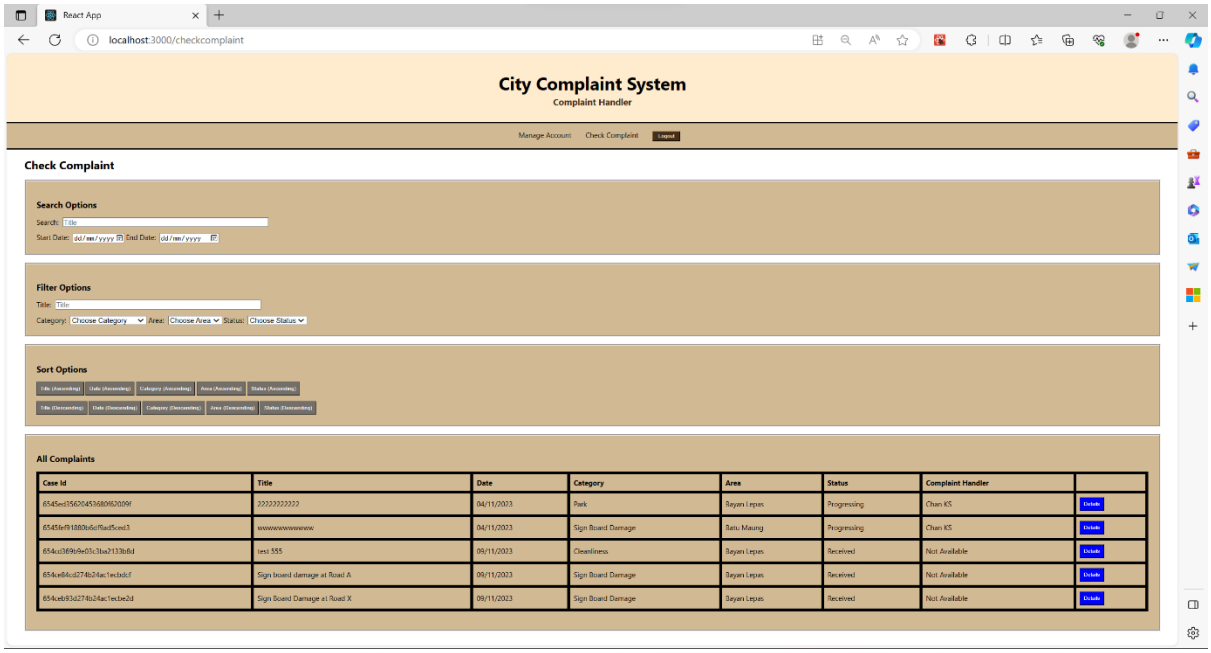


Figure 4.26 Check complaint page

4.5.2 Search, filter and sort

This subtopic user is split into two parts, the first part is available to all types of users, whereas the second part is available to administrator only.

4.5.2(a) Search, filter and sort (Available to all types of users)

If the list of complaint is very long (more applicable to complaint handler and administrator), the user can also limit the range of date at the search option. The user is able to use the search bar to specify a complaint title which is not limited by the date range. As for filter options, the user can filter the complaint based on category, area or status by selecting from the dropdown, as shown in Figure 4.27. The filtering is done by comparing the filter text and the text in the relevant field, and hide the rows which the texts do not match (W3 Schools, n.d.). The user can also use the input at the filter options to filter the complaints by specifying substring in complaint title. As for sort options as seen in Figure 4.27, the user can sort by title, date, category, area and status by ascending or descending order by clicking on the relevant sort button. The sorting is done by comparing alphabetical order or date value of the relevant fields in neighbouring rows and swapping the rows if necessary (W3 Schools, n.d.).

Figure 4.27 Search, filter and sort options

4.5.2(b) Search complaint handler (Available to administrator only)

Figure 4.28 is the search options for administrator. This is related to the delete complaint handler operation described in subtopic 4.2.2(b). When the administrator specifies the name or email address of the target complaint handler, all complaints handled by the complaint handler will be included in the list of complaints.

Figure 4.28 Administrator search option

4.5.3 Complaint details page

As shown in Figure 4.26 above, on the check complaint page, the user can click on the details button on the rightmost column of the row of any complaint to view the details. Figure 4.29 is the complaint details page from complaint handler view. The summary of the

complaint, which includes complaint details, complainant personal information, date of received and current status will be displayed in point form.

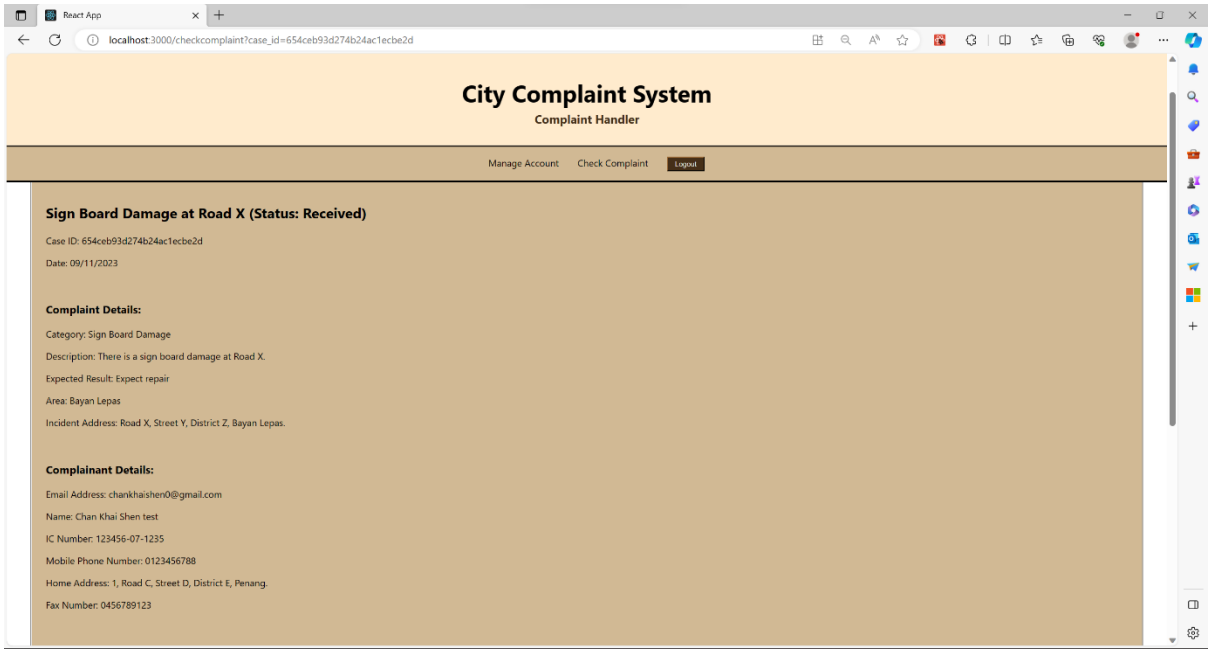


Figure 4.29 Complaint details page

4.5.4 Histories section

Figure 4.30 is the history section in complaint details page. As shown in Figure 4.30, the histories section will display the past histories, which includes change of status, change of complaint handler (only available to complaint handler and administrator) and text message from both complainant side and complaint handling side. The change of complaint handler cannot be seen by the complainant.

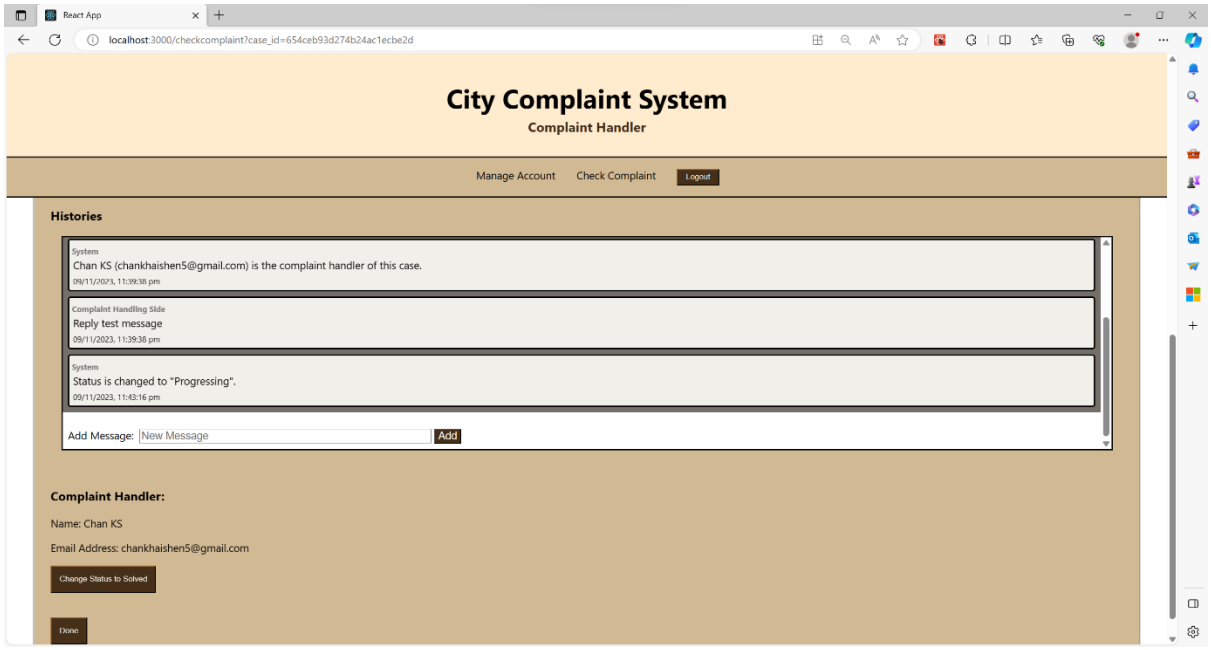


Figure 4.30 Histories section

4.5.5 Add message

Figure 4.31 is the add message flow chart. The add message feature is available to complainant (only for his complaint case), complaint handler (only for his complaint case) and administrator (for all cases). The user can enter the message on the new message field beneath histories section, as shown in Figure 4.32 After the user clicks the add message button, the message and the case ID will be sent to the server. The server will check the user's credentials and the case information. If the user meets the condition (as mentioned above) to add message to that case, the message will be saved to database. If the complaint case does not have complaint handler message is from complaint handler or administrator, then the user will be set as the complaint handler of the case. If the message is added by complainant, it is labelled as "complainant side", else if the message is added by complaint handler or administrator, it is labelled as "complaint handling side". If the message is labelled as "complaint handling side", a notification email will be sent to the complainant to notify the new message, as shown in Figure 4.33. Finally, the message will appear in the histories section, as shown in Figure 4.32.

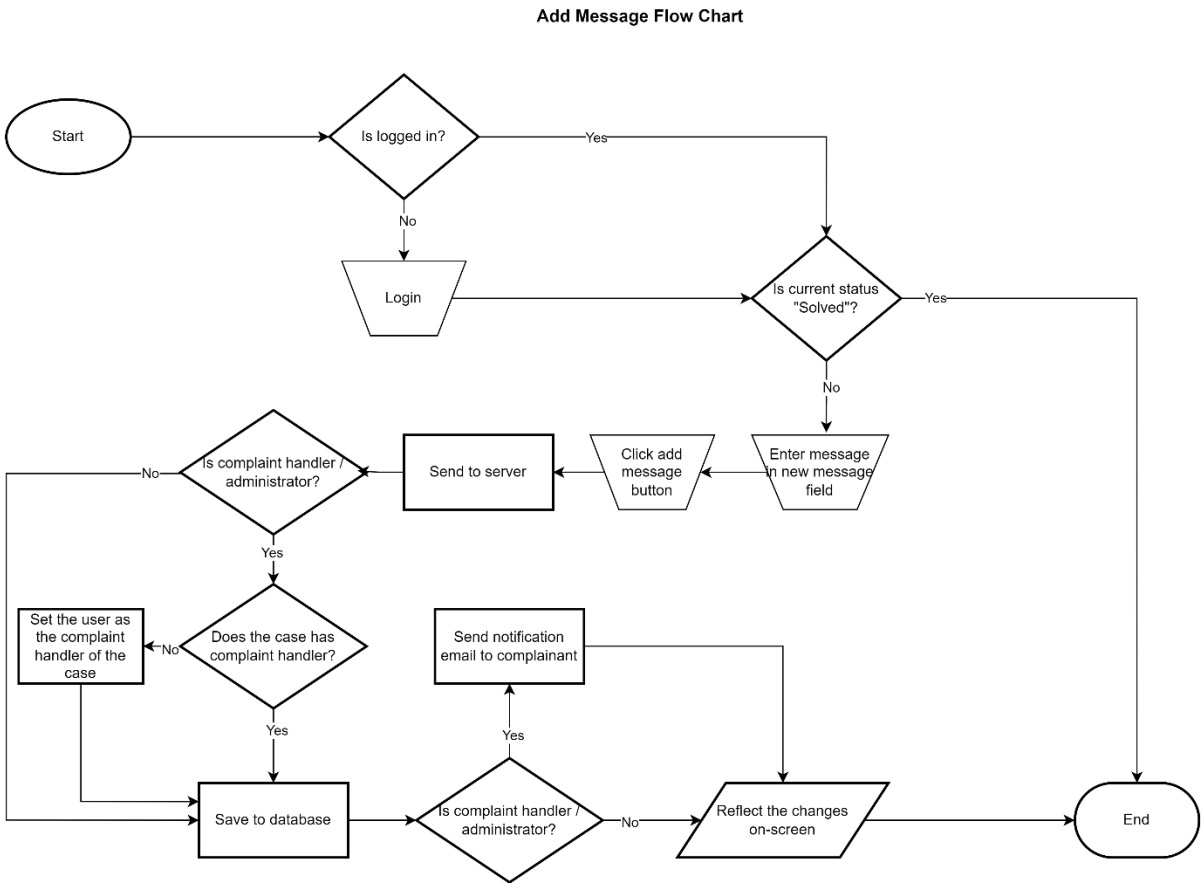


Figure 4.31 Add message flow chart

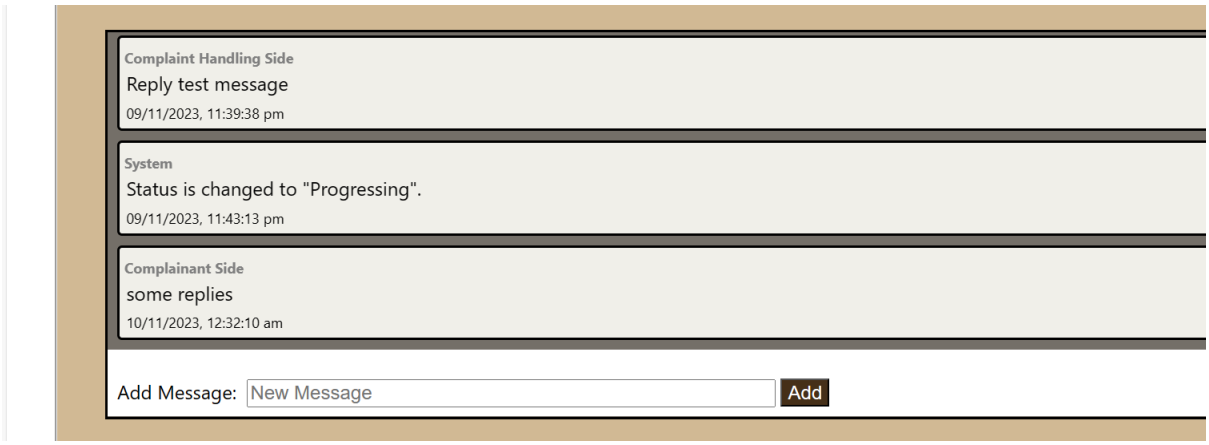


Figure 4.32 Add message bar

Figure 4.33 Notification email (New message from complaint handling side)

4.5.6 Change status

Figure 4.34 is the change status flow chart. Only the complaint handler who is in-charge of the case and the administrator can change the status of the complaint case. The change status button, which is beneath the complaint handler information section as shown in Figure 4.35, is rendered according to the current status of the complaint case. If the status is “Received”, “Change to Progressing” button will appear, else if the status is “Progressing”, “Change to Solved” button will appear, else if the status is ‘Solved”, there will be no button. The order of change of status is depicted in Figure 4.36. It is a one-way journey, and reversing of status is not allowed in the system.

After the user clicking the change status button, the new status and the case ID will be sent to the server. At the server, a process of validation is carried out. If the eligibility of the user (as mentioned above) is valid, and the status is valid, then the change of status of the complaint case will be updated to database. If the case does not have a complaint handler, then the complaint handler / administrator who changes the status will be set as the complaint handler. After that, a notification email will be sent to the complainant to notify the change of status, as shown in Figure 4.37. Finally, the change of status will be reflected in the status displayed on-screen, as well as the history section.

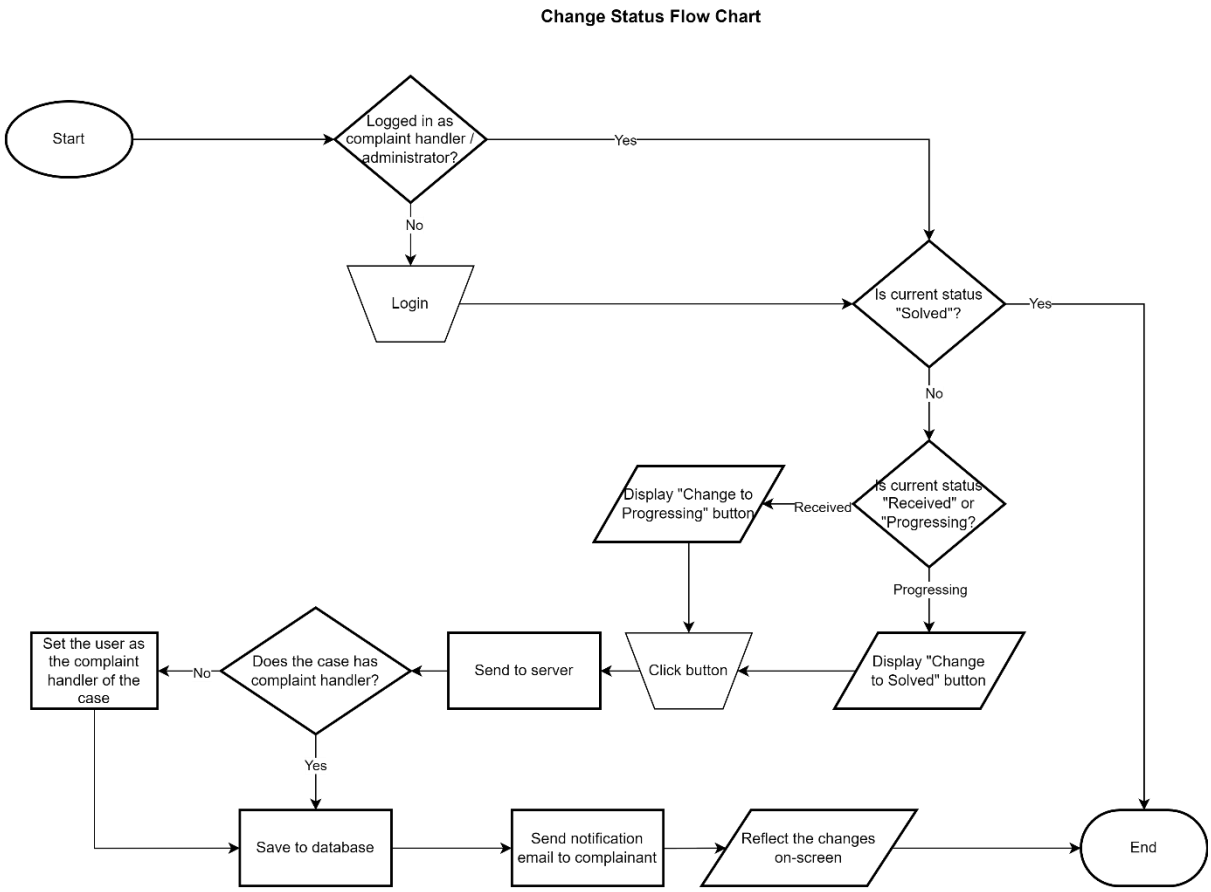


Figure 4.34 Change status flow chart

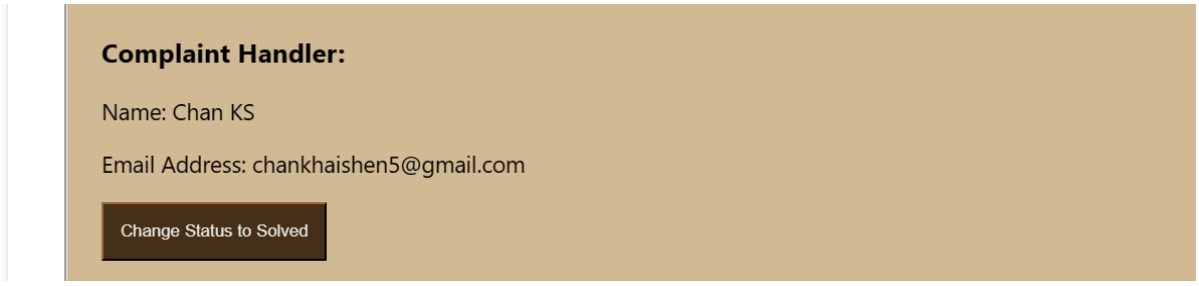


Figure 4.35 Change status button

The order of change of status

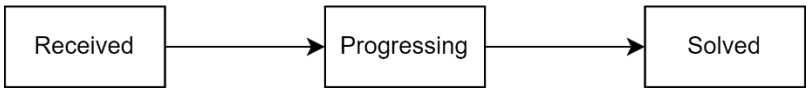


Figure 4.36 Order of change of case status

Figure 4.37 Notification email (Change status)

4.5.7 Assign complaint handler to a case (Complaint handler / Administrator)

There are two ways of complaint handler assignment to a case. The first one is complaint handler self-assign and the second one is administrator assign complaint handler.

4.5.7(a) Complaint handler self-assign

In the list of complaint cases available to a complaint handler, all cases which complaint handler is not available, is available to be chosen by any complaint handler. The self-assignment happens automatically when the complaint handler adds a message to the complaint case or change the status of the case. When the complaint handler is changed, it will reflect on the histories section. This change will only be displayed to complaint handler and administrator. This is also explained in both subtopic 4.5.5 and 4.5.6.

4.5.7(b) Administrator assign complaint handler to a case

The assign complaint handler feature is available only to the complaint handler. This is related to the delete complaint handler operation when the administrator needs to re-assign complaint handler (see subtopic 4.2.2(b) for more explanation). As shown in Figure 4.38, the administrator can choose a complaint handler from the dropdown list and click the assign button. The complaint handler's email address and the case ID will be sent to the server. Then, the change of complaint handler will be updated to the database. Finally, the change will be reflected in the histories section.

Figure 4.38 Choose complaint handler (Administrator view)

4.6 Statistics Reporting

Figure 4.39 is the statistics reporting page, which is only available to the administrator. The administrator can oversee the key statistics of the complaint cases in the system from this page (more explanation on code snippet subtopic).

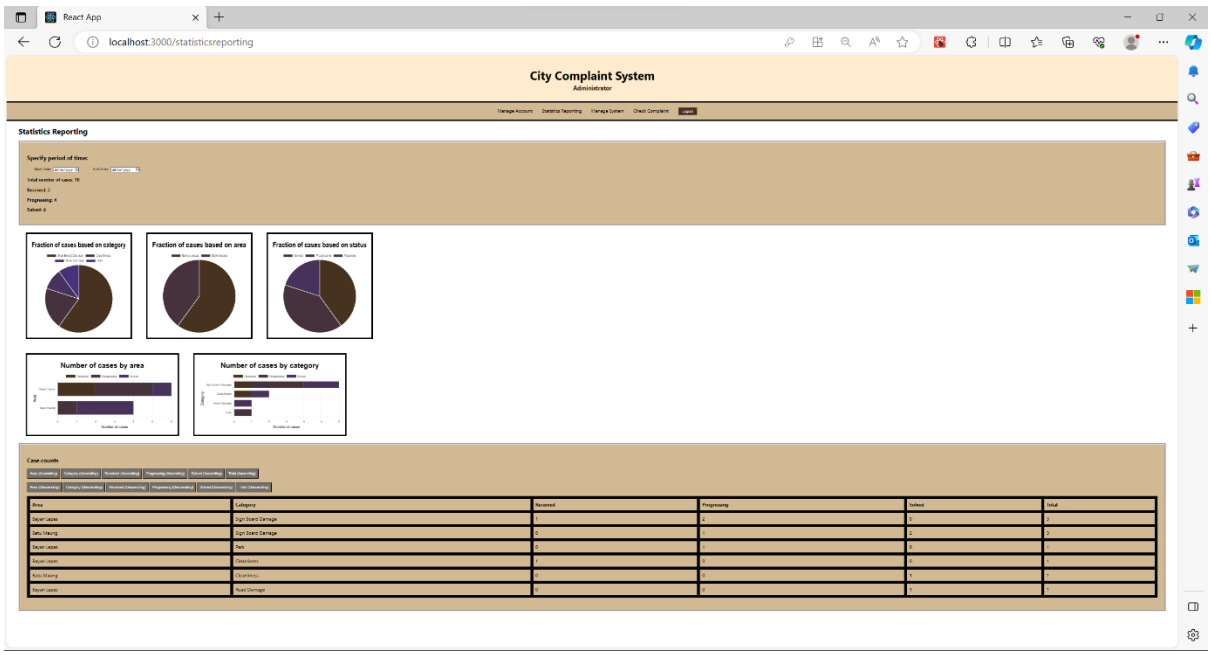


Figure 4.39 Statistics reporting page

4.6.1 Specify date section and summary of complaint count section

The administrator can specify the period of time for the all statistics on the statistics reporting page, as shown in Figure 4.40. When the administrator specifies the start date and end date of the statistics reporting, it will take effect on the entire page. Besides that, the summary of complaint count, in terms of total number of cases and number of cases in each

status (namely received, progressing and solved), is displayed beneath the specify date section, as shown in Figure 4.40.

Figure 4.40 Specify range of date and display summary of complaint count section

4.6.2 Pie chart section

Figure 4.41 is the pie chart section of the statistics reporting page. As shown in Figure 4.41, the complaint cases in the entire system within the specified range of date is first classified into fractions based on their category, are and status. The size of fraction represents the number of cases. The grouped data are generated into three different pie charts, for category, are and status respectively, to give the administrator with an overview of the complaint cases from different perspectives.

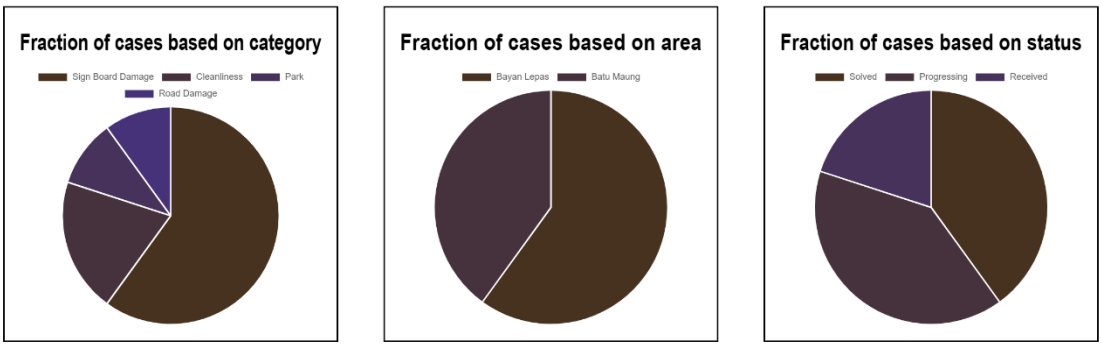


Figure 4.41 Pie chart section

4.6.3 Stacked bar chart section

Figure 4.42 is the stacked bar chart section. Based on the same set of complaint cases as above, this is view from yet another perspective. The complaint cases are grouped by area and category, then in each group (of area or category), the complaint cases are further

grouped by status. The data, in terms of number of cases, are presented to the administrator in two bar charts, each for area and category respectively, and stacked by the current status of case.

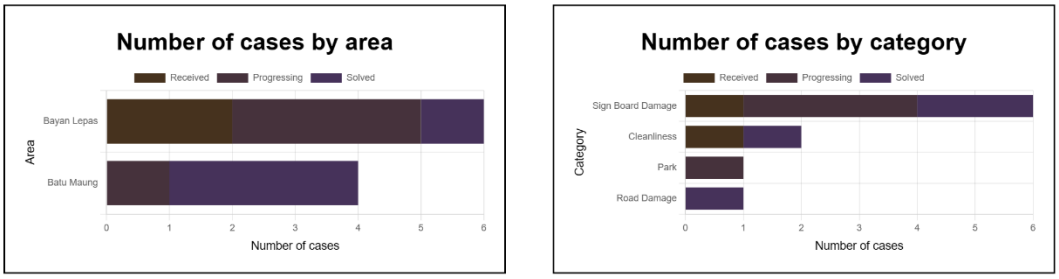


Figure 4.42 Stacked bar chart section

4.6.4 Case count section

Figure 4.43 is the case count section in statistics reporting page. The complaint cases within the specified range of date are grouped based on area, category and status, and the number of cases in each group are presented to the administrator in tabular form, as shown in Figure 4.43. In the table, each row represents a different combination of area and category. In that row, the total number of cases in that area and that category, as well as the number of cases in each status, are displayed. Besides that, as shown in Figure 4.43, the administrator can also sort the case count table based on area, category, number of cases in received, progressing and solved status, or total number of cases, by clicking the relevant sort button on top of the table.

Case counts

Area (Ascending) | Category (Ascending) | Received (Ascending) | Progressing (Ascending) | Solved (Ascending) | Total (Ascending)

Area (Descending) | Category (Descending) | Received (Descending) | Progressing (Descending) | Solved (Descending) | Total (Descending)

Area	Category	Received	Progressing	Solved	Total
Batu Maung	Sign Board Damage	0	1	2	3
Bayan Lepas	Park	0	1	0	1
Bayan Lepas	Cleanliness	1	0	0	1
Bayan Lepas	Sign Board Damage	1	2	0	3
Batu Maung	Cleanliness	0	0	1	1
Bayan Lepas	Road Damage	0	0	1	1

Figure 4.43 Case counts section

4.7 Code Snippet

4.7.1 Email verification through one-time password (OTP)

In the proposed work, email verification through OTP is used to validate that the email address is real during complainant registration. Besides that, it is also being used to verify the identity of complaint handler when setting password, as well as used for reset password.

4.7.1(a) Generating OTP

For email address verification through OTP, the first step is to have a collection for OTP in the database, which can be used to persist the OTP for some time (Sandeep, 2023). Figure 4.44 is the code snippet of the schema for the otp collection. which will store the email address and the OTP temporarily. The record will only stay for 5 minutes.

```
const otpSchema = new mongoose.Schema({
    emailAddress: {
        type: String,
        required: true
    },
    otp: {
        type: String,
        required: true
    },
    created: {
        type: Date,
        default: Date.now,
        expires: 60 * 5     // expires in 5 minutes
    }
});

const otpModel = mongoose.model('otps', otpSchema);
```

Figure 4.44 OTP Schema

When a user requests for an OTP, a random OTP will be generated, then sent to the user's email address, then saved to the database (Sandeep, 2023). OTP generation is done by using the Math library to generate a random integer for six times, as shown in Figure 4.45 (Mdn Web Docs, n.d.). Then, the email containing the OTP sent out to the user's email address using node mailer library. Finally, the OTP is saved to the otp schema.

```
var otp = '';
for (var i=0; i<6; i++) {
    var randomNumber = Math.floor(Math.random() * 10);
    randomNumber = randomNumber.toString();
    otp += randomNumber;
}
console.log('otp', otp);
```

Figure 4.45 Generate OTP

4.7.1(b) Verifying OTP

The final step in email verification is when the user sends a request along with the OTP, the OTP will be verified against the record in the database (Sandeep, 2023). As shown in Figure 4.46, the OTP will be checked with the record in the otp collection, which has the latest date. The latest OTP is first in the returned list (Sandeep, 2023). If the OTPs match, then the email address is verified. If there is no OTP stored that has association with that email address, then the verification is considered failed. Finally, to prevent the OTP from being reused, all OTPs associated with the email address are deleted from database.

```
otpModel.find({emailAddress: emailAddress}).sort({'created': 'descending'}).then(result=>{
    if (result == null){
        console.log('register: No otp');
        res.status(400).json({
            message: "Email verification fail"
        });
        return;
    }

    if (otp !== result[0].otp) {
        console.log('register: wrong otp');
        res.status(400).json({
            message: "Email verification fail"
        });
        return;
    }

    otpModel.deleteMany({emailAddress: emailAddress}).then(response=>{  // Otp will be deleted after verified
        console.log('register: delete otp', response);
```

Figure 4.46 Verify OTP

4.7.2 Statistics reporting

4.7.2(a) Generation of grouped data

Figure 4.47 is the code snippet for grouping complaint case. First, the complaint case is filtered based on the start date and end date specified. This is done by using the match operator as the first step. Then, in the second step, the filtered complaint cases are grouped based on category, area and status, and the count of the case of each group is returned. This

is done by using the group operator and specifying the group field inside the "_id" field (Mongo DB, n.d.). This means that if category field in _id field is set as "$category", the complaints will be grouped based on the category field in complaints collection.

```
complaintModel.aggregate(
    [
        {
            $match: {
                created: {
                    $gte: startDate == null ? new Date('2023-01-01T00:00:00+08:00') : new Date(startDate + 'T00:00:00+08:00'),
                    $lte: endDate == null ? Date.now : new Date(endDate + 'T23:59:59+08:00')
                }
            }
        },
        {
            $group: {
                _id: {
                    category: byCategory != null && byCategory === '1' ? '$category' : null,
                    area: byArea != null && byArea === '1' ? '$area' : null,
                    status: byStatus != null && byStatus === '1' ? '$status' : null
                },
                count: {
                    $sum: 1
                }
            }
        }
    ]
```

Figure 4.47 Generate grouped data

4.7.2(b) Front end handling

Figure 4.48 is the code snippet for handling data at the front end for to prepare data for pie chart and stacked bar chart. The case count based on area is directly used as the input for the area pie chart. The case count is sorted in descending order and only the top six largest category is displayed on pie chart. This is the same for category and status. As for the stacked bar chart, another request is made, which will return case count based on area and status. Then, for each of the top categories, the case count based on status is put into the variables. Then, this is used as the input data for stacked bar chart.

```
const counts = result.data.counts;
const areaCategoryStatusCounts = [];

for (var i = 0; i < counts.length; i++) {
    const area = counts[i]._id.area;
    const category = counts[i]._id.category;

    if (areaCategoryStatusCounts.find(count => count.area === area && count.category === category) == null) {
        areaCategoryStatusCounts.push({
            id: crypto.randomUUID(),
            area: area,
            category: category,
        });
    }
}

for (var j = 0; j < areaCategoryStatusCounts.length; j++) {

    for (var k = 0; k < counts.length; k++) {

        if (counts[k]._id.area === areaCategoryStatusCounts[j].area && counts[k]._id.category === areaCategoryStatusCounts[j].category) {

            if (counts[k]._id.status === 'Received') {
                areaCategoryStatusCounts[j].received = counts[k].count;
            }
            else if (counts[k]._id.status === 'Progressing') {
                areaCategoryStatusCounts[j].progressing = counts[k].count;
            }
            else if (counts[k]._id.status === 'Solved') {
                areaCategoryStatusCounts[j].solved = counts[k].count;
            }

        }

    }

    const receivedCount = areaCategoryStatusCounts[j].received == null ? 0 : areaCategoryStatusCounts[j].received;
    const progressingCount = areaCategoryStatusCounts[j].progressing == null ? 0 : areaCategoryStatusCounts[j].progressing;
    const solvedCount = areaCategoryStatusCounts[j].solved == null ? 0 : areaCategoryStatusCounts[j].solved;

    const totalCount = receivedCount + progressingCount + solvedCount;
    areaCategoryStatusCounts[j].total = totalCount;
}

console.log(areaCategoryStatusCounts);
setAreaCategoryStatusCounts(areaCategoryStatusCounts);
```

Figure 4.48 Front end handing of grouped data

4.7.2(c) Rendering chart

Figure 4.49 is the code snippet for rendering pie chart. This is done by using the react-chartjs-2 library. This done by feeding the data field with a list of background colours and a list of count data of each class, whereas the label field is fed with a list of names of each class (Maduabuchi, 2022). In the area pie chart, the array of predefined background colours and the array of area count generated previously are fed into data field, whereas the label field is fed with the array of area names.

```
<Pie
    data={
        {
            labels: areaCounts.map(areaCount => areaCount.area),
            datasets: [
                {
                    backgroundColor: backgroundColors.slice(0, areaCounts.length),
                    data: areaCounts.map(areaCount => areaCount.count)
                }
            ]
        }
    }
    options={
        {
            responsive: true,
            plugins: {
                title: {
                    text: `Fraction of cases based on area${startDate===''?'':' from '+startDate}${endDate===''?'':' until '+endDate}`,
                    display: true,
                    padding: {
                        top: 10,
                        bottom: 10
                    },
                    font: {
                        size: 30,
                        weight: 'bold'
                    },
                    color: 'black'
                },
                legend: {
                    display: true
                }
            }
        }
    }
></Pie>
```

Figure 4.49 Rendering chart

5. Testing

5.1 Unit Testing

Unit testing is carried out on the system. The test case is summarised below.

Table 5.1 Test case

Page	Test case	Expected result	Screenshot	Code snippet	Result
Registration	User enter a not valid password “123456”	Prompt error message beneath password field	Appendix A1	Appendix A2	Pass
Registration	User enter invalid email “myemail”	Prompt error message	Appendix A3	Appendix A4	Pass
Lodge complaint	User enter something to title then backspace until left blank	Prompt error message	Appendix A4	Appendix A5	Pass
Check complaint	User enter start date which is later than end date	Prompt error message	Appendix A5	Appendix A6	Pass

5.2 Integration testing

In Figure 5.1, administrator successful view the data in statistics reporting, which means front end and server end is integrated.

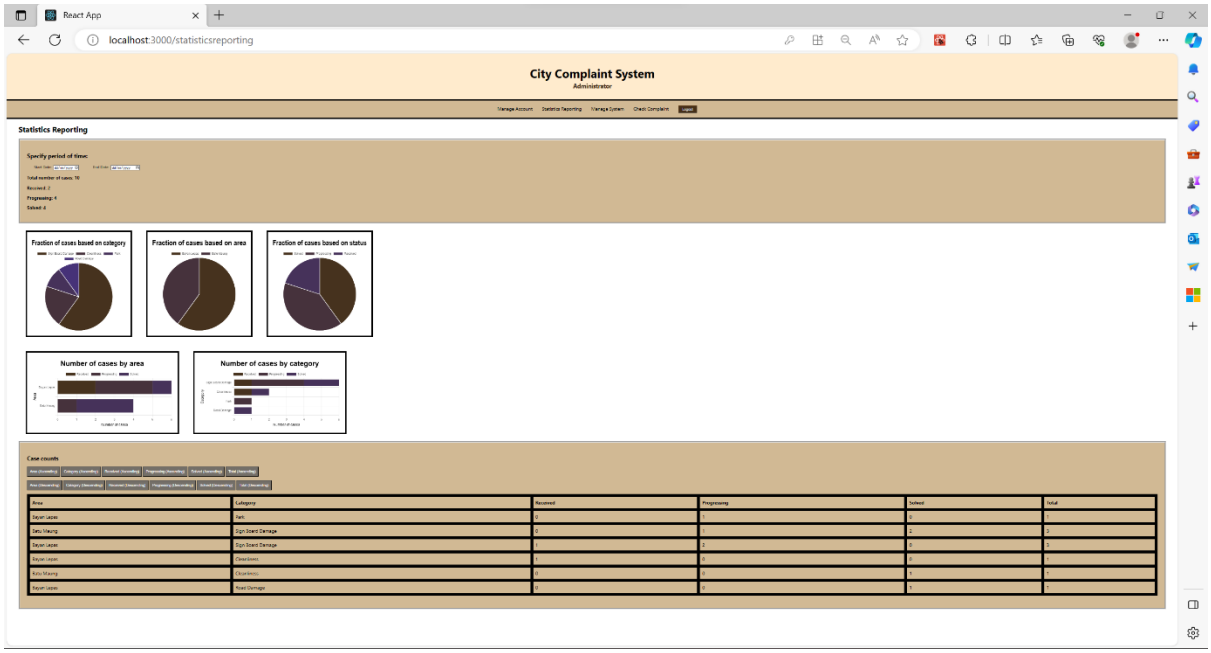


Figure 5.1 Integration testing

5.3 Security testing

In Figure 5.2, when user enter {1==1} in title search bar, it does not happen situation where all complaint case is displayed, so it is secured from No-SQL injection.

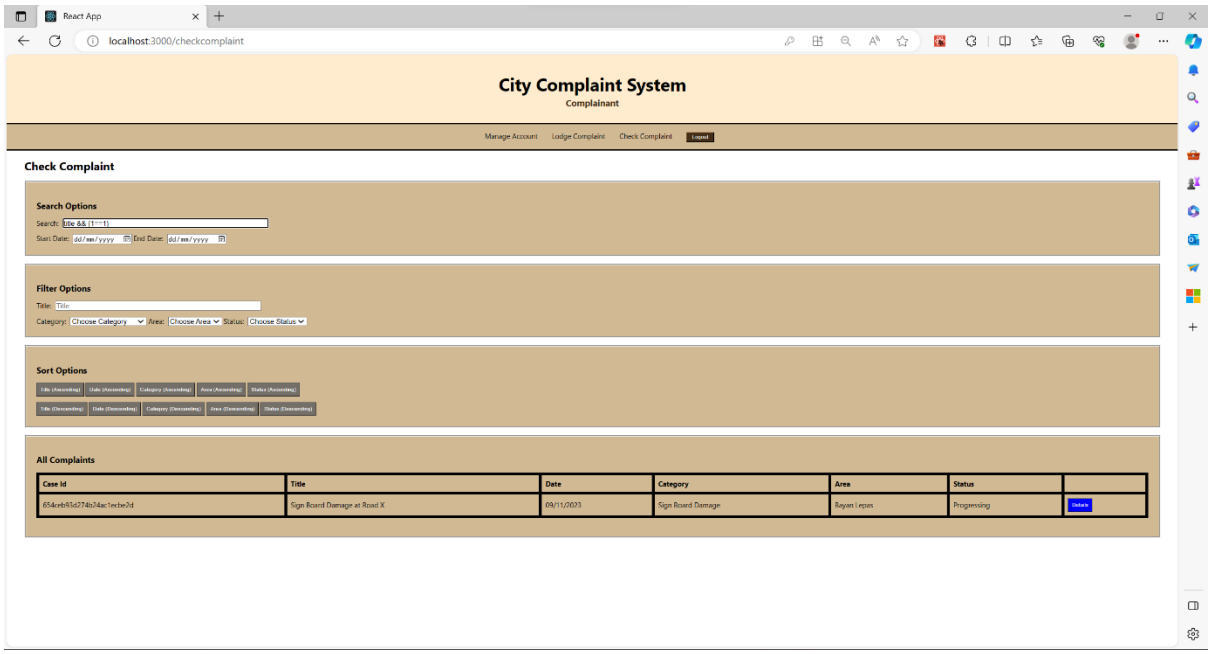


Figure 5.2 Security testing

REFERENCES

Afify, E. A. & Kadry, M. A., 2019. Electronic-Customer Complaint Management (E-CCMS) - A generic approach. *International Journal of Advanced Networking and Applications ,* 11(1), pp. 4125-4141.

Bhadouria, L. S., Kumar, N., Faisal, A. & Devi, S., 2021. Online complaint management system. *Turkish Journal of Computer and Mathematics Education,* 12(6), pp. 5144-5150.

Crack station, n.d. *Hashing security.* [Online]
Available at: https://crackstation.net/hashing-security.htm
[Accessed 2 November 2023].

DBKL, n.d. *Feedback enquiry.* [Online]
Available at: http://adukl.dbkl.gov.my/eApps/sdmscasepool/SdmsCasePool/check.do
[Accessed 24 June 2023].

DBKL, n.d. *Submit feedback.* [Online]
Available at: http://adukl.dbkl.gov.my/eApps/sdmscasepool/SdmsCasePool/add.do
[Accessed 24 June 2023].

Express Js, n.d. *Body Parser.* [Online]
Available at: https://expressjs.com/en/resources/middleware/body-parser.html
[Accessed 11 October 2023].

Geeks for geeks, 2021. *Node.js crypto.pbkdf2() method.* [Online]
Available at: https://www.geeksforgeeks.org/node-js-crypto-pbkdf2-method/
[Accessed 3 November 2023].

Maduabuchi, C., 2022. *Using Chart.js in React.* [Online]
Available at: https://blog.logrocket.com/using-chart-js-react/
[Accessed 20 October 2023].

MBPP, n.d. *Membership of E-complaints.* [Online]
Available at: https://aduan.mbpp.gov.my/members_eng.cfm
[Accessed 23 June 2023].

MBPP, n.d. *New Complaint.* [Online]
Available at: https://aduan.mbpp.gov.my/complaint_eng.html
[Accessed 23 June 2023].

MBPP, n.d. *SMS procedure.* [Online]
Available at: https://aduan.mbpp.gov.my/sms_eng.html
[Accessed 23 June 2023].

MBPP, n.d. *View complaint.* [Online]
Available at: https://aduan.mbpp.gov.my/view_eng.html]
[Accessed 23 June 2023].

Mdn Web Docs, n.d. *Math.random().* [Online]
Available at: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/random
[Accessed 11 October 2023].

Mongo DB, n.d. *group (aggregation).* [Online]
Available at: https://www.mongodb.com/docs/manual/reference/operator/aggregation/group/
[Accessed 21 October 2023].

Mongoose Js, n.d. *Guides.* [Online]
Available at: https://mongoosejs.com/docs/guides.html
[Accessed 6 October 2023].

Morgan, J., 2022. *How To Add Login Authentication to React Applications.* [Online]
Available at: https://www.digitalocean.com/community/tutorials/how-to-add-login-authentication-to-react-applications#step-1-building-a-login-page
[Accessed 6 October 2023].

React Chart Js 2, n.d. *Examples.* [Online]
Available at: https://react-chartjs-2.js.org/examples
[Accessed 20 October 2023].

React, n.d. *Conditional Rendering.* [Online]
Available at: https://react.dev/learn/conditional-rendering
[Accessed 20 October 2023].

React, n.d. *Rendering Lists.* [Online]
Available at: https://react.dev/learn/rendering-lists
[Accessed 19 October 2023].

Sandeep, S., 2023. *How to Implement OTP Verification in Authentication System with Express.js and MongoDB.* [Online]
Available at: https://sandydev.medium.com/how-to-implement-otp-verification-in-authentication-system-with-express-js-and-mongodb-c4f1c1314aed
[Accessed 11 October 2023].

Statista, 2023. *Number of population in Malaysia as of July 2023, by state.* [Online]
Available at: https://www.statista.com/statistics/1040670/malaysia-population-distribution-by-state/
[Accessed 30 August 2023].

Suryotrisongko, H., Jayanto, D. P. & Tjahyanto, A., 2017. Design and Development of Backend Application for Public Complaint Systems Using Microservice Spring Boot. *Procedia Computer Science,* Volume 124, pp. 736-743.

Tutorials Point, n.d. *SDLC - Waterfall model.* [Online]
Available at: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm
[Accessed 15 September 2023].

W3 Schools, n.d. *How to - Filter / search table.* [Online]
Available at: https://www.w3schools.com/howto/howto_js_filter_table.asp
[Accessed 2 November 2023].

W3 Schools, n.d. *How to -Sort a table.* [Online]
Available at: https://www.w3schools.com/howto/howto_js_sort_table.asp
[Accessed 3 November 2023].

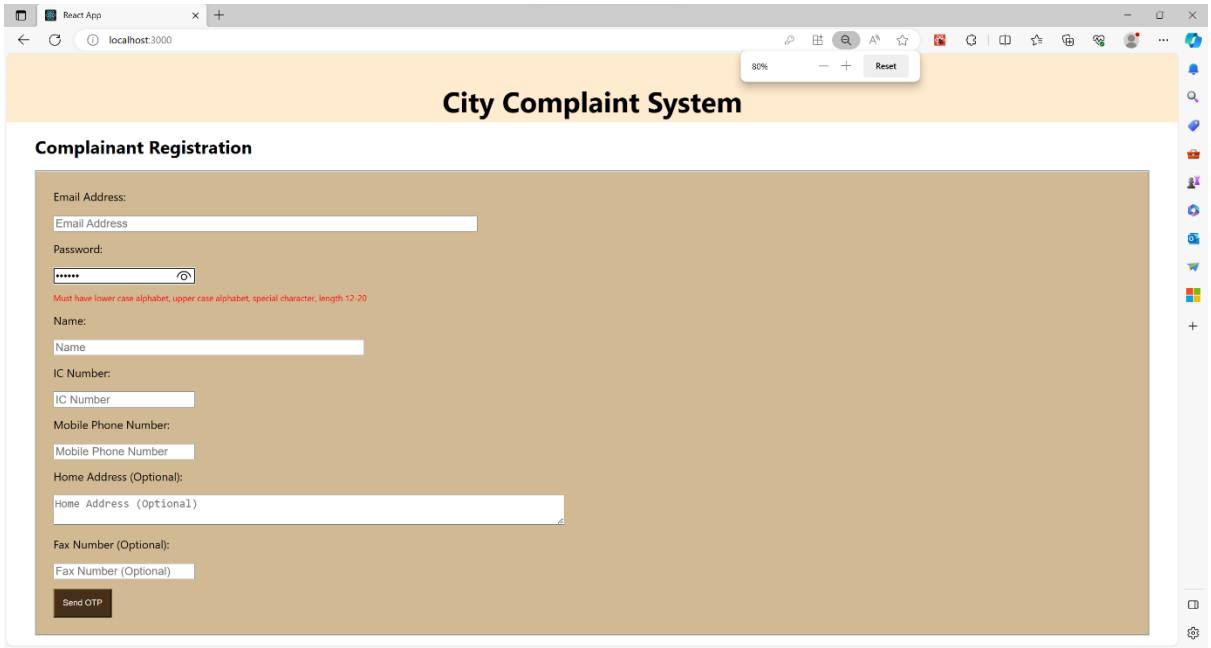
W3 Schools, n.d. *Node.js send an email.* [Online]
Available at: https://www.w3schools.com/nodejs/nodejs_email.asp
[Accessed 10 October 2023].

W3 Schools, n.d. *React Forms.* [Online]
Available at: https://www.w3schools.com/react/react_forms.asp
[Accessed 10 October 2023].

W3 Schools, n.d. *React Router.* [Online]
Available at: https://www.w3schools.com/react/react_router.asp
[Accessed 5 October 2023].

6. Appendix A

A1



A2

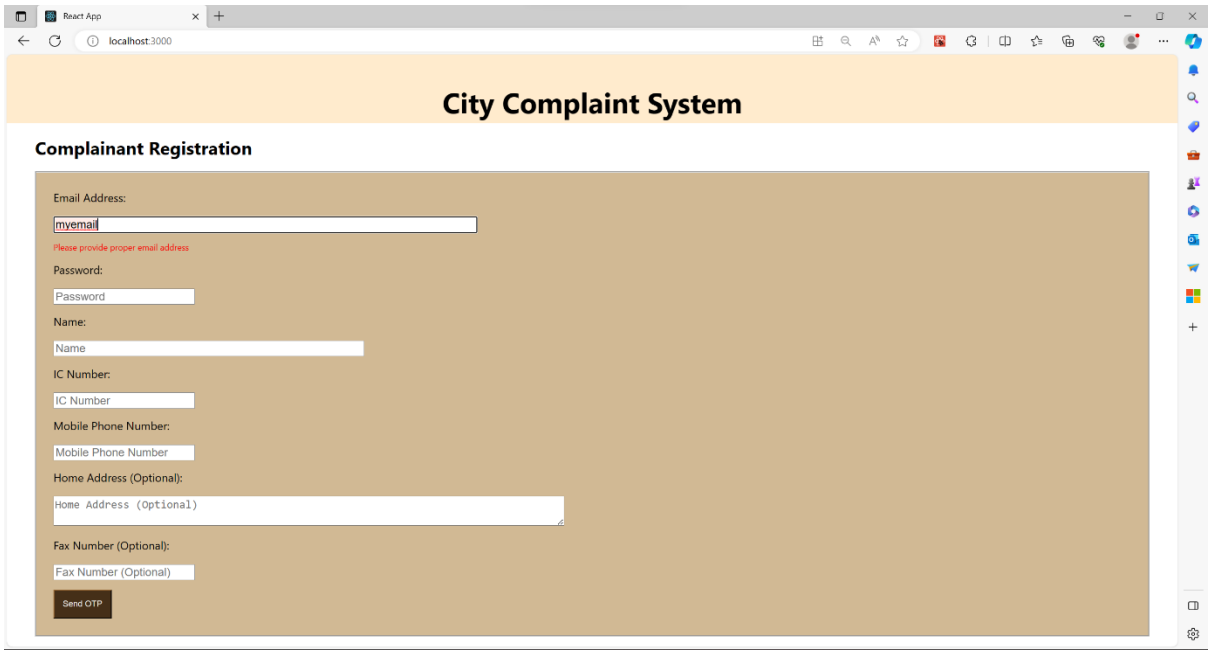
```
case "password":
    const password = event.target.value;
    setPassword(password);

    if (password.trim().length === 0)
        setPasswordError("Cannot be empty");
    else if (password.trim().length > 20)
        setPasswordError("Cannot exceed 20 characters");
    else if (/[\s]+/.test(password))
        setPasswordError("Cannot have white space");
    else {
        var errorMessage = "";

        if (!/[0-9]+/.test(password))
            errorMessage = "Must have digit";
        if (!/[a-z]+/.test(password))
            errorMessage += errorMessage==""?"Must have lower case alphabet":", lower case alphabet";
        if (!/[A-Z]+/.test(password))
            errorMessage += errorMessage==""?"Must have upper case alphabet":", upper case alphabet";
        if (!/[!|@|#|$|%|^|&|*|(|)|+|-|?|.|,]+/.test(password))
            errorMessage += errorMessage==""?"Must have special character":", special character";
        if (!/.{12,20}/.test(password))
            errorMessage += errorMessage==""?"Must have length 12-20":", length 12-20";

        setPasswordError(errorMessage);
    }
    break;
```

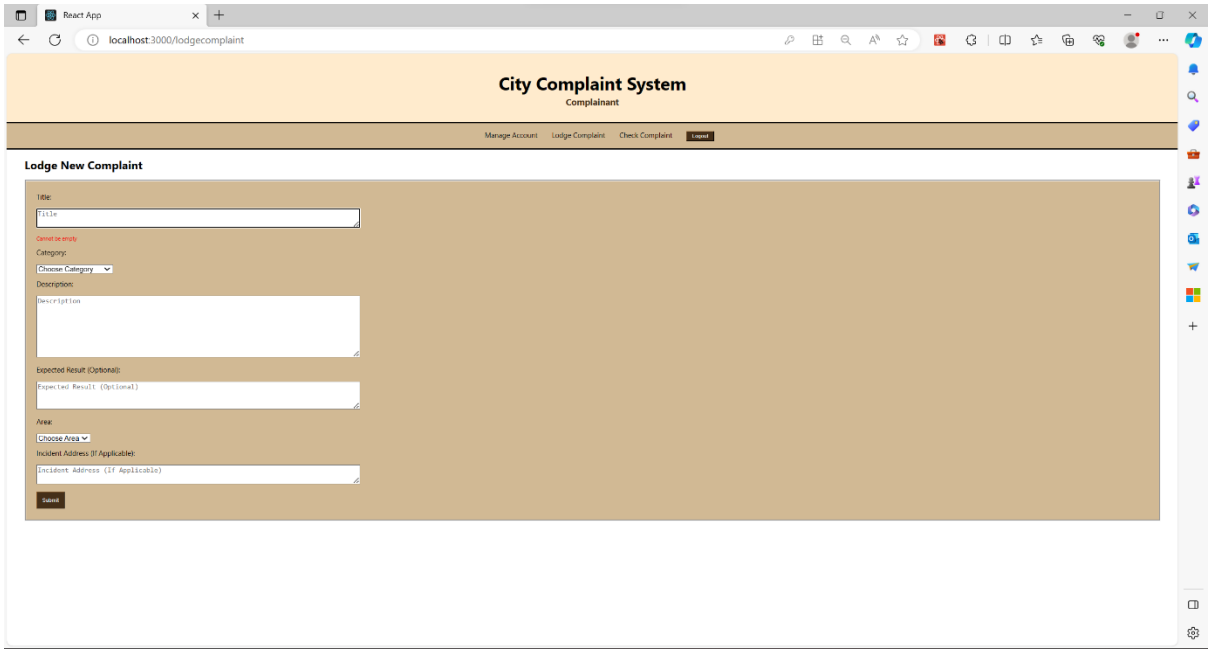
A3



A4

```
case "emailAddress":
    const emailAddress = event.target.value;
    setEmailAddress(emailAddress);
    if (emailAddress.trim().length === 0)
        setEmailAddressError("Cannot be empty");
    else if (emailAddress.length > 70)
        setEmailAddressError("Cannot exceed 70 characters");
    else if(!(/^[^\s@]+@[^\s@]+\.[^\s@]+$/.test(emailAddress)))
        setEmailAddressError("Please provide proper email address");
    else
        setEmailAddressError("");
    break;
```

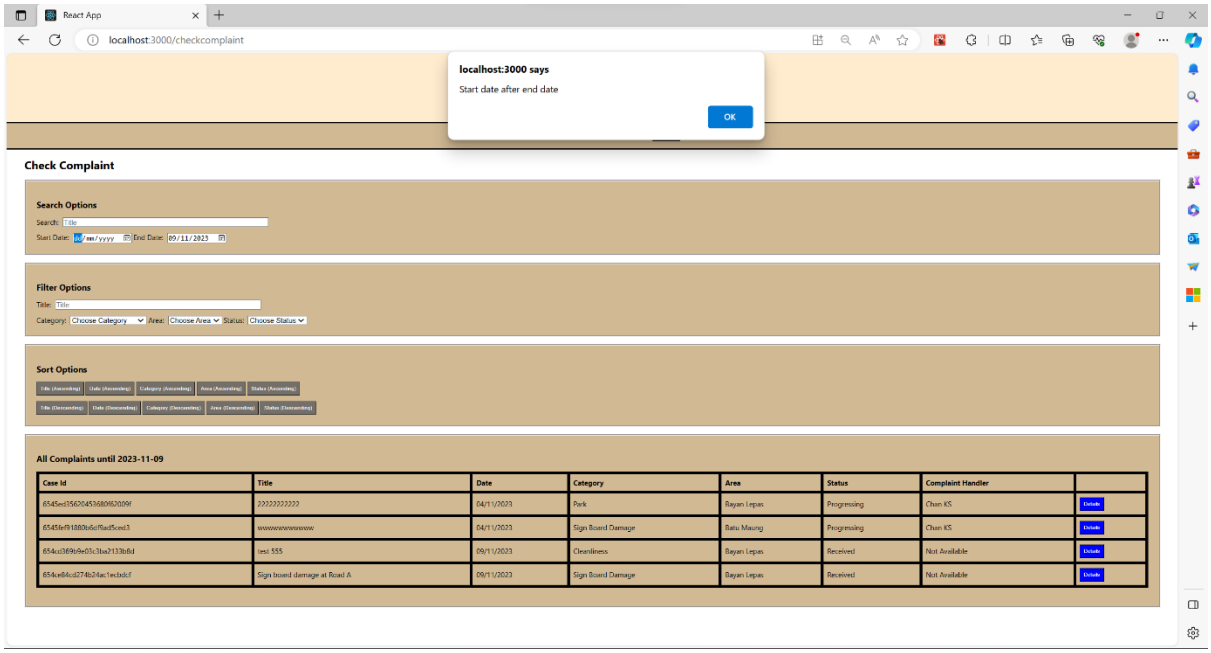
A5



A6

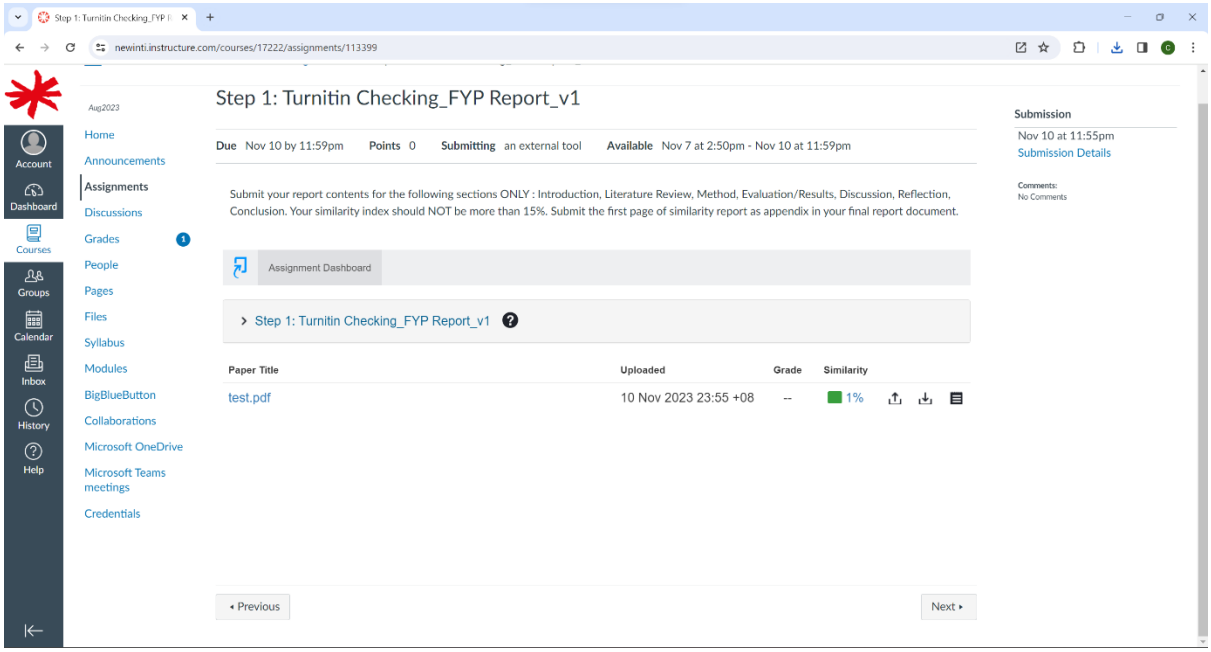
```
case "title":
    var title = event.target.value;
    if (title.trim().length === 0)
        setTitleError("Cannot be empty");
    else if (title.length > 100)
        setTitleError("Cannot exceed 100 characters");
    else
        setTitleError("");
    setTitle(title);
    break;
```

A7



A8

```
case 'startDate':
    if (new Date(event.target.value) > new Date(endDate) ) {
        alert ('Start date after end date');
    }
    else {
        setStartDate(event.target.value);
    }
    break;
```


Step 1: Turnitin Checking_FYP
newinti.instructure.com/courses/17222/assignments/113399
Account
Dashboard
Courses
Groups
Calendar
Inbox
History
Help
Aug2023
Home
Announcements
Assignments
Discussions
Grades
People
Pages
Files
Syllabus
Modules
BigBlueButton
Collaborations
Microsoft OneDrive
Microsoft Teams meetings
Credentials
Step 1: Turnitin Checking_FYP Report_v1
Due Nov 10 by 11:59pm
Points 0
Submitting an external tool
Available Nov 7 at 2:50pm - Nov 10 at 11:59pm
Submit your report contents for the following sections ONLY : Introduction, Literature Review, Method, Evaluation/Results, Discussion, Reflection, Conclusion. Your similarity index should NOT be more than 15%. Submit the first page of similarity report as appendix in your final report document.
Assignment Dashboard
Step 1: Turnitin Checking_FYP Report_v1
Paper Title
Uploaded
Grade
Similarity
test.pdf
10 Nov 2023 23:55 +08
--
1%
Previous
Next
Submission
Nov 10 at 11:55pm
Submission Details
Comments:
No Comments