

# **ONLINE COMPLAINT PORTAL IN MUNICIPALITY USING JAVA**

Submitted in partial fulfillment of requirement for the award of the Degree

Bachelor of Computer Science

In the faculty of Computer Science of Bharathiar University, Coimbatore

Submitted by

**B.KARTHIKA**

**(Reg.No.2022K0133)**

Under the guidance of

**Dr.A.FINNY BELWIN, MCA., M.Sc., Ph.D.**

Guest lecturer, Department of Computer Science



**Department of Computer Science**

**L.R.G GOVERNMENT ARTS COLLEGE FOR WOMEN**

**(Affiliated To Bharathiar University)**

**TIRUPUR-4**

**APRIL-2023**

*CERTIFICATE*

# **CERTIFICATE**

This is to certify that the project work entitled“**ONLINE COMPLAINT PORTAL IN MUNICIPALITY USING JAVA**” Submitted to Bharathiar University in partial fulfilled of the requirement for the award of the Degree of Bachelor of computer science is a record of the original work done by**Ms.B.KARTHIKA(Reg.No.2022K0133)** Under my supervisor and that project work has not formed the basis for the any Degree/Diploma/Association/Fellowship or similar title to any candidate of any university.

**Internal Guide**

**Dr.A.FINNY BELWIN,M.CA., M.Sc., Ph.D**

**Head of the Department**

**.Dr.R.PARIMALA,M.Sc.,M.Phil.,Ph.D**

Viva-voce examination is held on \_\_\_\_\_ L.R.G Government Arts College for Women, Tirupur-641604.

**Internal Examiner**

**External Examiner**

# *DECLARATION*

## **DECLARATION**

I hereby declare that the project work submitted to the **Department of Computer Science, L.R.G. Government Arts College for Women, Tirupur**, affiliated to Bharathiar University, Coimbatore in the partial fulfillment of the required for the award of Bachelor of Computer Science is an original work done by me during the Sixth semester.

**Place:**

**Date:**

**Signature of the Candidate**

**(B.KARTHIKA)**

**(Reg.No:2022K0133)**

# *ACKNOWLEDGEMENT*

## ACKNOWLEDGEMENT

As first and foremost I would like to external my thankfulness to the almighty for blessing the work of my hands, I am grateful to my parents for continued encouragement that they had given to me.

I would like to external my profound gratitude and sincere thanks to **Dr.M.R.YEZHILI,MA.,M.Phil.,Ph.D**Principal, L.R.G Government Arts College For Women , for the encouragement rendered to me during this project.

It's my privilege to thank **Dr.R.PARIMALA,MSc.,M.Phil.,Ph.D** Assistant Professor Department of Biochemistry L.R.G Government Arts College For Women , Incharge Headof the department of computer science for her valuable guidance and support throughout the project development work.

I express my deep sense of gratitude and sincere thanks to my guide**Dr.A.FINNYBELWIN M.CA., M.Sc., Ph.D**.Guest Lecturer, Department of computer science, for providing all sorts of facilities to complete my project work successfully.

I also extend my sincere thanks to all the other faculty member of the department and technical assistance for their co-operation and valuable guidance.

I thank our college library for providing me with many informative books that help me to enrich my knowledge to bring out the project successfully.

# *CONTENTS*



# CONTENTS

<b>TITLE</b>	<b>PAGE NO</b>
<b>SYNOPSIS</b>	
<b>1. INTRODUCTION</b>	
1.1 PROJECT DESCRIPTION	1
1.2 DESCRIPTION OF MODULES	2
<b>2. SYSTEM SPECIFICATION</b>	3
2.1 HARDWARE REQUIREMENTS	5
2.2 SOFTWARE SPECIFICATION	5
<b>3. SYSTEM ANALYSIS</b>	
3.1 EXISTING SYSTEM	6
3.1.1 DRAWBACKS	
3.2 PROPOSED SYSTEM	6
3.2.1 FEATURES	
<b>4. SYSTEM DESIGN</b>	
4.1 FILE DESIGN	7
4.2 INPUT DESIGN	8
4.3 OUTPUT DESIGN	9
4.4 DATABASE DESIGN	10
<b>5. TESTING AND IMPLEMENTATION</b>	
5.1 SYSTEM TESTING	11
5.2 TYPES OF TESTING	12
5.3 SYSTEM IMPLEMENTATION	13
<b>6. CONCLUSION</b>	19
<b>7. BIBLIOGRAPHY</b>	20
<b>8. APPENDICES</b>	
A. DATA FLOW DIAGRAM	21
B. TABLE STRUCTURE	23
C. SAMPLE CODING	25
D. SAMPLE INPUT	43
E. SAMPLE OUTPUT	46

## *SYNOPSIS*

# SYNOPSIS

The common people under the jurisdiction of a municipal corporation to register their grievances about day-to-day problems in their ward through a web application.

It will provide a common man to deliver his complaints and problems to municipal authority as well as let the municipal authorities to address the problem in a short period of time.

An interface to register one's complained and follows it up. It provides a complaint module which helps clicking up a picture of any problem that people are facing and upload its image along with the complaint.

In India we don't have any direct communication between the government and public in an efficient way for solving the problems.

I.e., for getting a problem solved in our place we have to bribe the officials and get them solved in 2 months which can be solved actually in 1 month of time. In order to make the goal of NIC come true we are going to develop a system which will be able to provide the complete information to the public at any point of time regarding the problems.

They are facing currently and what is the impact of it and then how effectively the funds are utilized for the development purpose can be known by public which also includes the online discussion forums and feedback forms which will help them to communicate well with the government.

# *INTRODUCTION*

# **1. INTRODUCTION**

## **OBJECTIVE**

The Online Complaint Management System provides a website solution for community issues by saving time.

### **1.1 PROJECT DESCRIPTION**

The common people under the jurisdiction of a municipal corporation to register their grievances about day-to-day problems in their ward through a web application.

It will provide a common man to deliver his complaints and problems to municipal authority as well as let the municipal authorities to address the problem in a short period of time.

An interface to register one's complaint and follow it up. It provides a complaint module which helps clicking up a picture of any problem that people are facing and upload its image along with the complaint.

In India we don't have any direct communication between the government and public in an efficient way for solving the problems.

I.e., for getting a problem solved in our place we have to bribe the officials and get them solved in 2 months which can be solved actually in 1 month of time. In order to make the goal of NIC come true we are going to develop a system which will be able to provide the complete information to the public at any point of time regarding the problems.

They are facing currently and what is the impact of it and then how effectively the funds are utilized for the development purpose can be known by public which also includes the online discussion forums and feedback forms which will help them to communicate well with the government.

## **1.2 DESCRIPTION OF MODULES**

- ❖ New Citizens Registration
- ❖ Apply Complaint Request
- ❖ View Complaints
- ❖ View All Citizens Details
- ❖ Apply Request Actions

### **New Citizens Registration**

These modules the citizens are manually visit the page and give the details and register manually. We can collect the citizen some unique and important information from this application. Once the request has been submitted the admin can handle the approve.

### **Apply Complaint Request**

After the login for citizen, raise the complaint we just collect the issue as complaint and raised the issue. This notification will send to the admin portal. Admin can also check the status of the current process of the issue.

### **View Complaints**

Once the admin portal has been viewing the requested information is completed, then admin changed the status of the process. We can check the admin status as a point of important time.

### **View All Citizens Details**

Admin can check the citizen details in the portal, only admin can see the details of the other citizen, only citizen can check the profile itself.

### **Apply Request Actions**

Once the admin gets take an action for the issue, has to change the status as in progress, completed. Which mean user can easily identify the status of the issues .

# *SYSTEM SPECIFICATION*

## 2. SYSTEM SPECIFICATION

System Requirements Specification also known as Software Requirements Specification, is a document or set of documentation that describes the features and behavior of a software application

### **WINDOWS OS**

Windows is a graphical operating system developed by Microsoft. It allows users to view and store files, run the software, play games, watch videos, and provides a way to connect to the internet. It was released for both home computing and professional works.

Microsoft introduced the first version as 1.0.

It was released for both home computing and professional functions of Windows on 10 November 1983. Later, it was released on many versions of Windows as well as the current version, Windows 10.

In 1993, the first business-oriented version of Windows was released, which is known as Windows NT 3.1. Then it introduced the next versions, Windows 3.5, 4/0, and Windows 2000. When the XP Windows was released by Microsoft in 2001, the company designed its various versions for a personal and business environment.

It was designed based on standard x86 hardware like Intel and AMD processor. Accordingly, it can run on different brands of hardware, such as HP, Dell, and Sony computers, including home-built PCs.

Editions of Windows:

Microsoft has produced several editions of Windows, starting with Windows XP. These versions have the same core operating system, but some versions included advance features with an additional cost. There are two most common editions of Windows:

- Windows Home
- Windows Professional

Windows Home is basic edition of Windows. It offers all the fundamental functions of Windows, such as browsing the web, connecting to the Internet, playing video games, using office software, watching videos. Furthermore, it is less expensive and comes pre-installed with many new computers.



## **JAVA**

Java is a high-level programming language developed by Sun Microsystems. It was originally designed for developing programs for set-top boxes and handheld devices, but later became a popular choice for creating web applications.

The Java syntax is similar to C++, but is strictly an object-oriented programming language. For example, most Java programs contain classes, which are used to define objects, and methods, which are assigned to individual classes. Java is also known for being stricter than C++, meaning variables and functions must be explicitly defined. This means Java source code may produce errors or "exceptions" more easily than other languages, but it also limits other types of errors that may be caused by undefined variables or unassigned types.

Unlike Windows executable (.EXE files) or Macintosh applications (.APP files), Java programs are not run directly by the operating system. Instead, Java programs are interpreted by the Java Virtual Machine, or JVM, which runs on multiple platforms. This means all Java programs are multiplatform and can run on different platforms, including Macintosh, Windows, and Unix computers. However, the JVM must be installed for Java applications or applets to run at all. Fortunately, the JVM is included as part of the Java Runtime Environment (JRE).

## **SQL**

Structured query language (SQL) is a programming language for storing and processing information in a relational database. A relational database stores information in tabular form, with rows and columns representing different data attributes and the various relationships between the data values. You can use SQL statements to store, update, remove, search, and retrieve information from the database. You can also use SQL to maintain and optimize database performance.

Relational database management systems use structured query language (SQL) to store and manage data. The system stores multiple database tables that relate to each other. MS SQL Server, MY SQL, or MS Access are examples of relational database management systems. The following are the components of such a system.

A SQL table is the basic element of a relational database. The SQL database table consists of rows and columns. Database engineers create relationships between multiple database tables to optimize data storage space.

SQL statements, or SQL queries, are valid instructions that relational database management systems understand. Software developers build SQL statements by using different SQL language elements. SQL language elements are components such as identifiers, variables, and search conditions that form a correct SQL statement.

## **2.1 HARDWARE SPECIFICATION**

- Processor : Intel core i3
- RAM : 4 GB RAM
- Hard Disk Drive : 500GB
- Keyboard : Standard Keyboard
- Mouse : Optical mouse

## **2.2 SOFTWARE SPECIFICATION**

- Operating System : Windows10
- Front End : JAVA
- Back End : SQL

# *SYSTEM ANALYSIS*

## **3. SYSTEM ANALYSIS**

### **3.1 EXISTING SYSTEM**

The existing system of Municipal Corporation is difficult to handle the peoples issue without any software. There are a lot of complaints and issues are possible to happen in day-to-day life. The municipal corporation team can't handle all the process without any struggle.

#### **3.1.1 DRAWBACKS**

- Sometimes we can't handle the issue
- No records can be track
- Lot of issues can face everyday

### **3.2 PROPOSED SYSTEM**

In this system will overcome the existing issues, in this software can manage the citizen details, complaints everything. Citizens are manually can check the status of the complaint via this application. This may help to change all the irregular process of the government. Most of the peoples can get benefit for this application.

#### **3.2.1 FEATURES**

- Track the status of the requested issue
- We can get the solution in a proper way
- Can avoid the waiting time for in a long in queue, for raising complaint in a government office

# *SYSTEM DESIGN*

## **4. SYSTEM DESIGN**

### **4.1 FILE DESIGN**

The selection of the file system design approach is done according to the needs of the developers what are the needed requirements and specifications for the new design. It allowed us to identify where our proposal fitted in with relation to current and past file system development. Our experience with file system development is limited so the research served to identify the different techniques that can be used. The variety of file systems encountered show what an active area of research file system development is. The file systems may be from one of the two fundamental categories. In one category, the file system is developed in user space and runs as a user process. Another file system may be developed in the kernel space and runs as a privileged process. Another one is the mixed approach in which we can take the advantages of both aforesaid approaches. Each development option has its own pros and cons. In this article, these design approaches are discussed.

A file system is the data structure designed to support the abstraction of the data blocks as an archive and collection of files. This data structure is unique because it is stored on secondary storage (usually the disk), which is a very slow device.

The file system structure is the most basic level of organization in an operating system. Almost all of the ways an operating system interacts with its users, applications, and security model are dependent upon the way it organizes files on storage devices.

File Design Information systems in business are file and database oriented. Data are accumulated into files that are processed or maintained by the system. The systems analyst is responsible for designing files, determining their contents and selecting a method for organizing the data.

The most important purpose of a file system is to manage user data. This includes storing, retrieving and updating data. Some file systems accept data for storage as a stream of bytes which are collected and stored in a manner efficient for the media.

## 4.2 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:’

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

### OBJECTIVES

- Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
- It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
- When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user
- Will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

## **4.3 OUTPUT DESIGN**

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

### **External Outputs**

Manufacturers create and design external outputs for printers. External outputs enable the system to leave the trigger actions on the part of their recipients or confirm actions to their recipients.

Some of the external outputs are designed as turnaround outputs, which are implemented as a form and re-enter the system as an input.

### **Internal outputs**

Internal outputs are present inside the system, and used by end-users and managers. They support the management in decision making and reporting.

### **Output Integrity Controls**

Output integrity controls include routing codes to identify the receiving system, and verification messages to confirm successful receipt of messages that are handled by network protocol.

Printed or screen-format reports should include a date/time for report printing and the data. Multipage reports contain report title or description, and pagination. Pre-printed forms usually include a version number and effective date.



## 4.4 DATABASE DESIGN

Today's businesses depend on their databases to provide information essential for day-to-day operations, especially in case of electronic commerce businesses who has a definite advantage with up-to-date database access. Good design forms the foundation of any database, and experienced hands are required in the automation process to design for optimum and stable performance.

Software Solutions have been constantly working on these platforms and have attained a level of expertise. We apply proven methodologies to design, develop, integrate and implement database systems to attain its optimum level of performance and maximize security to meet the client's business model.

### Business needs addressed:

- Determine the basic objects about which the information is stored
- Determine the relationships between these groups of information and the objects
- Effectively manage data and create intelligent information
- Remote database administration or on site administrative support
- Database creation, management, and maintenance
- Information retrieval efficiency, remove data redundancy and ensure data security

The most important consideration in designing the database is how the information will be used. The main objective of designing a database is Data Integration, Data Integrity and Data Independence.

### Data

Integration database information from sever all files is coordinated, accessed and operated up on as through it is in a single file .Logically, the information is centralized physically the data may be located on different devices, connected through data communication facilities.

### Data Integrity

Data integrity means storing all data in one place only and how each application accesses it. This approach results in more consistent information, one update being sufficient to achieve and record status for all applications. This lead stoles data redundancy that is data items need not be duplicated.

### Data Independence

Data independence is the insulation of application programs from changing aspects of physical data organization. This objective seeks to allow changes in the content and organization of physical data without reprogramming of application .

# *TESTING AND IMPLEMENTATION*

## **5. TESTING AND IMPLEMENTATION**

### **TESTING METHODOLOGIES**

System testing is state of implementation, which is aimed at ensuring that the system works accurately and efficiently as expect before live operation commences. It certifies that the whole set of programs hang together.

System testing requires a test plan that consists of several key activities and step for run program, string, system and user acceptance testing. The implementation of newly designed package is important in adopting a successful new system

Testing is the important stage in software development. the system test in implementation stage in software development process. The system testing implementation should be confirmation that all is correct and an opportunity to show the users that the system works as expected. It accounts the largest percentage of technical effort in the software development process.

Testing phase in the development cycle validates the code against the functional specification testing is vital to achievement of the system goals. The objective of the testing is to discover errors to fulfill this objective a series of test step unit, integration. Validation and system tests were planned and executed the test steps are:

### **5.1 SYSTEM TESTING**

Testing is an integral part of any system development life cycle. Insufficient and untested applications may tend to crash and the result is loss of economic and manpower investment besides user's dissatisfaction and downfall of reputation. Software testing can be looked upon as one among many processes, an organization performs, and that provides the lost opportunity to correct any flaws in the developed system. Software testing includes selecting test data that have more probability of giving errors.

The first step in system testing is to develop a plan that tests all aspects of the system. Completeness, correctness, reliability and maintainability of the software are to be tested for the best quality assurance that the system meets the specification and requirements for its intended use and performance. System testing is the most useful practical process of executing a program with the implicit intention of finding errors that make the program fails.

## 5.2 TYPES OF TESTING

System testing is done in three phases.

- Unit Testing
- Integration Testing
- Validation Testing

### UNIT TESTING

Unit testing focuses verification effort on the smallest unit of software the module. Using the detailed design and the process specification testing is done to registration by the user with in the boundary of the Login module. The login form receives the username and password details and validates the value with the database. If valid, the home page is displayed.

### INTEGRATION TESTING

Integration Testing is the process of this activity can be considered as testing the design and hence module interaction. The primary objective of integration testing is to discover errors in the interfaces between the components. Log information registration form are integrated and tested together. If the user is newly registered, the received details will be stored in the registration table. While logging in, the application will check for valid user name and password in the registration table and if valid the user is prompted for submitting complaints.

Data can be lost across an interface, one module can have adverse effect on another sub function when combined it may not produce the desired major functions. Integration testing is a systematic testing for constructing test to uncover errors associated within an interface.

The objectives taken from unit tested modules and a program structure is built for integrated testing. All the modules are combined and the test is made.

A correction made in this testing is difficult because the vast expenses of the entire program complicated the isolation of causes. In this integration testing step, all the errors are corrected for next testing process.

### VALIDATION TESTING

Validation are independent procedures that are used together for checking that a product, service, or system meets requirementsand specificationsand that it fulfills its in purpose the actual result from the expected result for the complaint process. Select the complaint category of the complaint by user. The input given to various forms fields are validated effectively. Each module is tested independently. It is tested that the complaint module fields receive the

correct input for the necessary details such as complaint category, complaint id, reference name, complaint description, and email for further process.

After the completion of the integrated testing, software is completely assembled as a package; interfacing error has been uncovered and corrected and a final series of software test validation begins.

Validation testing can be defined in many ways but a simple definition is that validation succeeds when the software function in a manner that can be reasonably expected by the customer. After validation test has been conducted, one of two possible conditions exists.

## **OUTPUT TESTING**

The next process of validation testing, is output testing of the proposed system, since no system could be successful if it does not produce the required output in the specified format. Asking the user about the format required, list the output to be generated or displayed by the system under considerations.

Output testing is a different test whose primary purpose is to fully exercise the computer based system although each test has a different purpose all the work should verify that all system elements have been properly integrated and perform allocated functions.

The output format on the screen is found to be corrected as the format was designed in the system design phase according to the user needs for the hard copy also; the output testing has not resulted in any correction in the system.

## **5.3 SYSTEM IMPLEMENTATION**

When the initial design was done for the system, the client was consulted for the acceptance of the design so that further proceedings of the system development can be carried on. After the development of the system a demonstration was given to them about the working of the system. The aim of the system illustration was to identify any malfunction of the system.

After the management of the system was approved the system implemented in the concern, initially the system was run parallel with existing manual system. The system has been tested with live data and has proved to be error free and user friendly.

Implementation is the process of converting a new or revised system design into an operational one when the initial design was done by the system; a demonstration was given to the end user about the working system.

This process is used to verify and identify any logical mess working of the system by

feeding various combinations of test data. After the approval of the system by both end user and management the system was implemented.

System implementation is made up of many activities. The six major activities are as follows.

## **CODING**

Coding is the process of whereby the physical design specifications created by the analysis team turned into working computer code by the programming team. A design code may be a tool which helps ensure that the aspiration for quality and quantity for customers and their requirements, particularly for large scale projects, sought by the water agency Design pattern are documented tried and tested solutions for recurring problems in a given context. So basically you have a problem context and the proposed solution for the same.

## **INSTALLATION**

Installation is the process during which the current system is replaced by the new system. This includes conversion of existing data, software, and documentation and work procedures to those consistent with the new system.

## **DOCUMENTATION**

Documentation is descriptive information that describes the use and operation of the system. The user guide is provided to the end user as the student and administrator. The documentation part contains the details as follows,

User requirement and water agency details administration has been made online. Any customer can request their water requirement details through online and also use of documentation, they can view the purpose of each purpose, The admin could verify the authentication of the users, users requirements and need to take delivery process, thus the documentation is made of full view of project thus it gives the guideline to study the project and how to execute also.

## **USER TRAINING AND SUPPORT**

The software is installed at the deployment environment, the developer will give training to the end user of the regional transport officer and police admin officer in that software. The

goal of an end user training program is to produce a motivated user who has the skills needed to apply what has been learned to perform the job related task. The following are the instruction which is specified the handling and un-handling events in the application,

- The authenticated user of admin and office workers only login in the application with authorized username and password.
- Don't make user waste their time to come straight to the water agency or make a phone call.
- It can easily track through online by the user.
- Very user friendliness software.

## **IMPLEMENTATION PROCEDURES**

Implementation includes all the activities that take place to convert the old system to the new one. Proper implementation is essential to provide a reliable system to meet the organization requirements. Implementation is the stage in the project where the theoretical design is turned into a working system. The most crucial stage is achieving a successful new system & giving the user confidence in that the new system will work efficiently & effectively in the implementation state.

### **PILOT RUNNING:**

Processing the current data by only one user at a time called the pilot running process. When one user is accessing the data at one system, the system is sets to be engaged and connected in network. This process is useful only in system where more than one user is restricted.

### **PARALLEL RUNNING:**

Processing the current data by more than one user at a time simultaneously is said to be parallel running process. This same system can be viewed and accessed by more than one user at the time. Hence the implementation method used in the system is a pilot type of implementation.

Implementation is the stage in the project where the theoretical design is turned into a working system. The most crucial stage is achieving a successful new system & giving the user confidence in that the new system will work efficiently & effectively in the implementation state.

The stage consists of,

- Testing the developed program with sample data.

- Detection's and correction of error.
- Creating whether the system meets user requirements.
- Making necessary changes as desired by the user.
- Training user personnel.

### **USER TRAINING:**

User Training is designed to prepare the user for testing & contenting the system.

- User Manual.
- Help Screens.
- Training Demonstration.

### **USER MANUAL:**

The summary of important functions about the system and software can be provided as a document to the user.

### **HELP SCREENS:**

This features now available in every software package, especially when it is used with a menu. The user selects the "Help" option from the menu. The system accesses the necessary description or information for user reference.

### **TRAINING DEMONSTRATION:**

Another User Training element is a Training Demonstration. Live demonstrations with personal contact are extremely effective for Training Users.

### **SYSTEM MAINTENANCE**

Maintenance is actually the implementation of the review plan. As important as it is, many programmers and analysts are to perform or identify themselves with the maintenance effort. There are psychological, personality and professional reasons for this. Analysts and programmers spend far more time maintaining programs than they do writing them. Maintenance accounts for 50-80 percent of total system development

Maintenance is expensive. One way to reduce the maintenance costs are through maintenance management and software modification audits.

- Maintenance is not as rewarding as exciting as developing systems. It is perceived as requiring neither skill not experience.



- Users are not fully cognizant of the maintenance problem or its high cost.
- Few tools and techniques are available for maintenance.
- A good test plan is lacking.
- Standards, procedures, and guidelines are poorly defined and enforced.
- Programs are often maintained without care for structure and documentation.
- There are minimal standards for maintenance.
- Programmers expect that they will not be in their current commitment by time their programs go into the maintenance cycle.

### **Corrective Maintenance**

It means repairing, processing or performance failure or making changes because of previously uncovered problems or false assumptions. Task performed to identify, isolate, and rectify a fault so that the failed equipment, machine, or system can be restored to an operational condition within the tolerances or limits established for in-service operations.

Corrective maintenance can be subdivided into "immediate corrective maintenance" (in which work starts immediately after a failure) and "deferred corrective maintenance" (in which work is delayed in conformance to a given set of maintenance rules).

### **Perfective Maintenance**

It means changes made to a system to add new features or to improve performance. Preventive maintenance is predetermined work performed to a schedule with the aim of preventing the wear and tear or sudden failure of equipment components. process or control equipment failure can have adverse results in both human and economic terms. In addition to down time and the costs involved to repair and/or replace equipment parts or components, there is the risk of injury to operators, and of acute exposures to chemical and/or physical agents.

Time-based or run-based Periodically inspecting, servicing, cleaning, or replacing parts to prevent sudden failure .On-line monitoring of equipment in order to use important/expensive parts to the limit of their serviceable life. Preventive maintenance involves changes made to a system to reduce the chance of future system failure.

An example of preventive maintenance might be to increase the number of records that a system can process far beyond what is currently needed or to generalize how a system

sends report information to a printer so that so that the system can adapt to changes in printer technology.

### **Preventive Maintenance**

Changes made to a system to avoid possible future problems Perfective maintenance involves making enhancements to improve processing performance, interface usability, or to add desired, but not necessarily required, system features. The objective of perfective maintenance is to improve response time, system efficiency, reliability, or maintainability.

During system operation, changes in user activity or data pattern can cause a decline in efficiency, and perfective maintenance might be needed to restore performance. Usually, the perfective maintenance work is initiated by the IT department, while the corrective and adaptive maintenance work is normally requested by users.

# *CONCLUSION*

## **6. CONCLUSION**

Municipal Corporation Systems are critical for the efficient management and administration of local government services, including public health, sanitation, water supply, and waste management. These systems provide several benefits, including improved service delivery, increased citizen engagement, and enhanced efficiency.

Municipal Corporation Systems ensure improved service delivery by enabling local governments to efficiently manage and monitor service delivery in their respective areas. This includes effective planning and implementation of infrastructure projects, providing public services like water supply, sanitation, and waste management, and ensuring compliance with local regulations and laws.

Moreover, these systems provide increased citizen engagement by offering a platform for citizens to engage with local governments on issues related to service delivery and governance. This includes public hearings, town hall meetings, and online platforms for feedback and suggestions.

Municipal Corporation Systems also enhance efficiency by leveraging technology and data to optimize service delivery, reduce costs, and improve decision-making. This includes using data analytics and GIS technology to identify problem areas and optimize resource allocation, and implementing e-governance solutions to improve transparency and accountability.

In conclusion, Municipal Corporation Systems are critical for the efficient management and administration of local government services. They offer several benefits, including improved service delivery, increased citizen engagement, and enhanced efficiency. By leveraging these benefits, local governments can better serve their citizens, promote sustainable development, and enhance the overall quality of life in their communities.

# *BIBLIOGRAPHY*

## 7. BIBLIOGRAPHY

### BOOK REFERENCES:

- Bloch, Joshua. Effective Java: Programming Language Guide. Addison-Wesley, 2017.
- Eckel, Bruce. Thinking in Java. Prentice Hall, 2006.
- Shildt, Herbert. Java: A Beginner's Guide. McGraw-Hill Education, 2018.
- Sierra, Kathy, and Bert Bates. Head First Java, 2nd Edition. O'Reilly Media, 2005.
- Subramaniam, Venkat. Functional Programming in Java: Harnessing the Power of Java 8 Lambda Expressions. Pragmatic Bookshelf, 2014.

### WEBSITES:

- Geeks for Geeks. "Java Programming Language." Geeks for Geeks, 2023, <https://www.geeksforgeeks.org/java/>.
- Stack Overflow. "Questions tagged [java]." Stack Overflow, <https://stackoverflow.com/questions/tagged/java>.
- Tutorials Point. "Java Tutorial." Tutorials Point, 2023, <https://www.tutorialspoint.com/java/index.htm>.
- Vogella. "Java Tutorials." Vogella, 2023, <https://www.vogella.com/tutorials/java.html>.
- JetBrains. "IntelliJ IDEA Documentation." JetBrains, <https://www.jetbrains.com/help/idea/>.

# *APPENDICES*

## 8. APPENDICES

### A. DATA FLOW DIAGRAM

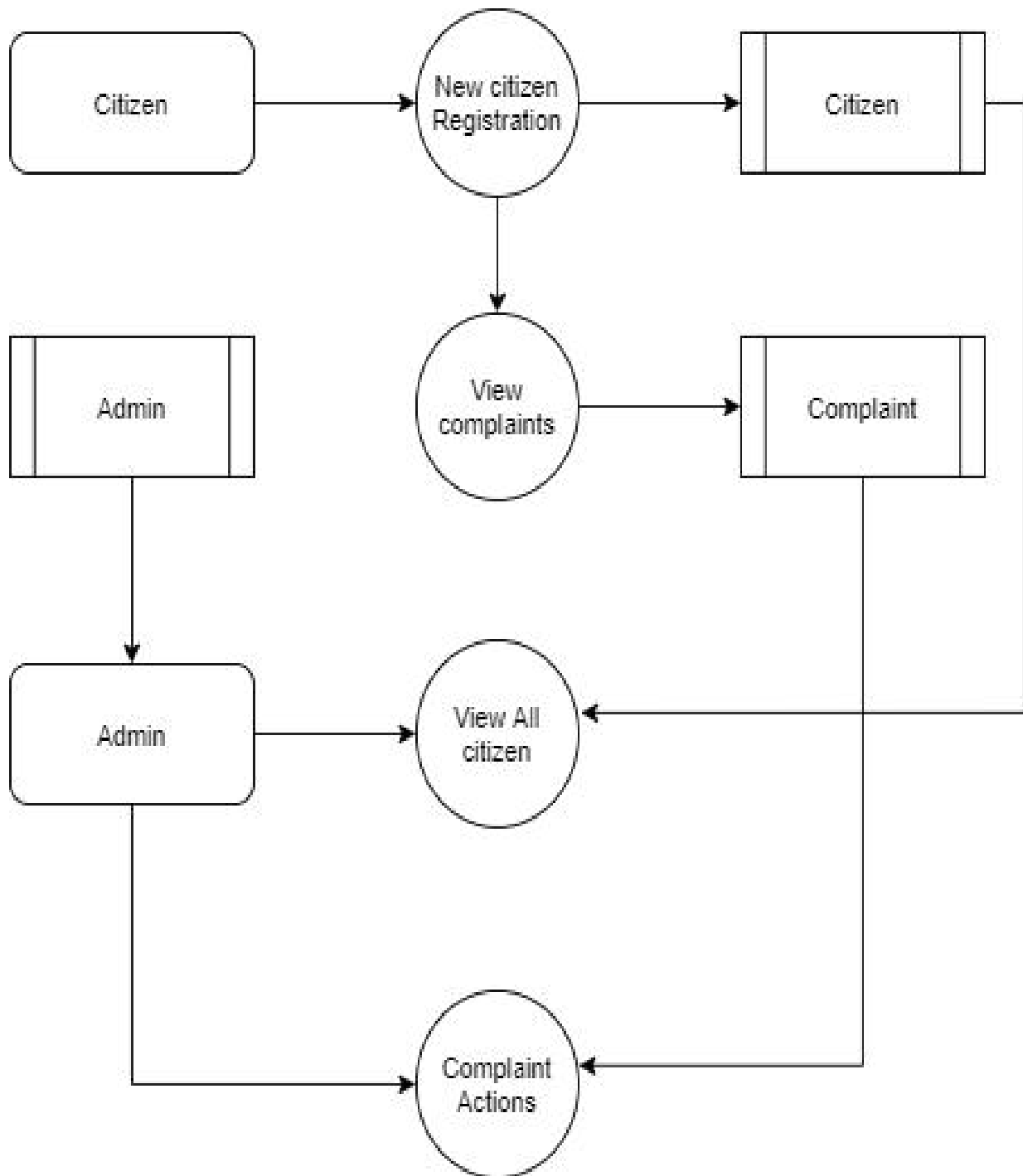
A data-flow diagram (DFD) is a way of representing a flow of a data of a process or system. The DFD also provides information about the outputs and inputs of each entity and process itself. A data-flow diagram is a part of structured-analysis modeling tools.

#### LEVEL 0:





**LEVEL 1:**



## B. TABLE STRUCTURE

The table needed for each module was designed and the specification of each and every column was given based on the records and details collected during record specification of the system study.

**TABLE NAME: ADMIN**

FIELD	DATATYPE	SIZE	CONSTRAINT
Admin id	Int	10	Primary key
Admin name	varchar	30	Not null
Admin mobile	Int	10	Not null
Password	varchar	20	Not null
address	varchar	50	Not null

**TABLE NAME: CITIZEN**

FIELD	DATATYPE	SIZE	CONSTRAINT
Citizen id	Int	10	Primary key
First name	varchar	20	Not null
Last name	Varchar	20	Not null
Mobile	Int	10	Not null
Full address	Varchar	50	Not null
City	Varchar	20	Not null
State	Varchar	20	Not null
Pincode	Int	6	Not null

**TABLE NAME: COMPLAINT**

<b>FIELD</b>	<b>DATATYPE</b>	<b>SIZE</b>	<b>CONSTRAINT</b>
Complaint id	Int	10	Primary key
Citizen id	Int	10	Foreign key
Mobile	Int	10	Not null
Complaint	Varchar	50	Not null
Full address	Varchar	50	Not null
Complaint reason	Varchar	20	Not null
Current status	Varchar	10	Not null

## C. SAMPLE CODING

```
package com.example.demo.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.demo.dao.ApiDao;
import com.example.demo.response.GetCitizenResponse;
import com.example.demo.response.GetComplaintResponse;
import com.example.demo.service.ApiService;

@RestController

@RequestMapping(value = { "/api" })

public class ApiController {

    @Autowired

    ApiService service;

    @Autowired

    ApiDao dao;

    @GetMapping("/login/{username}/{password}")

    public String login(@PathVariable String username,@PathVariable String password) {

        return service.login(username,password);

    }

}
```

```
@GetMapping("/citizen_register/{firstname}/{lastname}/{mobile}/{username}/{password}/{address}/{city}/{state}/{pincode}")
```

```
public String member_register(@PathVariable String firstname,  
  
    @PathVariable String lastname,  
  
    @PathVariable String mobile,  
  
    @PathVariable String address,  
  
    @PathVariable String city,  
  
    @PathVariable String state,  
  
    @PathVariable String pincode,  
  
    @PathVariable String username,  
  
    @PathVariable String password) {  
  
dao.citizen_register(firstname,lastname,mobile,address,city,state,pincode,username,password)  
;  
  
    return "Citizen Saved Sucessfully";  
  
}
```

```
@GetMapping("/get_citizen")
```

```
public ResponseEntity<List<GetCitizenResponse>> get_material() {  
  
    return ResponseEntity.ok().body(service.get_citizen());  
  
}
```

```
@GetMapping("/add_complaint/{citizen_id}/{mobile}/{address}/{reason}")
```

```
public String add_complaint(@PathVariable Integer citizen_id,  
  
    @PathVariable String mobile,  
  
    @PathVariable String address,  
  
    @PathVariable String reason
```

```

    ) {

        dao.add_complaint(citizen_id,mobile,address,reason);

        return "Complaint Saved Sucessfully";

    }

    @GetMapping("/complaint_action/{complaint_id}/{status}")

    public String add_complaint(@PathVariable Integer complaint_id,@PathVariable
String status

    ) {

        dao.complaint_action(complaint_id,status);

        return "Complaint Saved Updated";

    }

    @GetMapping("/get_complaints")

    public ResponseEntity<List<GetComplaintResponse>> get_complaints() {

        return ResponseEntity.ok().body(service.get_complaints());

    }

    @GetMapping("/get_complaints/{id}")

    public

        ResponseEntity<List<GetComplaintResponse>>

get_complaints1(@PathVariable Integer id) {

        return ResponseEntity.ok().body(service.get_complaints(id));

    }

}

package com.example.demo.configuration;

import java.util.Properties;

```

```

import javax.sql.DataSource;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.jdbc.datasource.DriverManagerDataSource;

import org.springframework.orm.hibernate5.HibernateTransactionManager;

import org.springframework.orm.hibernate5.LocalSessionFactoryBean;

import org.springframework.transaction.annotation.EnableTransactionManagement;

@Configuration

@EnableTransactionManagement

public class HibernateConfiguration {

    @Value("${db.driver}")

    private String DB_DRIVER;

    @Value("${db.password}")

    private String DB_PASSWORD;

    @Value("${db.url}")

    private String DB_URL;

    @Value("${db.username}")

    private String DB_USERNAME;

    @Value("${hibernate.dialect}")

    private String HIBERNATE_DIALECT;

    @Value("${hibernate.show_sql}")

    private String HIBERNATE_SHOW_SQL;

```

```

@Value("${hibernate.hbm2ddl.auto}")

private String HIBERNATE_HBM2DDL_AUTO;

@Value("${entitymanager.packagesToScan}")

private String ENTITYMANAGER_PACKAGES_TO_SCAN;

@Bean

public LocalSessionFactoryBean sessionFactory() {

    LocalSessionFactoryBean sessionFactory = new LocalSessionFactoryBean();

    sessionFactory.setDataSource(dataSource());

    sessionFactory.setPackagesToScan(ENTITYMANAGER_PACKAGES_TO_SCAN);

    Properties hibernateProperties = new Properties();

    hibernateProperties.put("hibernate.dialect", HIBERNATE_DIALECT);

    hibernateProperties.put("hibernate.show_sql", HIBERNATE_SHOW_SQL);

    //                                hibernateProperties.put("hibernate.hbm2ddl.auto",
HIBERNATE_HBM2DDL_AUTO);

    sessionFactory.setHibernateProperties(hibernateProperties);

    return sessionFactory;

}

@Bean

public DataSource dataSource() {

    DriverManagerDataSource dataSource = new DriverManagerDataSource();

    dataSource.setDriverClassName(DB_DRIVER);

    dataSource.setUrl(DB_URL);

    dataSource.setUsername(DB_USERNAME);

```



```

        dataSource.setPassword(DB_PASSWORD);

        return dataSource;
    }

    @Bean

    public HibernateTransactionManager transactionManager() {

        HibernateTransactionManager txManager = new HibernateTransactionManager();

        txManager.setSessionFactory(sessionFactory().getObject());

        return txManager;
    }
}

package com.example.demo.configuration;

import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.CorsRegistry;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;

@Configuration

@EnableWebMvc

public class WebConfig extends WebMvcConfigurerAdapter {

    @Override

    public void addCorsMappings(CorsRegistry registry) {

        registry.addMapping("/**");
    }
}

```

```

}

package com.example.demo.dao;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.List;

import javax.transaction.Transactional;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.NativeQuery;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Repository;

@Repository

@Transactional

public class ApiDao {

    @Autowired

    SessionFactory sf;

    public void citizen _ register (String first name, String last name, String mobile,
String address, String city, String state, String pincode, String username, String password) {

        // TODO Auto-generated method stub

        Session session = sf.getCurrentSession();

        String sql = "INSERT INTO `citizen` (`id`, `fname`, `lname`, `mobile`,
`address`, `city`, `state`, `pincode`, `username`, `password`) VALUES "

```

```
        + "(NULL, '"+firstname+"', '"+lastname+"', '"+mobile+"', '"+address+"', '"+city+"',  
        '"+state+"', '"+pincode+"', '"+username+"', '"+password+'");";
```

```
        session.createSQLQuery(sql).executeUpdate();
```

```
    }
```

```
    public void complaint_action(Integer complaint_id,String status) {
```

```
        // TODO Auto-generated method stub
```

```
        Session session = sf.getCurrentSession();
```

```
        String sql = "UPDATE `complaint` SET `status` = '"+status+"' WHERE  
        `complaint`.`id` = '"+complaint_id+'";
```

```
        session.createSQLQuery(sql).executeUpdate();
```

```
    }
```

```
    public List<Object[]> get_citizen() {
```

```
        // TODO Auto-generated method stub
```

```
        Session session = sf.getCurrentSession();
```

```
        String sql = "Select * from citizen";
```

```
        NativeQuery nq = session.createNativeQuery(sql);
```

```
        return nq.list();
```

```
    }
```

```
    public List<Object[]> get_complaints() {
```

```
        // TODO Auto-generated method stub
```

```
        Session session = sf.getCurrentSession();
```

```
        String sql = "Select * from complaint";
```

```
        NativeQuery nq = session.createNativeQuery(sql);
```

```

        return nq.list();

    }

    public List<Object[]> get_complaints(Integer id) {

        // TODO Auto-generated method stub

        Session session = sf.getCurrentSession();

        String sql = "Select * from complaint where citizen_id="+id;

        NativeQuery nq = session.createNativeQuery(sql);

        return nq.list();

    }

    public void add_complaint(Integer citizen_id,String mobile,String address,String
reason) {

        // TODO Auto-generated method stub

        Session session = sf.getCurrentSession();

        String sql = "INSERT INTO `complaint` (`id`, `citizen_id`, `mobile`,
`address`, `reason`, `status`) VALUES "

            + "(NULL, '"+citizen_id+"', '"+mobile+"', '"+address+"',
 '"+reason+"', '1');"

        session.createSQLQuery(sql).executeUpdate();

    }

    public String login(String username, String password) {

        // TODO Auto-generated method stub

        Session session = sf.getCurrentSession();

```

```
String sql = "select * from admin where username='"+username+"' and  
password='"+password+"'";;
```

```
NativeQuery nq = session.createNativeQuery(sql);
```

```
if (nq.list().size() != 0) {
```

```
    return "admin";
```

```
} else {
```

```
String sql1 = "select * from citizen where username='"+username+"' and  
password='"+password+"'";;
```

```
NativeQuery nq1 = session.createNativeQuery(sql1);
```

```
if (nq1.list().size() != 0) {
```

```
    List<Object[]> a = nq1.list();
```

```
    return "id="+a.get(0)[0];
```

```
} else {
```

```
    return "Invalid";
```

```
}
```

```
}
```

```
}
```

```
package com.example.demo.configuration;
```

```
import java.util.Properties;
```

```
import javax.sql.DataSource;
```

```
import org.springframework.beans.factory.annotation.Value;
```

```
import org.springframework.context.annotation.Bean;
```

```
import org.springframework.context.annotation.Configuration;
```

```

import org.springframework.jdbc.datasource.DriverManagerDataSource;

import org.springframework.orm.hibernate5.HibernateTransactionManager;

import org.springframework.orm.hibernate5.LocalSessionFactoryBean;

import org.springframework.transaction.annotation.EnableTransactionManagement;

@Configuration

@EnableTransactionManagement

public class HibernateConfiguration {

    @Value("${db.driver}")

    private String DB_DRIVER;

    @Value("${db.password}")

    private String DB_PASSWORD;

    @Value("${db.url}")

    private String DB_URL;

    @Value("${db.username}")

    private String DB_USERNAME;

    @Value("${hibernate.dialect}")

    private String HIBERNATE_DIALECT;

    @Value("${hibernate.show_sql}")

    private String HIBERNATE_SHOW_SQL;

    // @Value("${hibernate.hbm2ddl.auto}")

    private String HIBERNATE_HBM2DDL_AUTO;

    @Value("${entitymanager.packagesToScan}")

    private String ENTITYMANAGER_PACKAGES_TO_SCAN;

```

@Bean

```
public LocalSessionFactoryBean sessionFactory() {  
  
    LocalSessionFactoryBean sessionFactory = new LocalSessionFactoryBean();  
  
    sessionFactory.setDataSource(dataSource());  
  
    sessionFactory.setPackagesToScan(ENTITYMANAGER_PACKAGES_TO_SCAN);  
  
    Properties hibernateProperties = new Properties();  
  
    hibernateProperties.put("hibernate.dialect", HIBERNATE_DIALECT);  
  
    hibernateProperties.put("hibernate.show_sql", HIBERNATE_SHOW_SQL);  
  
    //                                hibernateProperties.put("hibernate.hbm2ddl.auto",  
HIBERNATE_HBM2DDL_AUTO);  
  
    sessionFactory.setHibernateProperties(hibernateProperties);  
  
    return sessionFactory;  
  
}
```

@Bean

```
public DataSource dataSource() {  
  
    DriverManagerDataSource dataSource = new DriverManagerDataSource();  
  
    dataSource.setDriverClassName(DB_DRIVER);  
  
    dataSource.setUrl(DB_URL);  
  
    dataSource.setUsername(DB_USERNAME);  
  
    dataSource.setPassword(DB_PASSWORD);  
  
    return dataSource;  
  
}
```

@Bean

```

public HibernateTransactionManager transactionManager() {

    HibernateTransactionManager txManager = new HibernateTransactionManager();

    txManager.setSessionFactory(sessionFactory().getObject());

    return txManager;

}

}

package com.example.demo.configuration;


import org.springframework.context.annotation.Configuration;

import org.springframework.web.servlet.config.annotation.CorsRegistry;

import org.springframework.web.servlet.config.annotation.EnableWebMvc;

import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;

@Configuration

@EnableWebMvc

public class WebConfig extends WebMvcConfigurerAdapter {

    @Override

    public void addCorsMappings(CorsRegistry registry) {

        registry.addMapping("/**");

    }

}

package com.example.demo.dao;

```



```

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.List;

import javax.transaction.Transactional;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.query.NativeQuery;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Repository;

@Repository

@Transactional

public class ApiDao {

    @Autowired

    SessionFactory sf;

    public void citizen _ register String first name, String last name, String mobile, String
address, String city, String state, String pincode, String username, String password) {

        // TODO Auto-generated method stub

        Session session = sf.getCurrentSession();

        String sql = "INSERT INTO `citizen` (`id`, `fname`, `lname`, `mobile`,
`address`, `city`, `state`, `pincode`, `username`, `password`) VALUES "

            + "(NULL, '"+firstname+"', '"+lastname+"', '"+mobile+'",
 '"+address+"', '"+city+"', '"+state+"', '"+pincode+"', '"+username+"', '"+password+"');";

        session.createSQLQuery(sql).executeUpdate();

```

```

    }

    public void complaint_action(Integer complaint_id,String status) {

        // TODO Auto-generated method stub

        Session session = sf.getCurrentSession();

        String sql = "UPDATE `complaint` SET `status` = '"+status+"' WHERE
`complaint`.`id` = "+complaint_id+";";

        session.createSQLQuery(sql).executeUpdate();

    }

    public List<Object[]> get_citizen() {

        // TODO Auto-generated method stub

        Session session = sf.getCurrentSession();

        String sql = "Select * from citizen";

        NativeQuery nq = session.createNativeQuery(sql);

        return nq.list();

    }

    public List<Object[]> get_complaints() {

        // TODO Auto-generated method stub

        Session session = sf.getCurrentSession();

        String sql = "Select * from complaint";

        NativeQuery nq = session.createNativeQuery(sql);

        return nq.list();

    }

```

```

public List<Object[]> get_complaints(Integer id) {

    // TODO Auto-generated method stub

    Session session = sf.getCurrentSession();

    String sql = "Select * from complaint where citizen_id="+id;

    NativeQuery nq = session.createNativeQuery(sql);

    return nq.list();

}

public void add_complaint(Integer citizen_id, String mobile, String address, String reason) {

    // TODO Auto-generated method stub

    Session session = sf.getCurrentSession();

    String sql = "INSERT INTO `complaint` (`id`, `citizen_id`, `mobile`,
`address`, `reason`, `status`) VALUES "

        + "(NULL, '"+citizen_id+"', '"+mobile+"', '"+address+"',
 '"+reason+"', '1')";

    session.createSQLQuery(sql).executeUpdate();

}

public String login(String username, String password) {

    // TODO Auto-generated method stub

    Session session = sf.getCurrentSession();

    String sql = "select * from admin where username='"+username+"' and
password='"+password+"'";

    NativeQuery nq = session.createNativeQuery(sql);

    if (nq.list().size() != 0) {

        return "admin";
    }
}

```

```

    } else {

        String sql1 = "select * from citizen where username='"+username+"'
and password='"+password+"'";

        NativeQuery nq1 = session.createNativeQuery(sql1);

        if (nq1.list().size() != 0) {

            List<Object[]> a = nq1.list();

            return "id="+a.get(0)[0];

        }else {

            return "Invalid";

        }

    }


}

}

```

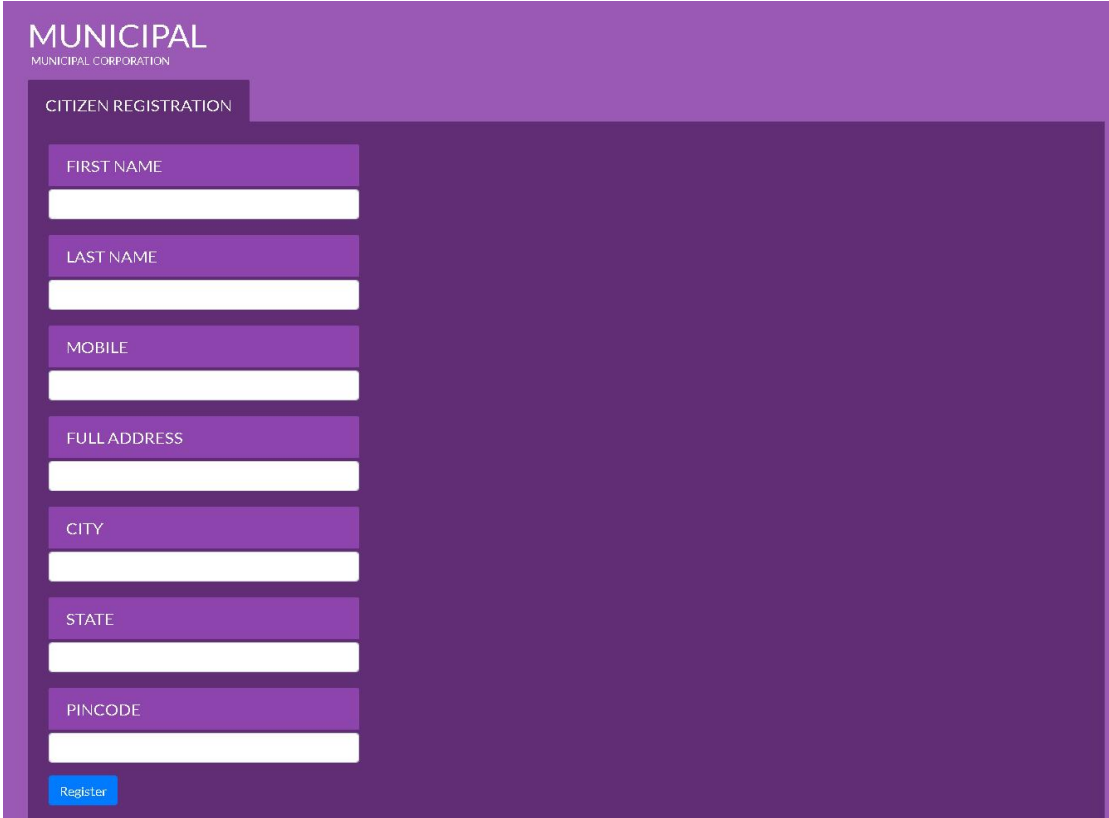
## D. SAMPLE INPUT

### ADMIN LOGIN



The image shows a login form titled "Municipal Corporation" on a light gray background. The form is contained within a white box with a subtle shadow. It features two input fields: the first is for the username, containing the text "admin"; the second is for the password, represented by five dots. Below these fields are two blue buttons: "Login" and "Register".

### CITIZEN REGISTRATION



The image displays a registration form titled "MUNICIPAL CORPORATION" and "CITIZEN REGISTRATION" on a purple background. The form is a vertical stack of white input fields, each with a label above it: "FIRST NAME", "LAST NAME", "MOBILE", "FULL ADDRESS", "CITY", "STATE", and "PINCODE". At the bottom of the form is a blue "Register" button.

## CITIZEN ENTER THE DETAILS

MUNICIPAL CORPORATION

CITIZEN REGISTRATION

FIRST NAME

karthika

LAST NAME

K

MOBILE

9829942292

USERNAME

karthi

PASSWORD

1234

FULL ADDRESS

26,jaisankarni nagar 2nd street golden nagar

CITY

tirupur

STATE

tamilnadu

PINCODE

641607

Register

## CITIZEN LOGIN

Municipal Corporation

karthika

....

Login

Register

## CITIZENS ADD COMPLAINT

MUNICIPAL

CORPORATION

ADD COMPLAINT

VIEW COMPLAINT STATUS

LOGOUT

MOBILE

9629843290

COMPLAINT

street light problem

FULL ADDRESS

26,jayalakshmi Nagar 2nd street golden Nagar

Add Complaint

## E. SAMPLE OUTPUT

### CITIZEN VIEW COMPLAIT STATUS

MUNICIPAL  
CORPORATION

ADD COMPLAINT

VIEW COMPLAINT STATUS

LOGOUT

#	Mobile	Complaint	Full Address	Current Status
1	9876543210	Water supply problem	Thekkalur, avinashi(po)	Action Requested

### VIEW ALL CITIZENS DETAILS

MUNICIPAL  
CORPORATION

ALL CITIZEN'S

VIEW ALL COMPLAINTS STATUS

COMPLAINT ACTIONS

LOGOUT

#	First Name	Last Name	Mobile	Full Address	City	State	Pincode
1	karthika	k	9629843290	26,jayalakshmi nagar 2nd street golden nagar	tirupur	tamilnadu	641607
2	keerthana	k	9876543210	Thekkalur Avinashi(po)	tirupur	taminadu	641633



## ADMIN TAKE COMPLAINT ACTIONS

MUNICIPAL  
CORPORATION

ALL CITIZEN'S

VIEW ALL COMPLAINTS STATUS

COMPLAINT ACTIONS

LOGOUT

#	Mobile	Complaint	Full Address	Take Action
1	9629843290	street light problem	26,jayalakshmi Nagar 2nd street golden Nagar	<div>In Review</div>
1	9876543210	Water supply problem	Thekkalur, avinashi(po)	<div>Action Requested</div>

## ADMIN VIEW COMPLAINT STATUS

MUNICIPAL  
CORPORATION

ALL CITIZEN'S

VIEW ALL COMPLAINTS STATUS

COMPLAINT ACTIONS

LOGOUT

#	Mobile	Complaint	Full Address	Current Status
1	9629843290	street light problem	26,jayalakshmi Nagar 2nd street golden Nagar	Action Requested
2	9876543210	Water supply problem	Thekkalur, avinashi(po)	Action Requested

CITIZENS VIEW COMPLAINT STATUS

MUNICIPAL  
CORPORATION

ADD COMPLAINT

VIEW COMPLAINT STATUS

LOGOUT

#	Mobile	Complaint	Full Address	Current Status
1	9629843290	street light problem	26,jayalakshmi Nagar 2nd street golden Nagar	In Review