INTERNSHIP REPORT

Online Complaints Portal Using Flask

A Report Submitted to

Jawaharlal Nehru Technological University Kakinada, Kakinada

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Name: KANDALA A V N VAMSI KRISHNA SAI
Regd No: 21KN1A0582

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NRI INSTITUTE OF TECHNOLOGY

Autonomous

(Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada)
Accredited by NBA (CSE, ECE & EEE), Accredited by NAAC with 'A' Grade
ISO 9001: 2015 Certified Institution
Pothavarappadu (V), (Via) Nunna, Agiripalli (M), Krishna Dist., PIN: 521212, A.P, India.

2023-2024

NRI INSTITUTE OF TECHNOLOGY

(An Autonomous Institution, Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada)

Accredited by NBA (CSE, ECE & EEE), Accredited by NAAC with 'A' Grade
ISO 9001: 2015 Certified Institution

Pothavarappadu (V), (Via) Nunna, Agiripalli (M), Krishna Dist., PIN: 521212, A.P, India.

CERTIFICATE

This is to certify that the "**Internship report**" submitted by **KANDALA A V N VAMSI KRISHNA (21KN1A0582),**is work done by him and submitted during 2022-2023 academic year, in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING,** at **Codegnan IT SOLUTIONS PVT LTD.**

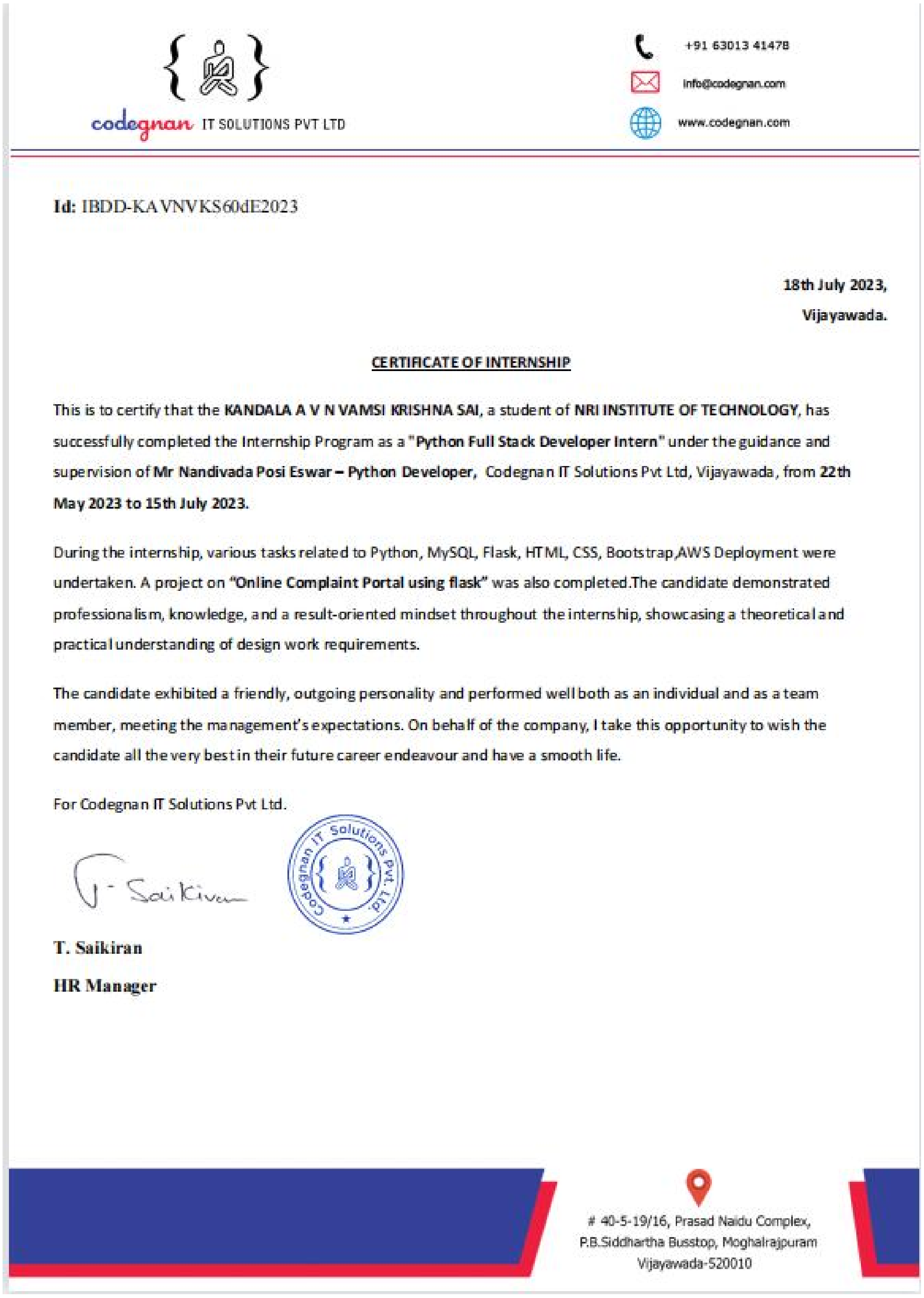
.

INTERNSHIP COORDINATOR

Head of the Department

(Dr. D. SUNEETHA)

CERTIFICATE OF INTERNSHIP



codegnan IT SOLUTIONS PVT LTD

+91 63013 41478
info@codegnan.com
www.codegnan.com

Id: IBDD-KAVNVKS60dE2023

18th July 2023,
Vijayawada.

CERTIFICATE OF INTERNSHIP

This is to certify that the **KANDALA A V N VAMSI KRISHNA SAI**, a student of **NRI INSTITUTE OF TECHNOLOGY**, has successfully completed the Internship Program as a **"Python Full Stack Developer Intern"** under the guidance and supervision of **Mr Nandivada Posi Eswar – Python Developer**, Codegnan IT Solutions Pvt Ltd, Vijayawada, from **22th May 2023 to 15th July 2023.**

During the internship, various tasks related to Python, MySQL, Flask, HTML, CSS, Bootstrap,AWS Deployment were undertaken. A project on **"Online Complaint Portal using flask"** was also completed.The candidate demonstrated professionalism, knowledge, and a result-oriented mindset throughout the internship, showcasing a theoretical and practical understanding of design work requirements.

The candidate exhibited a friendly, outgoing personality and performed well both as an individual and as a team member, meeting the management's expectations. On behalf of the company, I take this opportunity to wish the candidate all the very best in their future career endeavour and have a smooth life.

For Codegnan IT Solutions Pvt Ltd.

T. Saikiran

Codegnan IT Solutions Pvt. Ltd.

T. Saikiran

HR Manager

40-5-19/16, Prasad Naidu Complex,
P.B.Siddhartha Busstop, Moghalrajpuram
Vijayawada-520010

ACKNOWLEDGEMENT

We take this opportunity to thank all who have rendered their full support to our work. The pleasure, the achievement, the glory, the satisfaction, the reward, the appreciation and the construction of our project cannot be expressed with a few words for their valuable suggestions.

We are expressing our heartfelt thanks to **Head of the Department, Dr. D. SUNEETHA** garu for her continuous guidance for completion of our Project work.

We are extending our sincere thanks to **Dean of the Department, Dr. K. V. SAMBASIVA RAO** for his continuous guidance and support to complete our project successfully.

We are thankful to the **Principal, Dr. C. NAGA BHASKAR** garu for his encouragement to complete the Project work.

We are extending my sincere and honest thanks to the **Chairman, Dr. R. VENKATA RAO garu & Secretary, Sri K. Sridhar** garu for their continuous support in completing the Project work.

Finally, we thank the Administrative Officer, Staff Members, Faculty of Department of CSE, NRI Institute of Technology and my friends, directly or indirectly helped us in the completion of this project.

Name: KANDALA A V N VAMSI KRISHNA SAI Regd No: 21KN1A0582

INDEX

1. Abstract :

The emergence of the internet and digital technologies has transformed various aspects of our lives, including how individuals interact with businesses and organizations. However, with this increased connectivity, new challenges have arisen, such as the need for an efficient and effective system to handle customer complaints. This project abstract introduces an online complaints portal, aimed at revolutionizing the way grievances are registered, tracked, and resolved.

The Online Complaints Portal (OCP) is a web-based platform designed to provide users with a streamlined and user-friendly interface for submitting and managing their complaints. It serves as a centralized hub for individuals to report issues they have encountered with organizations.

The OCP features a simple and intuitive complaint registration process, where users can provide detailed information about their complaint, The portal also incorporates intelligent categorization to ensure that complaints are directed to the Administrator within the organization.

Upon Complaint registration, users receive a unique complaint ID and gain access to a personalized dashboard where they can track the progress of their complaint.

Firstly, it provides a convenient and accessible platform for individuals to voice their concerns, reducing the frustration and inconvenience associated with traditional complaint handling methods. Additionally, it enables organizations to collect valuable data and insights to improve their customer satisfaction. Secondly, The Administrator insight the Complaints registered and access to update the status of the regarding Complaint which the user can track the status of the complaint in user dashboard.

In conclusion, the Online Complaints Portal revolutionizes the way organizations manage and address complaints. By providing a transparent and efficient platform, it empowers stakeholders to raise their concerns, promotes effective communication, and enables organizations to proactively resolve issues. Ultimately enhancing stakeholder satisfaction and organizational effectiveness.

Main Objective:
In this project Online Complaints Portal Using Flask helps the users and administrator to give complaint about their issue and review the complaints, updating the status in an easy way.
This biggest advantage using this software is that it reduces the time taken to give complaint and review the complaints, updating.
Using this software, we can:

- We can give complaint by giving subject and problem description.
- We can get the status of our complaints using the complaint number.
- We can update the status of the complaint using the complaint number as an administrator.

2. **Introduction:**

Online Complaints Portal is involving in the Creation of Web Application using Python Flask including CRUD operations with connection of Database. The Project is Eco-Friendly and it is more interactive with the user for registering the accounts in the website and registering their complaints regarding their problems. And can see the status of their registered complaints. When the admin updates the status about the complaints listed then the respective user will get know the status about their complaint.

CRUD OPERATIONS:

CREATE Operation: Performs the INSERT statement to create a new record.

READ Operation: Reads table records based on the input parameter.

UPDATE Operation: Executes an update statement on the table. It is based on the input parameter.

DELETE Operation: Deletes a specified row in the table. It is also based on the input parameter.

2.1 EXISTING SYSTEM

Many people and officials face a lot of difficult to register and resolve the various problems at one place. Registering and resolving of various complaints is manually difficult. Large number of complaints have to be handled where the officials find difficult to interact with different people and the people also find difficult to find various departments officials that they require at only one place.

Disadvantages: More manual work. Time consuming. Needs more manual interactions.

2.2PROPOSED SYSTEM:

Online Complaints Portal is a medium where the people can register their complaints and people can mention and register their problems at only one place. It provides a good interface for both officials and people. The project decreases the manual work of both officials and people. People can register, login to their accounts and can add the complaints that they wish to register, whereas, the Officials can login to their accounts and can resolve the complaints that are available on Grievance box in an easy and efficient way.

2.3 Hardware and Software Requirements

Hardware and software specifications

HARDWARE AND SOFTWARE INTERFACES:

HARDWARE REQUIREMENTS

Client Site:

Processor	: Intel Pentium IV
Speed	: 2.00GHZ
RAM	: 1GB
Hard Disk	: 150 GB
Key Board (104 keys)	: Standard
Screen Resolution	: 1024 x 764 Pixels

Server Site:

Processor	: Intel Pentium IV
Speed	: 2.00GHZ
RAM	: 1GB
Hard Disk	: 150 GB
Key Board (104 keys)	: Standard
Screen Resolution	: 1024 x 764 Pixels

SOFTWARE REQUIREMENTS

OPERATING SYSTEM	: WINDOWS XP AND ABOVE
DATA BASE	: MySQL - mysql-connector-java-8.0.13.jar
PROGRAMMING LANGUAGE	: Python 3
IDE	: Visual Studio Code

2.4 FEASIBILITY STUDY:

Feasibility study is an important phase in the software development process. It enables the developer to have an assessment of the product being developed It refers to the feasibility study of the product in terms of outcomes of the product, operational use and technical support required for implementing it.

Feasibility study should be performed on the basis of various criteria and parameters. The various feasibility studies are:

- Economic Feasibility
- Operational Feasibility
- Technical Feasibility

Economic Feasibility: It refers to the benefits or outcomes we are deriving from the product compared to the total cost we are spending for developing the product. If the benefits are more or less the same as the older system, then it is not feasible to develop the product.

In the present system, the development of the new product greatly enhances the accuracy of the system and cuts short the delay in the processing of application.

The errors can be greatly reduced and at the same time providing a great level of security. Here we don't need any additional equipment except memory of required capacity. No need for spending money on client for maintenance because the database used is web enabled database.

Operational Feasibility: It refers to the feasibility of the product to be operational. Some products may work very well at design and implementation but may fail in the real time environment. It includes the study of additional human resource required and their technical expertise.

In the present system, all the operations can be performed easily compared to existing system and supports for the backlog data. Hence there is need for additional analysis. It was found that the additional modules added are isolated modules as far as the operational is concerned, so the Developed system is operationally feasible.

Technical Feasibility: It refers to whether the software that is available in the market fully supports the present application. It studies the pros and cons of using particular software for the development and its feasibility. It also studies the additional training needed to be given to the people to make the application work.

In the present system, the user interface is user friendly and does not require much expertise and training. It just needs a mouse click to do any sort of application. The software that is used for developing is server pages fully is highly suitable for the present application since the users require fast access to the web pages and with a high degree of security. This is achieved through integration of web server and database server in the same environment.

Implementation plan:

The main plan for the system developed is to upgrading existing system to the proposed system. There are mainly 4 methods of upgrading the existing system to proposed

- Parallel Run System
- Direct Cut-Over System
- Pilot System
- Phase-in Method

Parallel Run System: It is the most secure method of converting from an existing to new system. In this approach both the systems run in parallel for a specific period of time. During that period if any serious problems were identified while using the new system, the new system is dropped and the older system is taken at the start point again.

Direct Cut -Over Method: In this approach a working version of the system is implemented in one part of the organization such as single work area or department. When the system is deemed complete it is installed through out the organization either all at once (direct cut-over) or gradually (phase-in).

Phase-in Method: In this method a part of the system is first implemented and over time other remaining parts are implemented.

Implementation planed used: The workflow Management system is developed on the basis of "Parallel Run Method" because we upgraded the system, which is already in use to fulfill the requirements of the client. The system already in use is treated as the old system and the new system is developed on the basis of the old system and maintained the standards processed by the older system. The upgraded system is working well and is implemented on the client successfully. of the candidate recruitment.

Project Plan

It was decided to use good Software engineering principals in the development of the system since the company had quite a big client network & was aiming to provide staffing for the clients or to develop the internal projects of the companies& expand their operations in the near future. So the following Project Plan was drawn up:

1. The Analysts will interact with the current manual system users to get the Requirements. As a part of this the Requirements Specification Document will be created.

2. The requirements Specifications document will contain the Analysis & Design of the system & ML will be used as the modelling language to express the Analysis & Design of the System. According to Grady Booch et al, in The Unified Modelling Language User Guide **[UML-1998],** "The Unified Modelling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software intensive system. The UML gives you a standard way to write a system's blueprints, covering conceptual things, such as business processes and system functions, as well as concrete things, such as classes written in a specific programming language, database schemas, and reusable software components".

3. The Analysis, Design, Implementation & testing of the System itself will be broadly based on the Rational Unified Software Development process. According to Ivar Jacobson et al, in The Unified Software Development Process (The Addison-Wesley Object Technology Series) **[USDP-2000]**, the Unified Software Development Process contains Inception, Elaboration, Construction & Transition as the main Phases, which contain further cycles & iterations. This process will be followed to produce an incremental cycle, which will deliver milestones like the Requirements Specification Document etc., at the end of each of the iterations, Phases or cycles.

4. The Architecture & Technologies will be decided as a part of the Analysis of the requirements.

5. Once the Design is ready the Implementation & Testing strategy of the system will commence. Each will be independent of the other. The implementation of the system itself will be broken down into sub-systems following the Software Engineering principles for the development of robust software.

6. Once the implementation is ready, the System testing will take place. If the system is judged to be stable then Acceptance testing by the Users will take place & once the Users are satisfied the System will be rolled out to the Users & they will be trained on how to use it for an initial period.

The following chapters contain an account of how the Technology & architecture for the system were chosen.

3.Requirements Specification Document

3.1 Introduction

According to Roger Pressman in Software Engineering: A Practitioner's Approach (McGraw-Hill Publications) **[SEPA–1997]**, the requirement specification document is produced at the end of Analysis of the system. This document is a very comprehensive document & contains all the User requirements & Analysis diagrams. The Requirements are broadly divided into two groups:

1. Functional requirements

2. Non-functional requirements

3.2 Functional Requirements

The main purpose of functional requirements within the requirement specification document is to define all the activities or operations that take place in the system. These are derived through interactions with the users of the system. Since the Requirements Specification is a comprehensive document & contains a lot of data, it has been broken down into different Chapters in this report. The depiction of the Design of the System in UML is presented in a separate chapter. The Data Dictionary is presented in the Appendix of the system.

1. The System should allow the administrator to manage different levels of tests

   and their sequence.

2. It allows the administrator to manage the questions in each category.

3. It allows the administrator to manage the list of questions in each category.

4. Candidate can register himself for writing the test.

5. The system then takes the candidate the first level after logging in.

6. The system generates the test by generating questions randomly pick up the

   questions from the list

7. It allows the candidate to select answers of questions

8. This system finally evaluates the test, display the result and store it.

9. This system can then take the candidate to the next level.

10. It allows the administrator to generate the report bases on some cut off

    marks.

11. It allows the candidate the feedback

3.3 Non-Functional Requirements

The non-functional requirements consist of

1. Analysis, Design & Data requirements (Use-case diagrams, textual analysis, sequence diagrams, data dictionary etc.)
2. Constraints.
3. Guidelines.
4. Validation Criteria.

3.3.1 Analysis, Design & Data requirements

The use case diagrams, textual analysis and sequence diagrams & data dictionary fall into this category. Since each category above is of considerable importance, they have been dealt in separate chapters. An outline is only included here.

The Analysis & Design phases of the system yield Use Case diagrams, textual analysis, Sequence Diagrams, Class diagrams & Data Dictionary. Data dictionary consists of process statements showing how data is flowing from starting point to end point.

3.3.2 Constraints

These are the requirements that are not directly related to the functionality of the system. These should be considered as mandatory when the system is developed. The following Constraints were arrived at for the system:

1. The system should be available over the intranet so that the Users like the candidates can use the system from their system which was assigned to him.
2. For gaining entry into the system the users should be registered and should be able use login & passwords for gaining access to the system.
3. The users should be able to change their passwords for increased security.
4. The system should conform to the requirement specified and final deliverables of the project before some date.
5. The system should be easy to understand and organized in a structured way. The users should also receive feedback about any errors that occur.
6. There should be no limitation about the hardware platform that is to be used to run the

system.

7. Data integrity should be maintained if an error occurs or the whole system comes down.

8. A user should to be registered in the system once in 6 months only.

9. A user can take-up the next level test once he clears previous level.

3.3.3 Guidelines

We have discussed mandatory requirements in the previous section. The requirements in this section should be taken as suggestions & they should be thought of as recommendations to further enhance the usability of the system.

1. The system should display a menu for users to choose from.
2. The system should display users' requests in a reasonable time.
3. Services of the system should be available 24 hours a day.
4. The system should be designed in such a way that it is easy to enhance it with more functionality. It should be scalable & easily maintainable.

3.3.4 Validation Criteria

The Validation Criteria are dealt separately in the Chapter dealing with the Test Strategy & Test cases.

4. Architecture & Technologies

4.1. Introduction

General Methodology in Developing Software Project

The general methodology in developing a system in involved in different phases, which describe the system's life cycle model for developing software project. The concept includes not only forward motion but also have the possibility to return that is cycle back to an activity previously completed. This cycle back or feedback may occur as a result of the failure with the system to meet a performance objective or as a result of changes in redefinition of system activities. Like most systems that life cycle of the computer-based system also exhibits distinct phases.

Those are,

Requirement Analysis Phase

Design Phase

Development Phase

Coding Phase

Testing Phase

1. Requirement Analysis Phase:

This phase includes the identification of the problem, in order to identify the problem; we have to know information about the problem, the purpose of the evaluation for problem to be known. We have to clearly know about the client's requirements and the objectives of the project.

2. Design Phase:

Software design is a process through which the requirements are translated into a representation of software. One of the software requirements have been analyzed and specified, the software design involves three technical activities: design, coding generation and testing. The design of the system is in modular form i.e. the software is logically partitioned into components that perform specific functions and sub functions. The design phase leads to modules that exhibit independent functional characteristics. It even leads to interfaces that reduce the complexity of the connections between modules and with the external environment. The design phase is of main importance because in this activity, decisions ultimately affect the success of software implementation and maintenance.

3. Development Phase:

The development phase includes choosing of suitable software to solve the particular problem given. The various facilities and the sophistication in the selected software give a better development of the problem.

4. Coding Phase:

The coding phase is for translating the design of the system-produced during the design phase into code in a given programming language, which can be executed by a computer and which performs the computation specified by the design.

5. Testing Phase:

Testing is done in various ways such as testing the algorithm, programming code; sample data debugging is also one of following the above testing.

Requirement Specification:

Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specification are addressed during this activity.

The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase.

The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium through which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

Purpose:

The purpose of this document is to describe all external requirements or client provisioning. It also describes the interfaces for the system.

Scope:

This document is the only one that describes the requirements of the system. It is meant for the use by the developers, and will also use by the basis for validating the final delivered system. Any changes made to the requirements in the future will have to go through a formal change approval process. The developer is responsible for asking for clarifications, where necessary, and will not make any alternations without the permission of the client.

System Design:

Design:

Design of software involves conceiving, planning out and specifying the externally observable characteristics of the software product. We have data design, architectural design and user interface design in the design process. These are explained in the following section. The goal of design process is to provide a blue print for implementation, testing and maintenance activities.

The primary activity during data design is to select logical representations of data objects identified during requirement analysis and software analysis. A data dictionary explicitly represents the relationships among data objects and constraints on the elements of the data structure. A data dictionary should be established and used to define both data and program design.

Design process is in between the analysis and implementation process. The following design diagrams (Data Flow Diagrams and E-R Diagrams) make it easy to understand and implement

The design process for software system has two levels.

1. System Design or Top-Level Design.
2. Detailed Design or Logical Design.

System Design or Top-Level Design:

In the system design the focus is on deciding which modules are needed for the system, the specification of these modules and how these modules should be interconnected.

Detailed Design or Logical Design:

In detailed design the interconnection of the modules or how the specifications of the modules can be satisfied is decided. Some properties for a software system design are

- Verifiability.
- Completeness.
- Consistency.
- Trace ability.
- Simplicity/Understandability.

The Requirements provided by the users are converted into Users Requirement Specifications described above. The URS documents are then revised, validated, authorized and approved by the users. The development commences after the approval phase i.e. after the signing off of the URS documents. Thus, the URS is concerned to be the most important document from user and developer prospective. The Developer will try to adhere to the requirements specified in the URS documents in order to develop the required application. We have used Agile models a development model.

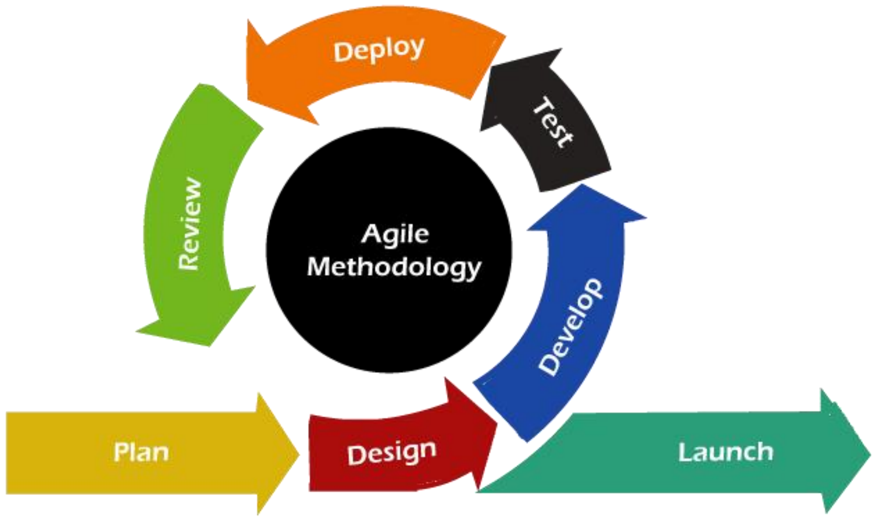


Fig. 2.1 Agile Methodology

4.2. Project Plan

Areas, which would be considered during the planning and analysis, would be:

- Ensure that the information flow is process driven.
- Reduce the manual effort so the maximum extent for all activities.
- Ensure validation at each and every level.
- Act as an effective tool in decision support.
- Provide user friendly system.

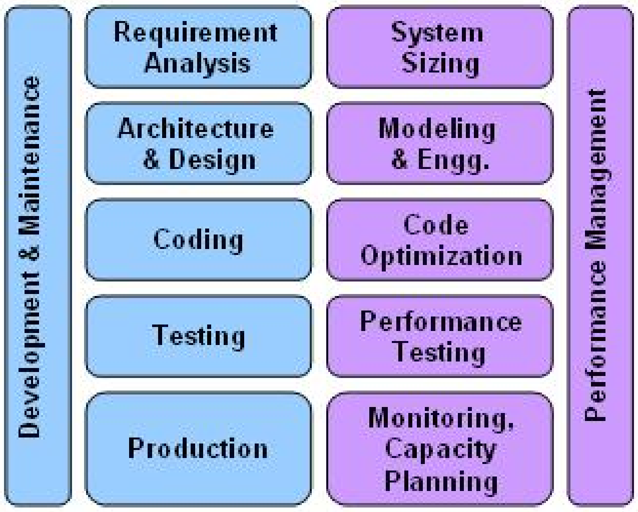


Fig. 2.2 Project Planning and Management Approach

4.3 SYSTEM ANALYSIS:

Analysis is the detailed study of the various operations performed by the system and their relationships within and outside of the system. A key question is: what must be done to solve the problem? One aspect of analysis is defining the boundaries of the system and determining whether or not candidate system should consider other related systems. During analysis, data are collected on the available files, decision points, and transactions handled by the present system.

System Requirement Specification

What is SRS?

Software Requirement Specification (SRS) is the starting point of the software developing activity. AS system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of the clients (the input) into a formal document (the output of the requirement phase).

The SRS phase consists of two basic activities:

1) Problem/Requirement Analysis: The process is order and more nebulous of the two, deals with understanding the problem, the goal and constraints.

2) Requirement Specification: Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity.

The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase.

Role of SRS:

The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium through which the client and the user needs are accurately specified. It forms the basis of the software development. A good SRS should satisfy all the parties involved in the system.

4.4. CLIENT SERVER MODEL

When an architect designs a building, he has a vision of the finished product and produces a result based on that vision. Client – server, on the other hand, is more like Darwinian model of evolution of a living species. No one has a vision of the finished products; rather, day-today events and gradual changes affect it over time in reaction to those events.

In the beginning, application was fairly simple, reading input transaction in a 'batch', processing them against a data store, and the output was paper. Record retrieval was usually a set of subroutines embedded in the updating program.

Common functions gradually migrated from the application to the operating system. Database processing was one of the first major functions to be removed from application control. Much of the time database functions in the application included retrieval, replacement and insertion. Since it was function had to be introduced database administration. This new function was separated from the application code and involved defining the structure of the database, value ranges backup, rollback, and so forth.

Advantages of Client – Server Model:

- ➢ The hardware and software can be placed where it will do the best.
- ➢ In Client – Server model PCs, the power can be spread across the client and the server.
- ➢ On client side, an Active X object is used to present data
- ➢ By having the client side, it can do more work
- ➢ The client software supplies the interface (Such as windowed program) and the knowledge of how to pass the request to the server and the format of the data for the user when it's returned from the server. The server's job is to manipulate the data according to the user's request.

4.5 TECHNOLOGIES

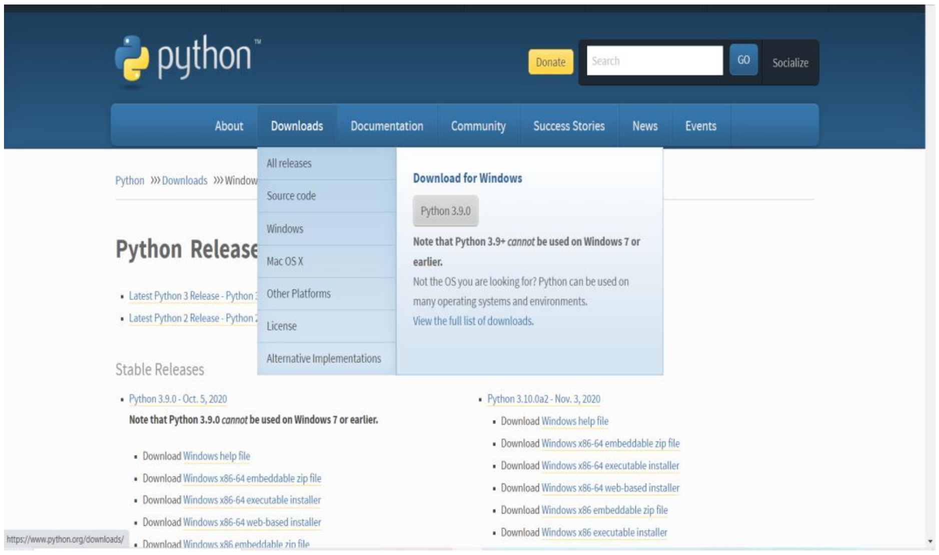
- IDE - Visual Studio Code
- Python 3
- MySQL - mysql-connector-java-8.0.13.jar

Python:

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems.

Python Installation and Setting Up Environment:

- Go to the www.python.org and click on the Downloads.
- Select the OS according to your Pc.



Python »» Downloads »» Window

Python Release

- Latest Python 3 Release - Python 3
- Latest Python 2 Release - Python 2

Download for Windows

Note that Python 3.9+ *cannot* be used on Windows 7 or earlier.

Not the OS you are looking for? Python can be used on many operating systems and environments.

View the full list of downloads.

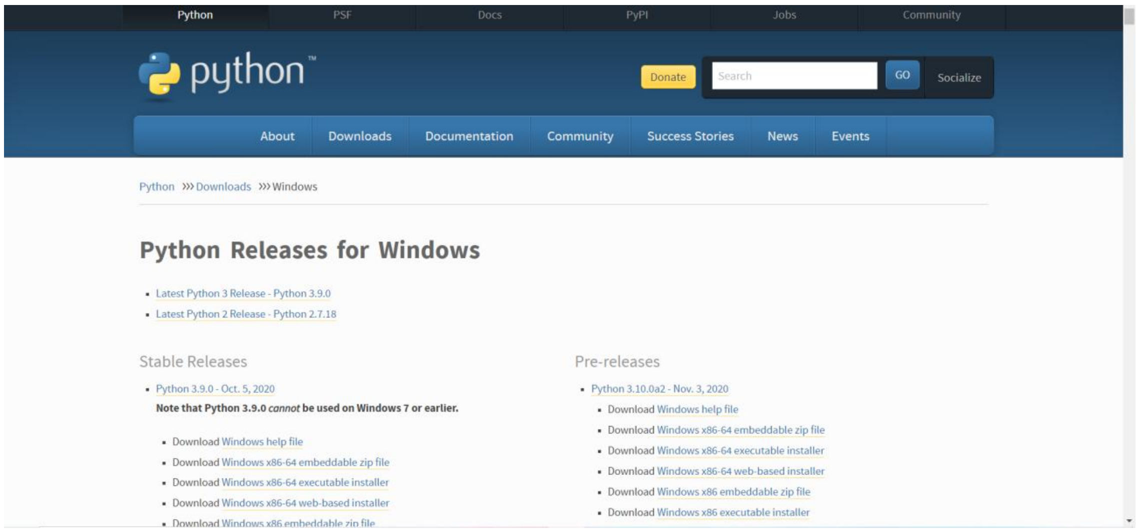
Stable Releases

- Python 3.9.0 - Oct. 5, 2020

  Note that Python 3.9.0 *cannot* be used on Windows 7 or earlier.
  - Download Windows help file
  - Download Windows x86-64 embeddable zip file
  - Download Windows x86-64 executable installer
  - Download Windows x86-64 web-based installer
  - Download Windows x86 embeddable zip file

- Python 3.10.0a2 - Nov. 3, 2020
  - Download Windows help file
  - Download Windows x86-64 embeddable zip file
  - Download Windows x86-64 executable installer
  - Download Windows x86-64 web-based installer
  - Download Windows x86 embeddable zip file
  - Download Windows x86 executable installer

- On the next page you will see all the version of python that are available for download.
- Python latest Version is 3.9.0 . Click on the link, It will start the downloading process.



Python >>> Downloads >>> Windows

Python Releases for Windows

- Latest Python 3 Release - Python 3.9.0
- Latest Python 2 Release - Python 2.7.18

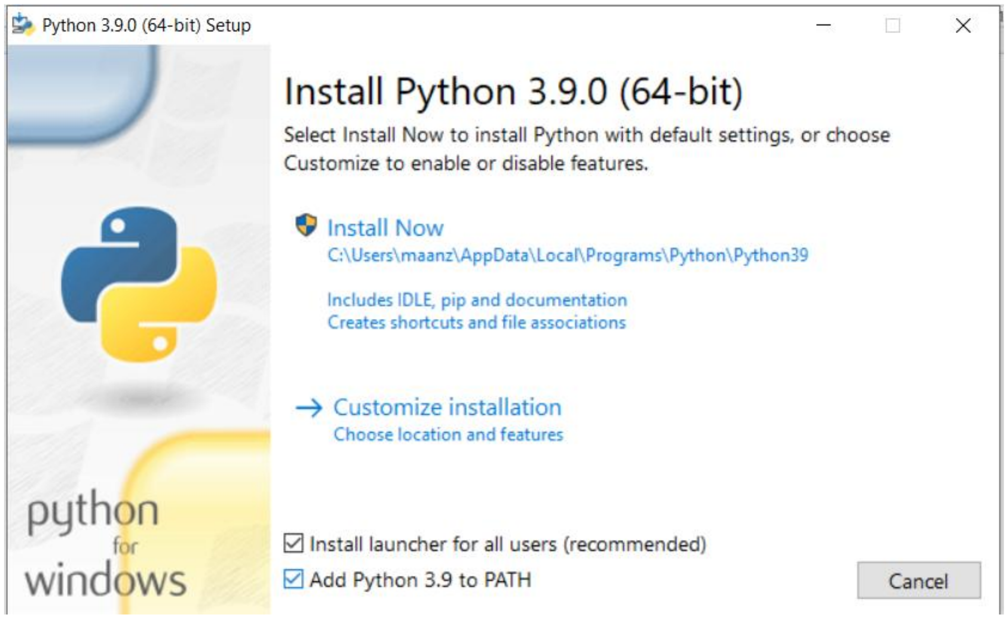
Stable Releases

- Python 3.9.0 - Oct. 5, 2020
  Note that Python 3.9.0 *cannot* be used on Windows 7 or earlier.
  - Download Windows help file
  - Download Windows x86-64 embeddable zip file
  - Download Windows x86-64 executable installer
  - Download Windows x86-64 web-based installer
  - Download Windows x86 embeddable zip file

Pre-releases

- Python 3.10.0a2 - Nov. 3, 2020
  - Download Windows help file
  - Download Windows x86-64 embeddable zip file
  - Download Windows x86-64 executable installer
  - Download Windows x86-64 web-based installer
  - Download Windows x86 embeddable zip file
  - Download Windows x86 executable installer

- After downloading, click on run. It will open the installation window.
- Check the box Add Python 3.9 to PATH and then click on install.



Python 3.9.0 (64-bit) Setup

Install Python 3.9.0 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\maanz\AppData\Local\Programs\Python\Python39

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.9 to PATH

<u>MySQL Database:</u>

MySQL is an RDBMS (Relational Database Management System) based on the SQL (Structured Query Language), which is the popular language for accessing and managing the records in the database. MySQL is open-source and free software under the GNU license. It is supported by Oracle Company. The data in a MySQL database are stored in tables. A table is a collection of related data, and it consists of columns and rows. Some of the features of MySQL are listed below:

- It is a database system used on the web
- It runs on a server
- It is ideal for both small and large applications

- It is very fast, reliable, and easy to use
- It uses standard SQL
- It is free to download and use

Python Flask:

Flask is a lightweight Python web framework that provides useful tools and features for creating web applications in the Python Language. It gives developers flexibility and is an accessible framework for new developers because you can build a web application quickly using only a single Python file.

HTML:

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

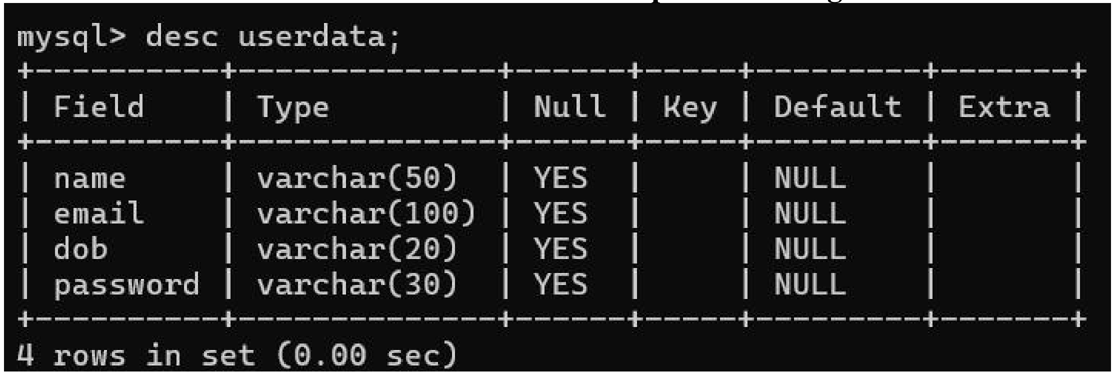
5.Designing:

5.1: Data Base Design

The database consists of three tables namely user data, admin details, user complaint, and each of their structure is given below:

5.1.1 Userdata table:

It consists of all the details that the user should provide to register

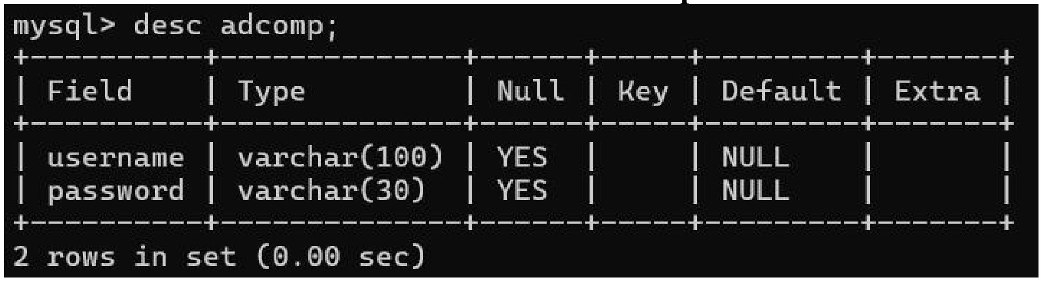


```
mysql> desc userdata;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| name     | varchar(50)  | YES  |     | NULL    |       |
| email    | varchar(100) | YES  |     | NULL    |       |
| dob      | varchar(20)  | YES  |     | NULL    |       |
| password | varchar(30)  | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

5.1.1 Userdata table

5.1.2 adcomp table:

It consists of all the details that the admin should provide to access of database.

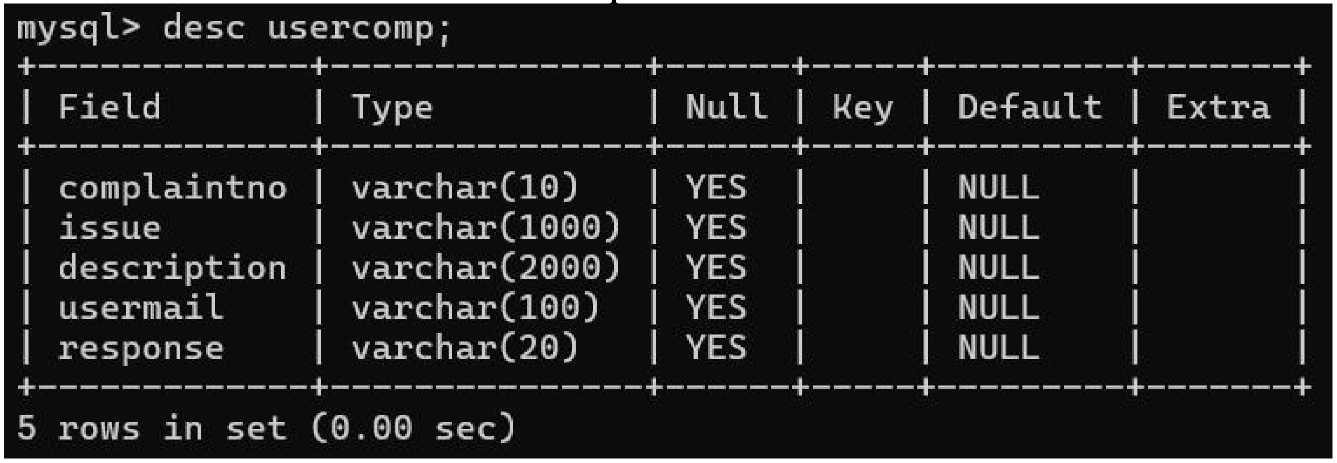


```
mysql> desc adcomp;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| username | varchar(100) | YES  |     | NULL    |       |
| password | varchar(30)  | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

5.1.2 adcomp table

5.1.3 usercomp table:

It consists of all the details that the user complaints in the database.



```
mysql> desc usercomp;
+-------------+---------------+------+-----+---------+-------+
| Field       | Type          | Null | Key | Default | Extra |
+-------------+---------------+------+-----+---------+-------+
| complaintno | varchar(10)   | YES  |     | NULL    |       |
| issue       | varchar(1000) | YES  |     | NULL    |       |
| description | varchar(2000) | YES  |     | NULL    |       |
| usermail    | varchar(100)  | YES  |     | NULL    |       |
| response    | varchar(20)   | YES  |     | NULL    |       |
+-------------+---------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

5.1.3 usercomp table

5.2: Coding:

5.2.1: Importing the required modules and establishing connection with database.

```
from flask import Flask, request, redirect,render_template,session,url_for,flash
import mysql.connector
import random
import math,os
from key import secret_key,salt
from itsdangerous import URLSafeTimedSerializer
from stoken import token
from cmail import sendmail


app = Flask(__name__)
app.secret_key =secret_key


mydb=mysql.connector.connect(host="localhost",user="root",password="vamsi",db="ocp")
cursor=mydb.cursor()
```

5.2.2: Code for the routing of Home page and User-Login.

```
 @app.route('/')
def h1():
    return redirect(url_for('home'))
@app.route('/home')
def home():
    return render_template('Home.html')

@app.route('/userlogin',methods=['GET','POST'])
def userlogin():
    if request.method == 'POST':
        usermail=request.form['email']
        up=request.form['password1']
        cursor=mydb.cursor(buffered=True)
        cursor.execute("select count(*) from userdata where email=%s and 
password=%s",(usermail,up))
        record=cursor.fetchone()[0]
        print(record)
        if record==1:
            session['loggedin']=True
            session['usermail']=usermail
            cursor.close()
            return redirect(url_for('userview'))
        else:
            flash('Invalid Username/Password')
            return render_template('login.html')

    return render_template('login.html')
```

5.2.3: Code for the User Registration.

```
@app.route('/userregistration',methods=['GET','POST'])
def userregistration():
    if request.method == 'POST':
        cursor=mydb.cursor(buffered=True)
        uname=request.form['name']
        umail=request.form['email']
        udob=request.form['date']
        upass=request.form['password2']
        cursor.execute('select count(*) from userdata where name=%s',[uname])
        count=cursor.fetchone()[0]
        cursor.execute('select count(*) from userdata where email=%s',[umail])
        count1=cursor.fetchone()[0]
        cursor.close()
        if count==1:
            flash('username already in use')
            return render_template('registration.html')
        elif count1==1:
            flash('Email already in use')
            return render_template('registration.html')
        data={'username':uname,'password':upass,'email':umail,'dob':udob}
        subject='Email Confirmation'
        body=f"Thanks for signing up\n\nfollow this link for further steps-
{url_for('confirm',token=token(data),_external=True)}"
        sendmail(to=umail,subject=subject,body=body)
        flash('Confirmation link sent to mail')
        return redirect(url_for('userlogin'))

    return render_template('registration.html')
```

5.2.4: Confirmation to register user details into the database and User-view.

```
@app.route('/confirm/<token>')
def confirm(token):
    try:
        serializer=URLSafeTimedSerializer(secret_key)
        data=serializer.loads(token,salt=salt,max_age=180)
    except Exception as e:
        #print(e)
        return 'Link Expired register again'
    else:
        cursor=mydb.cursor(buffered=True)
        username=data['username']
        cursor.execute('select count(*) from userdata where name=%s',[username])
        count=cursor.fetchone()[0]
        if count==1:
            cursor.close()
            flash('You are already registerterd!')
            return redirect(url_for('login'))
        else:
            cursor.execute('insert into userdata 
values(%s,%s,%s,%s)',[data['username'],data['email'],data['dob'],data['password']])
            mydb.commit()
            cursor.close()
            flash('Details registered!')
            return redirect(url_for('userlogin'))

@app.route('/userview',methods=['GET','POST'])
def userview():
    usermail=[session.get('usermail')]
    cursor=mydb.cursor(buffered=True)
    cursor.execute("select complaintno,issue,description,response from usercomp where 
usermail=%s",(usermail))
    record=cursor.fetchall()
    if record:
        cursor.close()
        return render_template('user_view.html',value=record)
    return render_template('user_view.html')
```

5.2.5: Code for the registration of Complaint from the User.

```
cursor=mydb.cursor(buffered=True)
@app.route('/usercomplaint',methods=['GET','POST'])
def usercomplaint():
    if request.method == 'POST':
        usub=request.form['subject']
        ubody=request.form['body']
        unmail=session.get('usermail')
        digits = "0123456789"
        a='pending'
        OTP=''
            #now we will use math module along with module to generate a customized
            #6 digit otp
        for i in range(6):
            OTP = OTP + digits[math.floor(random.random()*10)]

        data={'OTP':OTP,'subject':usub,'body':ubody,'usermail':unmail,'response':a}
        cursor=mydb.cursor(buffered=True)
        cursor.execute("select email,name from userdata where 
email=%s",([data['usermail']]))
        emailrec=cursor.fetchone()
        if emailrec:
            result=cursor.execute("insert into usercomp 
values(%s,%s,%s,%s,%s)",[data['OTP'],data['subject'],data['body'],data['usermail'],dat
a['response']])
            mydb.commit()
            cursor.close()
            return redirect(url_for('userview'))
        else:
            cursor.close()
            return render_template('complaint.html')

    return render_template('complaint.html')
```

5.2.6: Code for the Admin Login and Admin-view.

```
@app.route('/adminlogin',methods=['GET','POST'])
def adminlogin():
    if request.method == 'POST':
        un=request.form['email']
        up=request.form['password1']
        cursor=mydb.cursor(buffered=True)
        cursor.execute("select count(*) from adcomp where username=%s and 
password=%s",(un,up))
        record=cursor.fetchone()[0]
        if record:
            session['loggedin']=True
            session['username']=un
            mydb.commit()
            cursor.close()
            return redirect(url_for('adminview'))

        else:
            flash('Invalid Username/Password')
            return render_template('admin_login.html')
    return render_template('admin_login.html')

@app.route('/adminview',methods=['GET','POST'])
def adminview():
    cursor=mydb.cursor(buffered=True)
    cursor.execute("select * from usercomp")
    record=cursor.fetchall()
    #ab=cursor.execute("update usercomp set response=:s" ,(aa))
    mydb.commit()
    if record:
        cursor.close()
        return render_template('admin_view.html',value=record)


    return render_template('admin_view.html')
```

5.2.7: Code For the Update the Complaint Status in the Admin-view.

```
@app.route('/updatestatus',methods=['GET','POST'])
def updatestatus():
    if request.method == 'POST':
        aa=request.form['compno']
        ab=request.form['status']
        cno=[aa]
        cursor=mydb.cursor(buffered=True)
        result=cursor.execute("update usercomp set response=%s where 
complaintno=%s",(ab,aa))
        ac=cursor.execute("select usermail,issue from usercomp where 
complaintno=%s",(cno))
        ad=cursor.fetchone()
        if ab=='Solved':
            subject='Complaint Status'
            body=f"Your Registered Complaint no:{aa} regarding the problem:{ad[1]} is 
Solved.Thanks For Contacting The Online Complaints Portal."
            sendmail(to=ad[0],subject=subject,body=body)
            return redirect(url_for('adminview'))
        elif ab=='In Progress':
            subject='Complaint Status'
            body=f"Your Registered Complaint no:{aa} regarding the problem:{ad[1]} is 
Updated to In Progress.Thanks For Contacting The Online Complaints Portal."
            sendmail(to=ad[0],subject=subject,body=body)
            return redirect(url_for('adminview'))


        a=mydb.commit()


    return render_template('update_status.html')
```

5.2.8: Code for the Admin-logout and User-logout.

```
@app.route('/adminlogout')
def adminlogout():
    if session.get('user'):
        session.pop('user')
        flash('Successfully logged out')
        return redirect(url_for('home'))
    else:
        return redirect(url_for('home'))
```

```
@app.route('/logout')
def logout():
    if session.get('user'):
        session.pop('user')
        flash('Successfully logged out')
        return redirect(url_for('userlogin'))
    else:
        return redirect(url_for('userlogin'))

if __name__ == '__main__':
    app.run(debug=True)
```

5.3: Frontend Development:

5.3.1: Code for Home Page

CSS:

```
body{
    background-image:linear-gradient(15deg, #13547a 0%, #80d0c7 100%);
    }
    h1 {
        margin-top: 50px;
        font-size:50px;
        text-align:center;
        background: -webkit-linear-gradient(#989696, #f70909);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:800px;
        height:300px;
        border-radius: 5px;
        margin: 100 auto;
        font-size:30px;
        text-align: center;
        padding:50px;
    }
    .login {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family: Arial, Helvetica, sans-serif;
    }

    #login_position {
        position: fixed;
        bottom:150px;
        right: 350px;
    }
    .register {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family: Arial, Helvetica, sans-serif;
    }

    #register_position {
        position: fixed;
        bottom:150px;
        right: 550px;
    }
    .login:hover {
        background-color: brown;
        color: white;
    }
    .register:hover {
        background-color: brown;
        color: white;
    }

    .adminlogin {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family: Arial, Helvetica, sans-serif;
    }

    #adminlogin_position {
        position: fixed;
        bottom:150px;
        left: 350px;
    }
    .adminlogin:hover {
        background-color: brown;
```

```
        color: white;
    }
```

HTML:

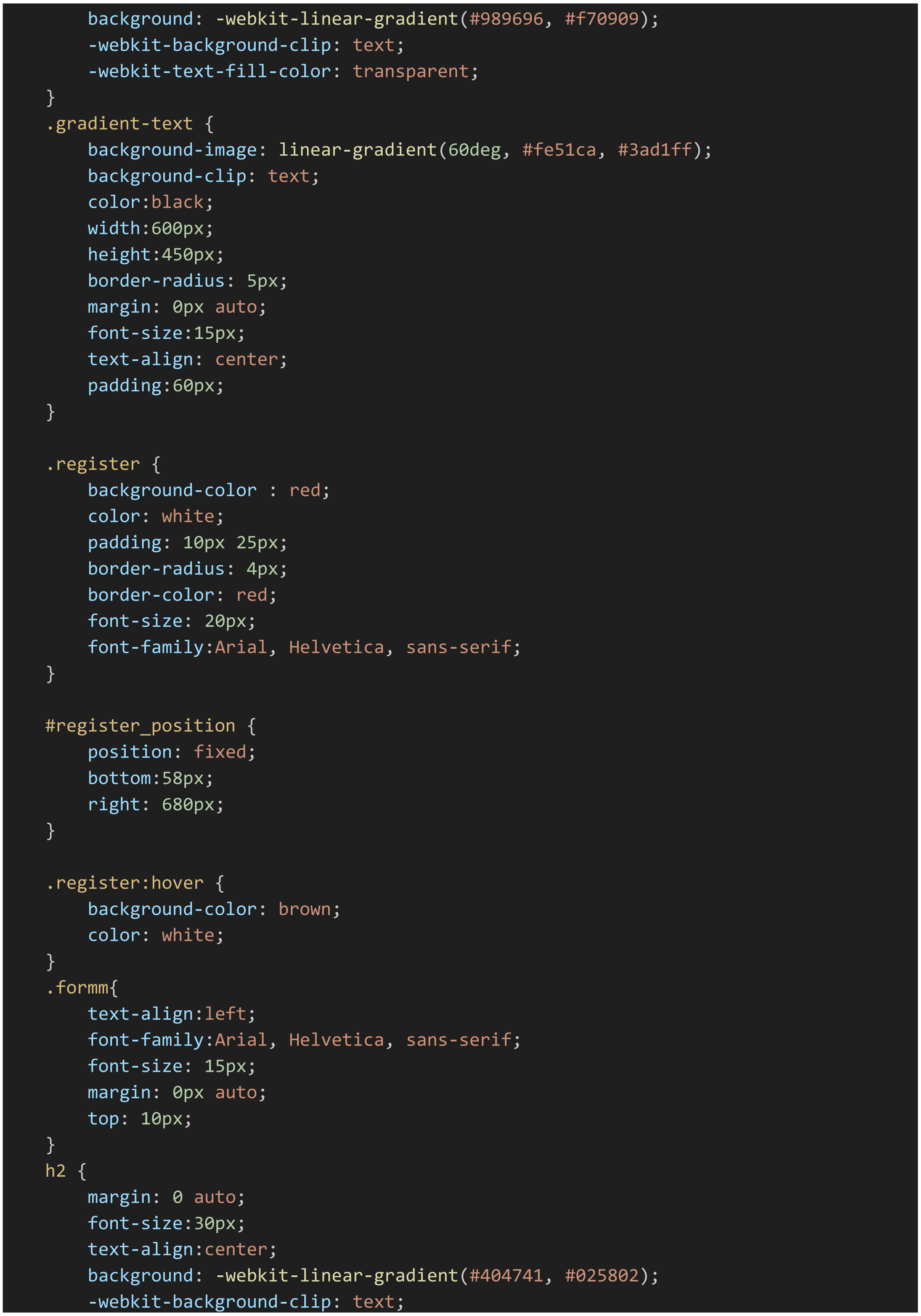
```
<html>
    <head>
        <link rel="<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
        crossorigin="anonymous">
        <link rel="stylesheet" href="{{url_for('static',filename='Home.css')}}">
    </head>
    <body>
        <h1>ONLINE COMPLAINTS PORTAL</h1>
        <div class="gradient-text">
        <i>Welcome to Online Complaints Portal.<br><br>

        Here you can give complaints regarding your issues.</i><br><br>
        <div id="login_position">
            <a href="{{url_for('userlogin')}}"><button class="login" id="login" >User
LOGIN</button></a>
        </div>
        <div id="adminlogin_position">
            <a href="{{url_for('adminlogin')}}"><button class="adminlogin"
id="login" >Admin LOGIN</button></a>
        </div>
        <div id="register_position">
            <a href="{{url_for('userregistration')}}"><button class="register"
id="register" >REGISTER</button></a>
        </div>
        </div>
    </body>
</html>
```

5.3.2: Code for the Registration Page

CSS

```
body{
    background-image:linear-gradient(15deg, #13547a 0%, #80d0c7 100%);
    }
    h1 {
        margin-top: 50px;
        font-size:50px;
        text-align:center;
```



```css
        background: -webkit-linear-gradient(#989696, #f70909);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:600px;
        height:450px;
        border-radius: 5px;
        margin: 0px auto;
        font-size:15px;
        text-align: center;
        padding:60px;
    }

    .register {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family:Arial, Helvetica, sans-serif;
    }

    #register_position {
        position: fixed;
        bottom:58px;
        right: 680px;
    }

    .register:hover {
        background-color: brown;
        color: white;
    }
    .formm{
        text-align:left;
        font-family:Arial, Helvetica, sans-serif;
        font-size: 15px;
        margin: 0px auto;
        top: 10px;
    }
    h2 {
        margin: 0 auto;
        font-size:30px;
        text-align:center;
        background: -webkit-linear-gradient(#404741, #025802);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text2 {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:500px;
        height:400px;
        border-radius: 5px;
        margin: 5px auto;
        font-size:15px;
        text-align: center;
        padding:35px;
    }
```

HTML

```
<html>
    <head>
        <link rel="<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
        crossorigin="anonymous">
        <link rel="stylesheet"
href="{{url_for('static',filename='registration.css')}}">
    </head>
    {% with messages = get_flashed_messages() %}
        {% if messages %}
            {% for message in messages %}
                <center><b>{{ message }}</b></center>
            {% endfor %}
        {% endif %}
    {% endwith %}
    <body>
        <h1>ONLINE COMPLAINTS PORTAL</h1>
        <div class="gradient-text">
            <h2>REGISTRATION</h2>
            <div class="gradient-text2">
            <form class="formm" method="post">


                <label>Name:</label>
                <p><input type="text" name="name"/></p>
                <label>Email:</label>
                <p><input type="email" name="email"/></p>
                <label>Date Of Birth:</label>
                <p><input type="date" name="date"/></p>
                <label>Gender:</label>
                <p><input type="radio" name="gender" value="Male">Male
                <input type="radio" name="gender" value="Female">Female</p>
                <label> PassWord:</label>
                <p><input type="password" name="password1" /></p>
                <label>Confirm PassWord:</label>
                <p><input type="password" name="password2"  /></p>
                <div id="register_position">
                    <button class="register" id="register" >Register</button></a>
                </div>
            </form>
            </div>
        </div>
    </body>
</html>
```

5.3.3: Code for the User Login

CSS

```
body{
    background-image:linear-gradient(15deg, #13547a 0%, #80d0c7 100%);
    }
    h1 {
        margin-top: 50px;
        font-size:50px;
        text-align:center;
        background: -webkit-linear-gradient(#989696, #f70909);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:600px;
        height:410px;
        border-radius: 5px;
        margin: 0px auto;
        font-size:15px;
        text-align: center;
        padding:60px;
    }
    .gradient-text2 {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:200px;
        height:150px;
        border-radius: 5px;
        margin: 5px auto;
        font-size:15px;
        text-align: center;
        padding:60px;
    }

    .login {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family: Arial, Helvetica, sans-serif;
    }

    #login_position {
        position: fixed;
        bottom:90px;
        right: 700px;
    }

    .login:hover {
        background-color: brown;
        color: white;
    }
    .formm{
        text-align:left;
        font-family: Arial, Helvetica, sans-serif;
        font-size: 15px;
        margin: 0px auto;
        top: 10px;
    }
    h2 {
        margin: 0 auto;
        font-size:30px;
        text-align:center;
        background: -webkit-linear-gradient(#404741, #025802);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
```

HTML

```
<html>
    <head>
```

```
        <link rel="<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
        crossorigin="anonymous">
        <link rel="stylesheet" href="{{url_for('static',filename='login.css')}}">
    </head>
    {% with messages = get_flashed_messages() %}
        {% if messages %}
            {% for message in messages %}
                <center><b>{{ message }}</b></center>
            {% endfor %}
        {% endif %}
    {% endwith %}
    <body>
        <h1>ONLINE COMPLAINTS PORTAL</h1>
        <div class="gradient-text">
            <h2>LOGIN</h2>
            <div class="gradient-text2">
            <form class="formm" method="post" >
                <label>Email:</label>
                <p><input type="email" name="email"/></p>
                <label>PassWord:</label>
                <p><input type="password" name="password1" /></p>
                <div id="register_position">
                    <a href="{{url_for('usercomplaint')}}"><button class="login"
id="login" >Login</button></a>
                </div>
            </form>
            </div>
        </div>
    </body>
</html>
```

5.3.4: Code for the admin Login

CSS

```
body{
    background-image:linear-gradient(15deg, #13547a 0%, #80d0c7 100%);
    }
    h1 {
        margin-top: 50px;
        font-size:50px;
        text-align:center;
        background: -webkit-linear-gradient(#989696, #f70909);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:600px;
        height:410px;
        border-radius: 5px;
        margin: 0px auto;
        font-size:15px;
        text-align: center;
        padding:60px;
    }
    .gradient-text2 {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:200px;
        height:150px;
        border-radius: 5px;
        margin: 30px auto;
        font-size:15px;
        text-align: center;
        padding:60px;
    }

    .login {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family:Arial, Helvetica, sans-serif;
    }

    #login_position {
        position: fixed;
        bottom:90px;
        right: 700px;
    }

    .login:hover {
        background-color: brown;
        color: white;
    }
    .formm{
        text-align:left;
        font-family: cursive;
        font-size: 15px;
        margin: 0px auto;
        top: 10px;
    }
    h2 {
        margin: 0 auto;
        font-size:30px;
        text-align:center;
        background: -webkit-linear-gradient(#404741, #025802);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
```

HTML

```
<html>
    <head>
        <link rel="<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
        crossorigin="anonymous">
        <link rel="stylesheet"
href="{{url_for('static',filename='admin_login.css')}}">
    </head>
    {% with messages = get_flashed_messages() %}
        {% if messages %}
            {% for message in messages %}
                <center><b>{{ message }}</b></center>
            {% endfor %}
        {% endif %}
    {% endwith %}
    <body>
        <h1>ONLINE COMPLAINTS PORTAL</h1>
        <div class="gradient-text">
            <h2> ADMIN LOGIN</h2>
            <div class="gradient-text2">
            <form class="formm" method="post" >
                <label>Email:</label>
                <p><input type="email" name="email"/></p>
                <label>PassWord:</label>
                <p><input type="password" name="password1" /></p>
                <div id="register_position">
                    <button class="login" id="login" >Login</button>
                </div>
            </form>
            </div>
        </div>
    </body>
</html>
```

5.3.5: Code for the User-view

CSS

```
body{
    background-image:linear-gradient(15deg, #13547a 0%, #80d0c7 100%);
    }
    h1 {
        margin-top: 50px;
        font-size:50px;
        text-align:center;
        background: -webkit-linear-gradient(#989696, #f70909);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:800px;
        height:auto;
        border-radius: 5px;
        margin: 100 auto;
        font-size:30px;
        text-align: center;
        padding:50px;
    }
    .complaint {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family:Arial, Helvetica, sans-serif;
    }

    #complaint_position {
        position: fixed;
        bottom:50px;
        right: 100px;
    }
    .complaint:hover {
        background-color: brown;
        color: white;
    }
    .logout {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family:Arial, Helvetica, sans-serif;
    }

    #Logout_position {
        position: fixed;
        bottom:50px;
        left: 100px;
    }
    .logout:hover {
        background-color: brown;
        color: white;
    }
```

HTML

```
<html>
    <head>
        <link rel="<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
        crossorigin="anonymous">
        <link rel="stylesheet" href="{{url_for('static',filename='user_view.css')}}">
    </head>
    <body>
        <h1>ONLINE COMPLAINTS PORTAL</h1>
        <div class="gradient-text">
            <div id="complaint_position">
                <a href="{{url_for('usercomplaint')}}"><button class="complaint"
id="complaint" >New Complaint</button></a>
            </div>
            <div id="Logout_position">
                <a href="{{url_for('logout')}}"><button class="logout"
id="logout" >Logout</button></a>
            </div>
            <table border="1" align="center" padding="5px" cellspacing="8px"
cellpadding="10px">
                <thead>
                <tr>
                    <th>Complaint ID</th>
```

```html
                    <th>Issue</th>
                    <th>Description</th>
                    <th>Response</th>
                </tr>
                </thead>
                <tbody>
                    {% for row in value %}
                        <tr>
                            <td>{{row[0]}}</td>
                            <td>{{row[1]}}</td>
                            <td>{{row[2]}}</td>
                            <td>{{row[3]}}</td>
                        </tr>

                    {% endfor %}

                </tbody>
            </table>
        </div>
    </body>
</html>
```

5.3.6: Code for the Admin-view

CSS

```css
body{
    background-image:linear-gradient(15deg, #13547a 0%, #80d0c7 100%);
    }
    h1 {
        margin-top: 50px;
        font-size:50px;
        text-align:center;
        background: -webkit-linear-gradient(#989696, #f70909);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:800px;
        height:auto;
        border-radius: 5px;
        margin: 100 auto;
        font-size:30px;
        text-align: center;
        padding:50px;
    }
    .complaint {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family: Arial, Helvetica, sans-serif;
    }

    #complaint_position {
        position: fixed;
        bottom:50px;
        right: 85px;
    }
    .complaint:hover {
        background-color: brown;
        color: white;
    }
    .logout {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family: Arial, Helvetica, sans-serif;
    }

    #Logout_position {
        position: fixed;
        bottom:50px;
        left: 100px;
    }
    .logout:hover {
        background-color: brown;
        color: white;
    }
```

HTML

```
<html>
    <head>
        <link rel="<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
        crossorigin="anonymous">
        <link rel="stylesheet" href="{{url_for('static',filename='admin_view.css')}}">
    </head>
    <body>
        <h1>ONLINE COMPLAINTS PORTAL</h1>
        <div class="gradient-text">
            <div id="Logout_position">
                <a href="{{url_for('adminlogout')}}"><button class="logout"
id="logout" >Logout</button></a>
            </div>
            <form method="post">
            <table border="1" align="center" padding="5px" cellspacing="8px"
cellpadding="10px">
                <thead>
                <tr>
                    <th>Complaint ID</th>
                    <th>Issue</th>
                    <th>Description</th>
                    <th>Username</th>
                    <th>Response</th>
                </tr>
                </thead>
                <tbody>
                    {% for row in value %}
                        <tr>
                            <td>{{row[0]}}</td>
                            <td>{{row[1]}}</td>
                            <td>{{row[2]}}</td>
                            <td>{{row[3]}}</td>
                            <td>{{row[4]}}</td>
                        </tr>

                    {% endfor %}

                </tbody>
            </table>
            </form>
        </div>
        <div id="complaint_position">
            <a href="{{url_for('updatestatus')}}"><button class="complaint"
id="complaint" >Update Status</button></a>
        </div>
    </body>
</html>
```

5.3.7: Code for the User-Complaint

CSS

```css
body{
    background-image:linear-gradient(15deg, #13547a 0%, #80d0c7 100%);
    }
    h1 {
        margin-top: 50px;
        font-size:50px;
        text-align:center;
        background: -webkit-linear-gradient(#989696, #f70909);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:600px;
        height:400px;
        border-radius: 5px;
        margin: 0px auto;
        font-size:15px;
        text-align: center;
        padding:90px;
    }

    .sub {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family: Arial, Helvetica, sans-serif;
    }

    #sub_position {
        position: fixed;
        bottom:58px;
        right: 680px;
    }

    .sub:hover {
        background-color: brown;
        color: white;
    }
    .formm{
        text-align:left;
        font-family: Arial, Helvetica, sans-serif;
        font-size: 15px;
        margin: 0px auto;
        top: 10px;
    }
    h2 {
        margin: 0 auto;
        font-size:30px;
        text-align:center;
        background: -webkit-linear-gradient(#404741, #025802);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text2 {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:500px;
        height:auto;
        border-radius: 5px;
        margin: 0px auto;
        font-size:15px;
        text-align: center;
        padding:15px;
    }
```

HTML

```
<html>
    <head>
        <link rel="<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
        crossorigin="anonymous">
        <link rel="stylesheet" href="{{url_for('static',filename='complaint.css')}}">
    </head>
    <body>
        <h1>ONLINE COMPLAINTS PORTAL</h1>
        <div class="gradient-text">
            <h2>NEW COMPLAINT </h2>
            <div class="gradient-text2">
            <form class="formm" method="post" >

                <label>Complaint Subject</label>
                <p><textarea rows="2" cols="50" placeholder="Enter The Subject Of the
Complaint" name="subject"></textarea></p>
                <label>Complaint Description</label>
                <p><textarea rows="15" cols="50" placeholder="Enter The Text"
name="body"></textarea></p>
                <div id="register_position">
                    <button class="sub" id="submit" >Submit</button></a>
                </div>
            </form>
            </div>
        </div>
        <script>
            {% if message %}
            alert('{{ message }}');
            {% endif %}
        </script>
    </body>
</html>
```

5.3.8: Code for the Update-Status

CSS

```
body{
    background-image:linear-gradient(15deg, #13547a 0%, #80d0c7 100%);
    }
    h1 {
        margin-top: 50px;
        font-size:50px;
        text-align:center;
        background: -webkit-linear-gradient(#989696, #f70909);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
    .gradient-text {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:600px;
        height:410px;
        border-radius: 5px;
        margin: 0px auto;
        font-size:15px;
        text-align: center;
        padding:60px;
    }
    .gradient-text2 {
        background-image: linear-gradient(60deg, #fe51ca, #3ad1ff);
        background-clip: text;
        color:black;
        width:200px;
        height:auto;
        border-radius: 5px;
        margin: 30px auto;
        font-size:15px;
        text-align: center;
        padding:60px;
    }

    .login {
        background-color : red;
        color: white;
        padding: 10px 25px;
        border-radius: 4px;
        border-color: red;
        font-size: 20px;
        font-family:Arial, Helvetica, sans-serif;
    }

    #login_position {
        position: fixed;
        bottom:90px;
        right: 700px;
    }

    .login:hover {
        background-color: brown;
        color: white;
    }
    .formm{
        text-align:left;
        font-family:Arial, Helvetica, sans-serif;
        font-size: 15px;
        margin: 0px auto;
        top: 10px;
    }
    h2 {
        margin: 0 auto;
        font-size:30px;
        text-align:center;
        background: -webkit-linear-gradient(#404741, #025802);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
    }
```

HTML

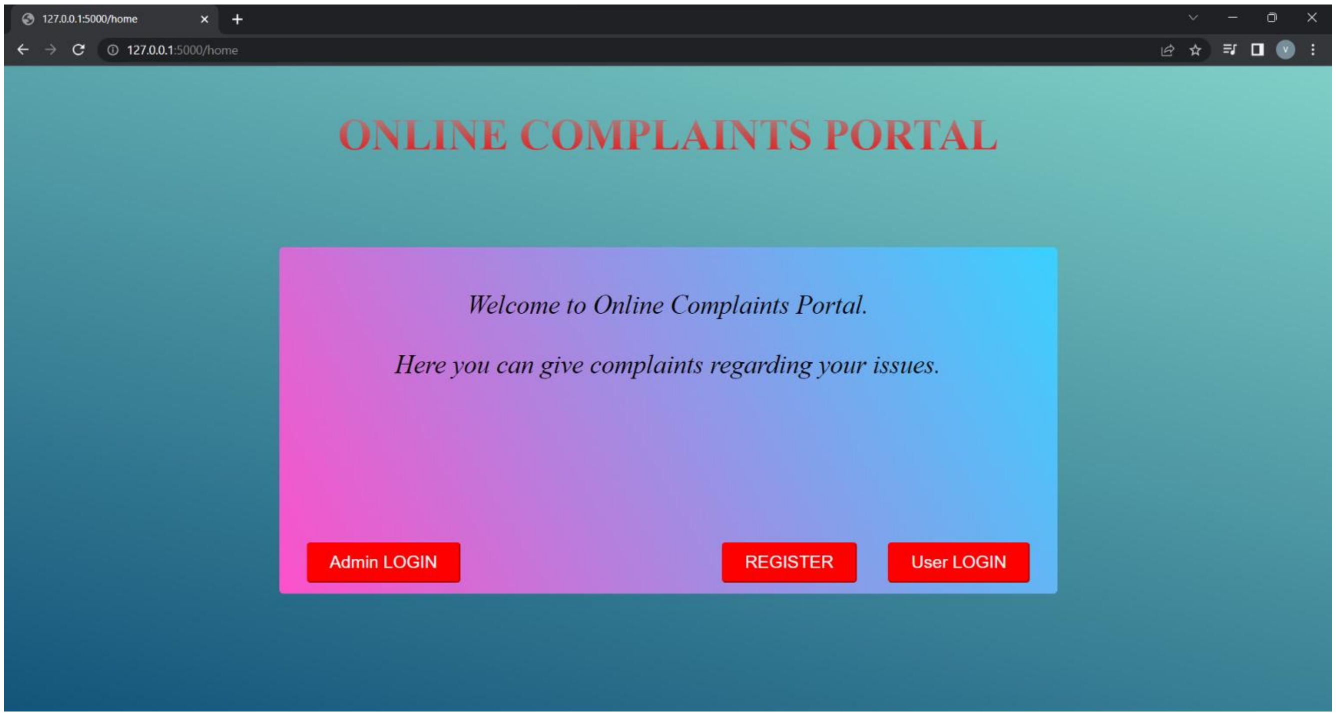
```
<html>
    <head>
```

```
        <link rel="<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
        crossorigin="anonymous">
        <link rel="stylesheet"
href="{{url_for('static',filename='update_status.css')}}">
    </head>
    <body>
        <h1>ONLINE COMPLAINTS PORTAL</h1>
        <div class="gradient-text">
            <h2> UPDATE COMPLAINT STATUS</h2>
            <div class="gradient-text2">
            <form class="formm" method="post" >
                <label>Enter the Complaint Number:</label>
                <p><input type="text" name="compno"/></p>
                <label>Select the Status:</label>
                <p><select
name="status"><option></option><option>Solved</option><option>In
Progress</option><option>Pending</option></select></p>
                <div id="register_position">
                    <button class="login" id="login" type="submit">Update</button>
                </div>
            </form>
            </div>
        </div>
    </body>
</html>
```

5.4: Screen Shots:

User Side:

USER VIEW

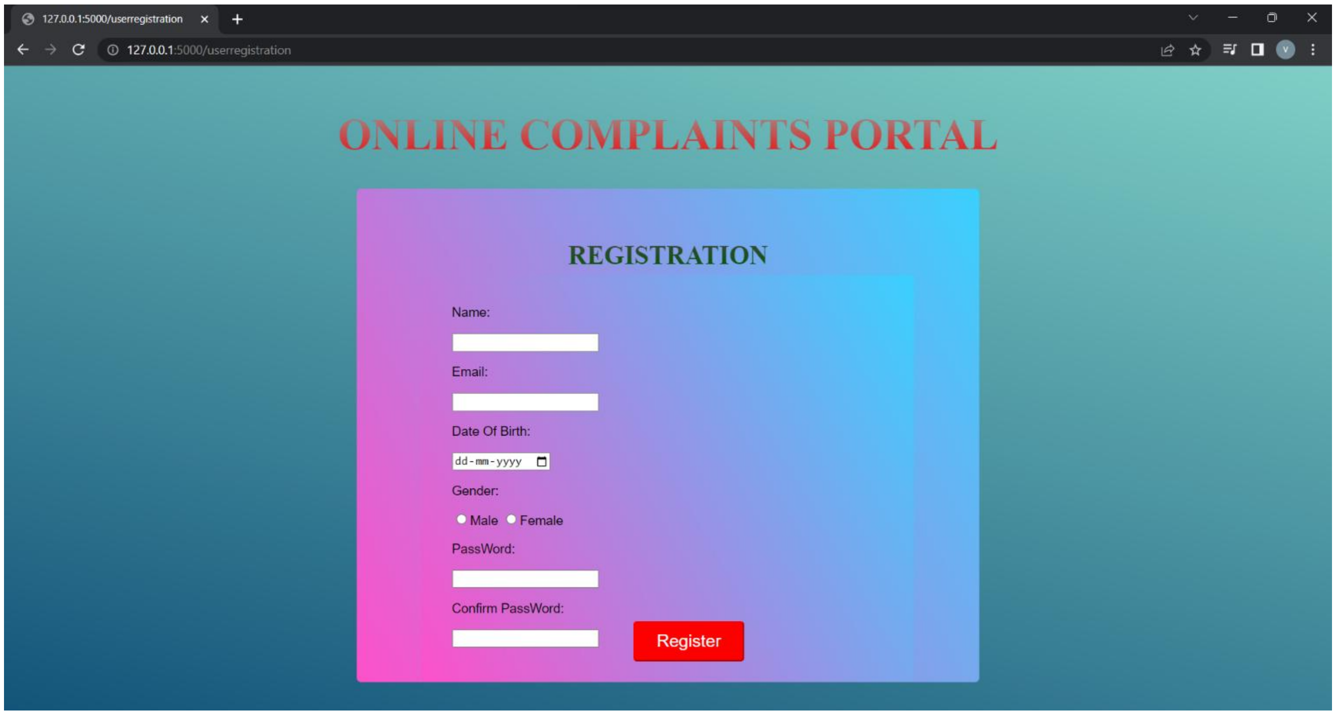


ONLINE COMPLAINTS PORTAL

Welcome to Online Complaints Portal.

Here you can give complaints regarding your issues.

Registration



ONLINE COMPLAINTS PORTAL

REGISTRATION

Name:

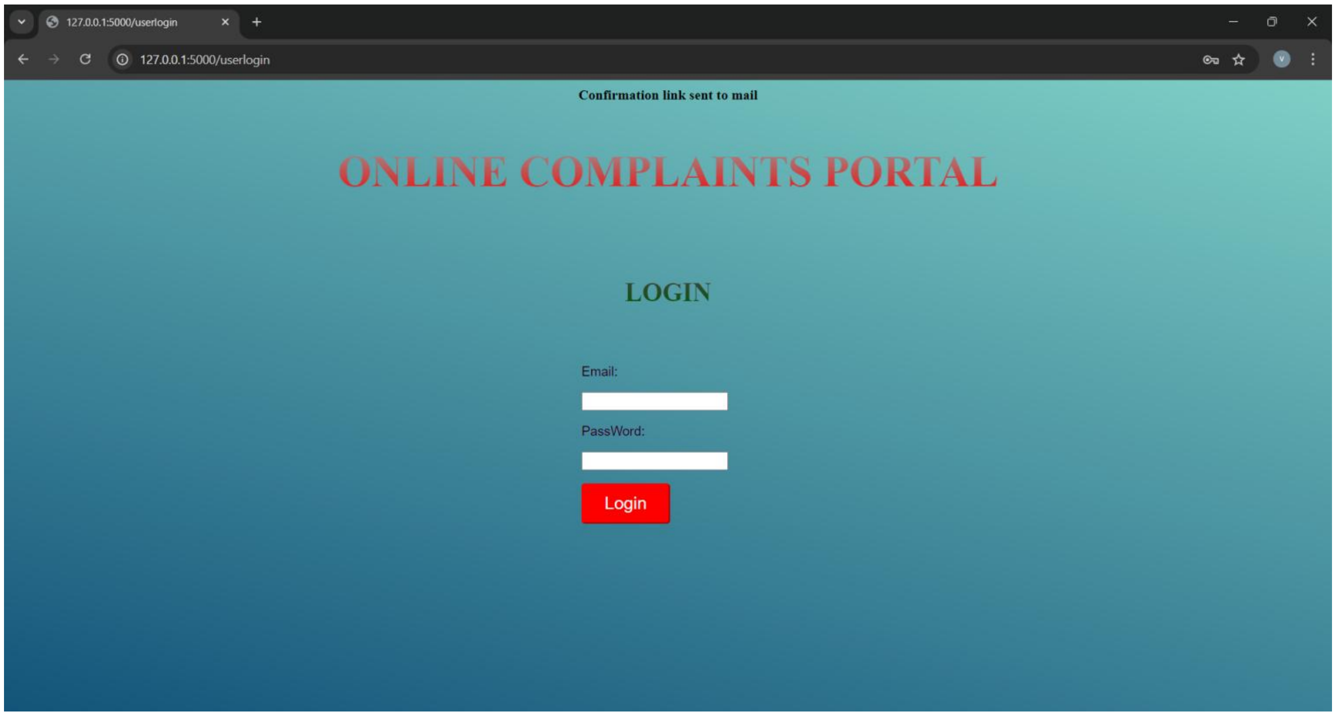
Email:

Date Of Birth: dd-mm-yyyy

Gender: Male Female

PassWord:

Confirm PassWord:



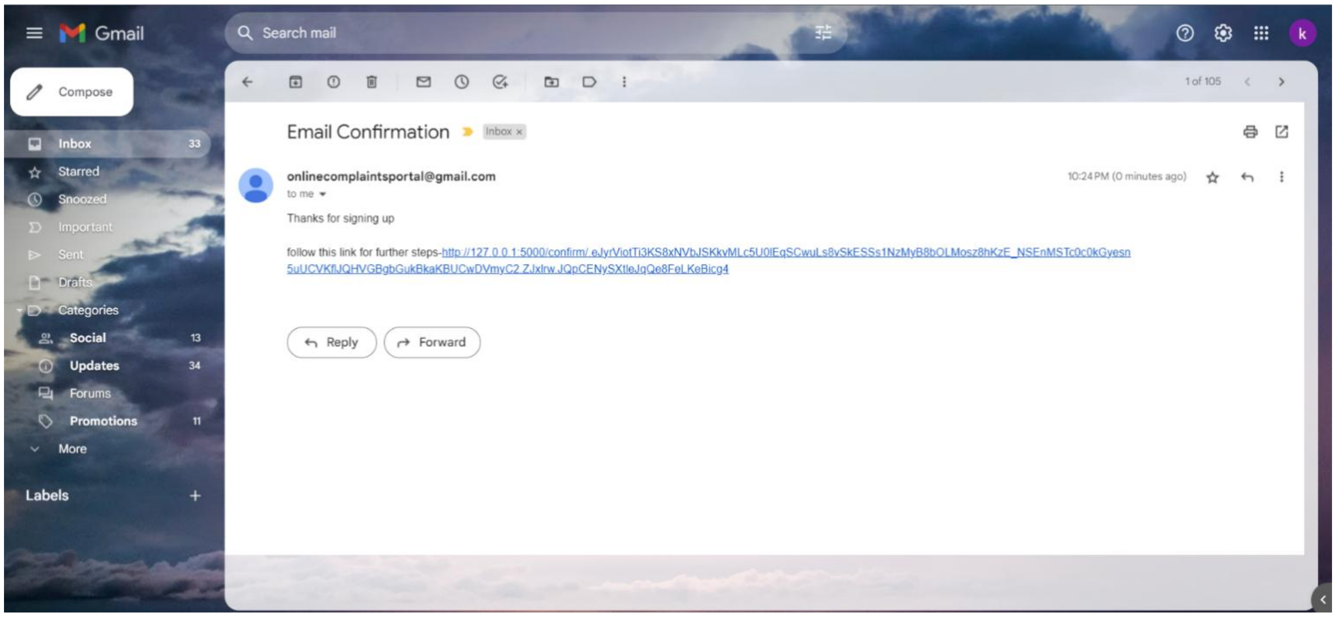
Confirmation link sent to mail

ONLINE COMPLAINTS PORTAL

LOGIN

Email:

PassWord:



Email Confirmation

onlinecomplaintsportal@gmail.com

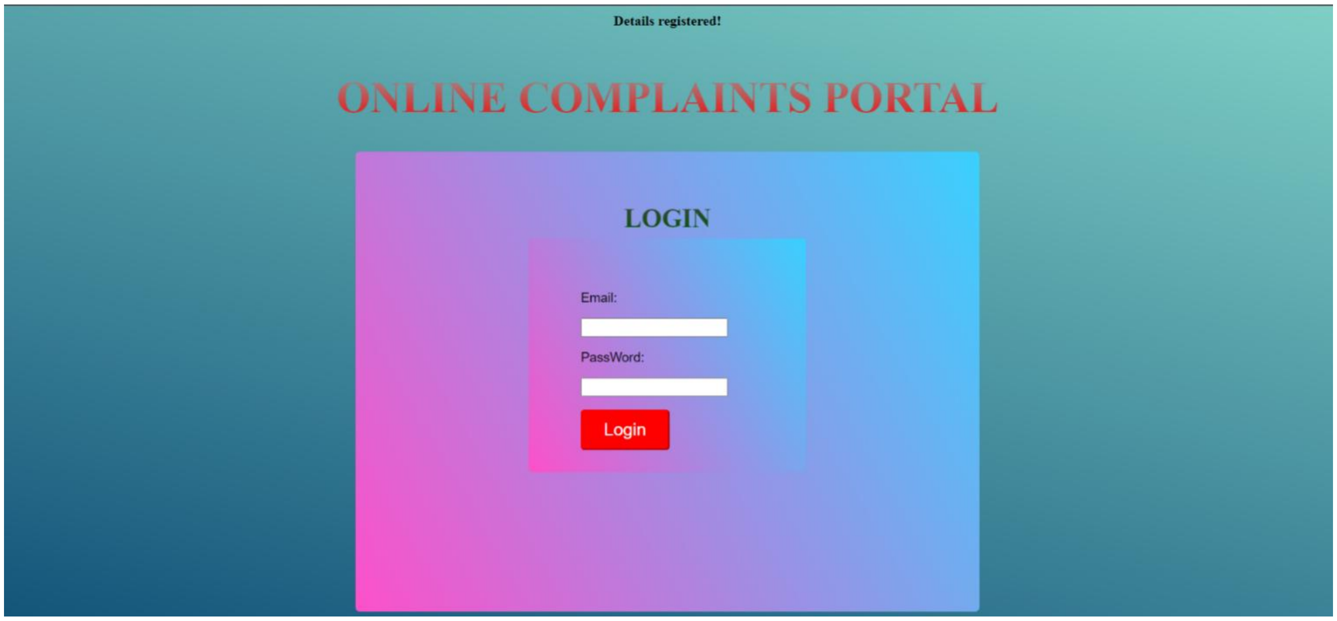
to me

10:24 PM (0 minutes ago)

Thanks for signing up

follow this link for further steps-http://127.0.0.1:5000/confirm/.eJyrViotTi3KS8xNVbJSKkvMLc5U0lEqSCwuLs8vSkESSs1NzMyB8bOLMosz8hKzE_NSEnMSTc0c0kGyesn5uUCVKfIJQHVGBgbGukBkaKBUCwDVmyC2.ZJxlrw.JQpCENySXtleJqQe8FeLKeBicg4

Login


Details registered!

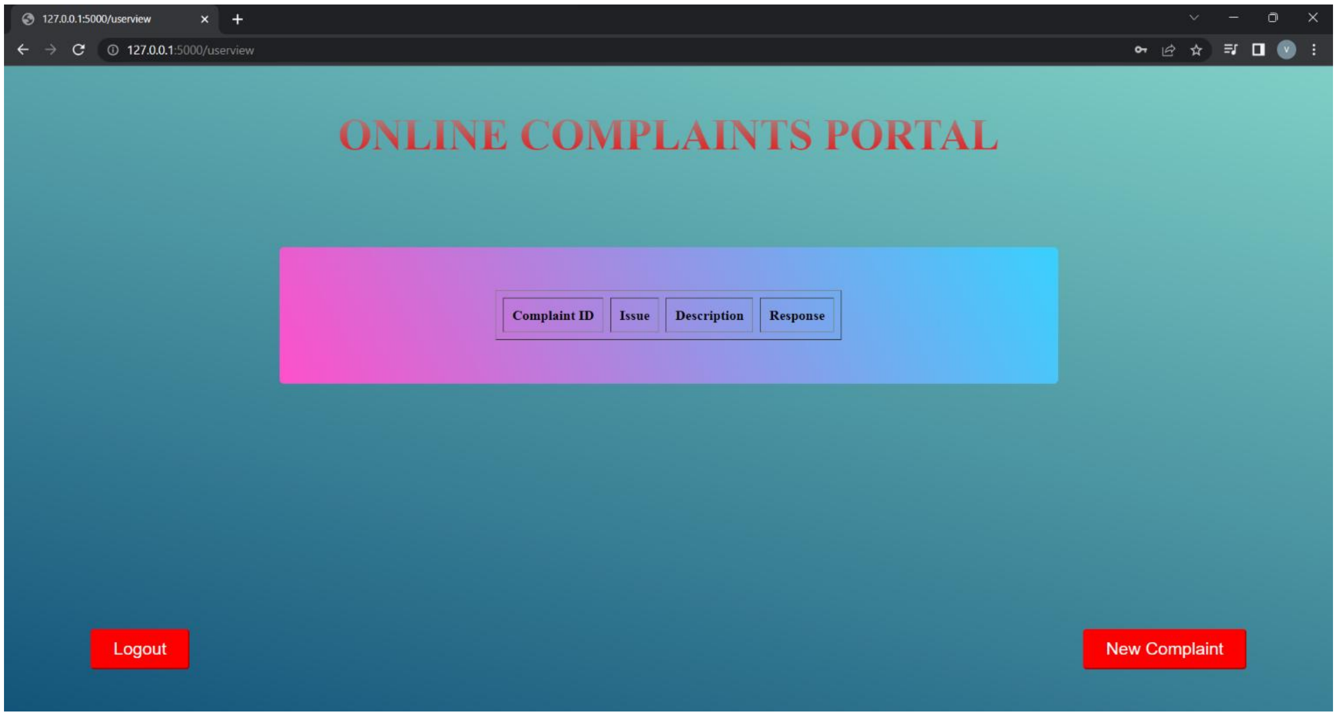
ONLINE COMPLAINTS PORTAL

LOGIN

Email:

PassWord:

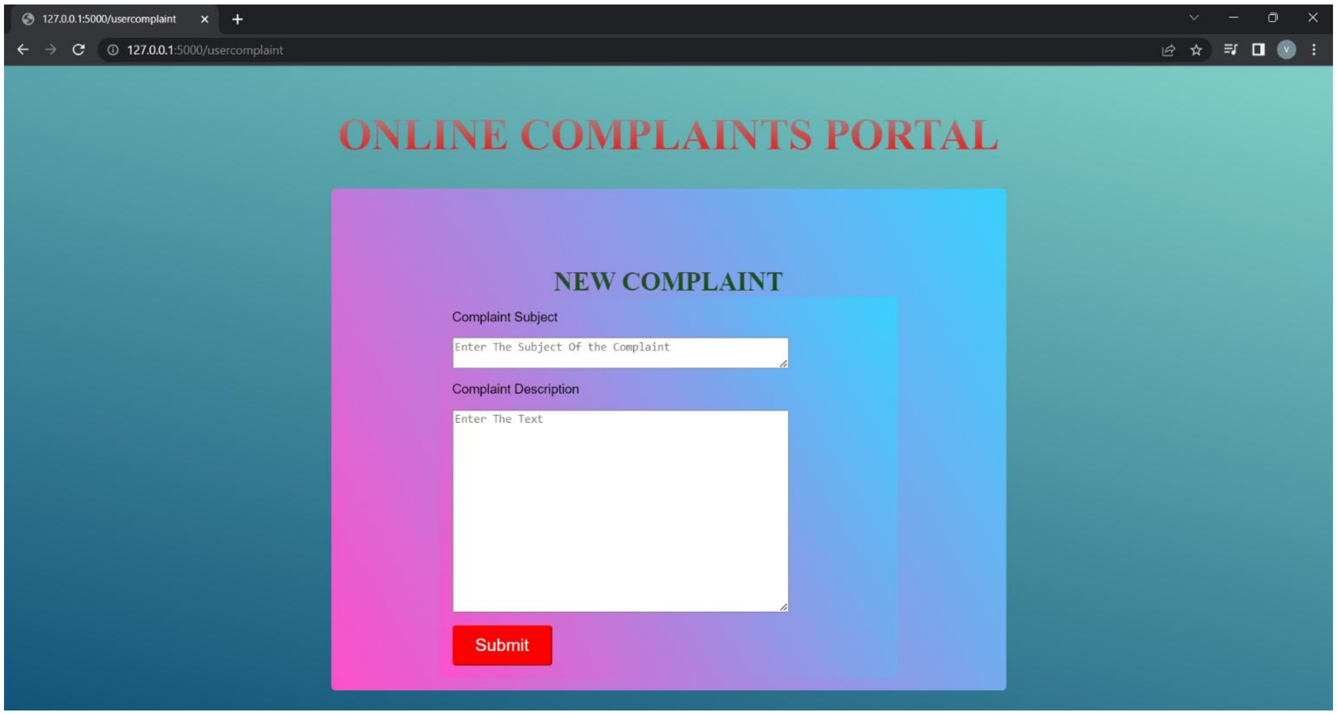
User View Before the Complaint Registered


127.0.0.1:5000/userview

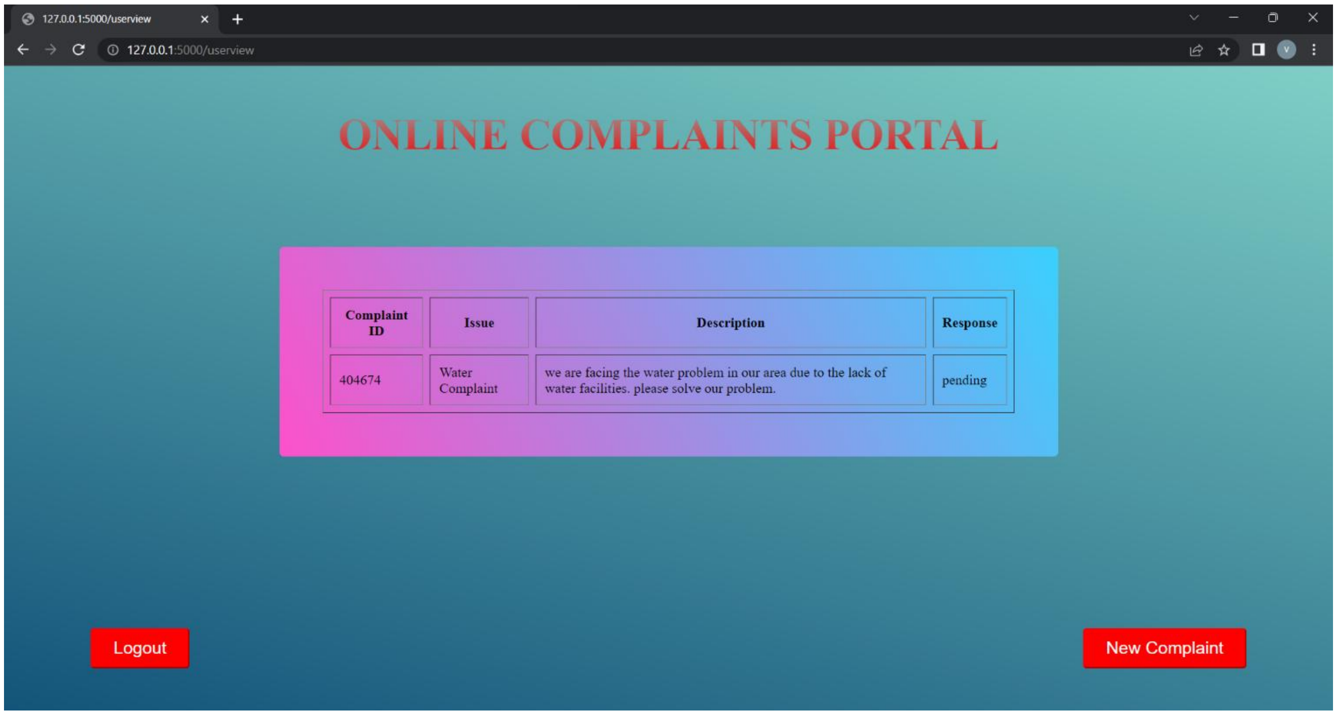
ONLINE COMPLAINTS PORTAL

Complaint ID	Issue	Description	Response

Complaint box



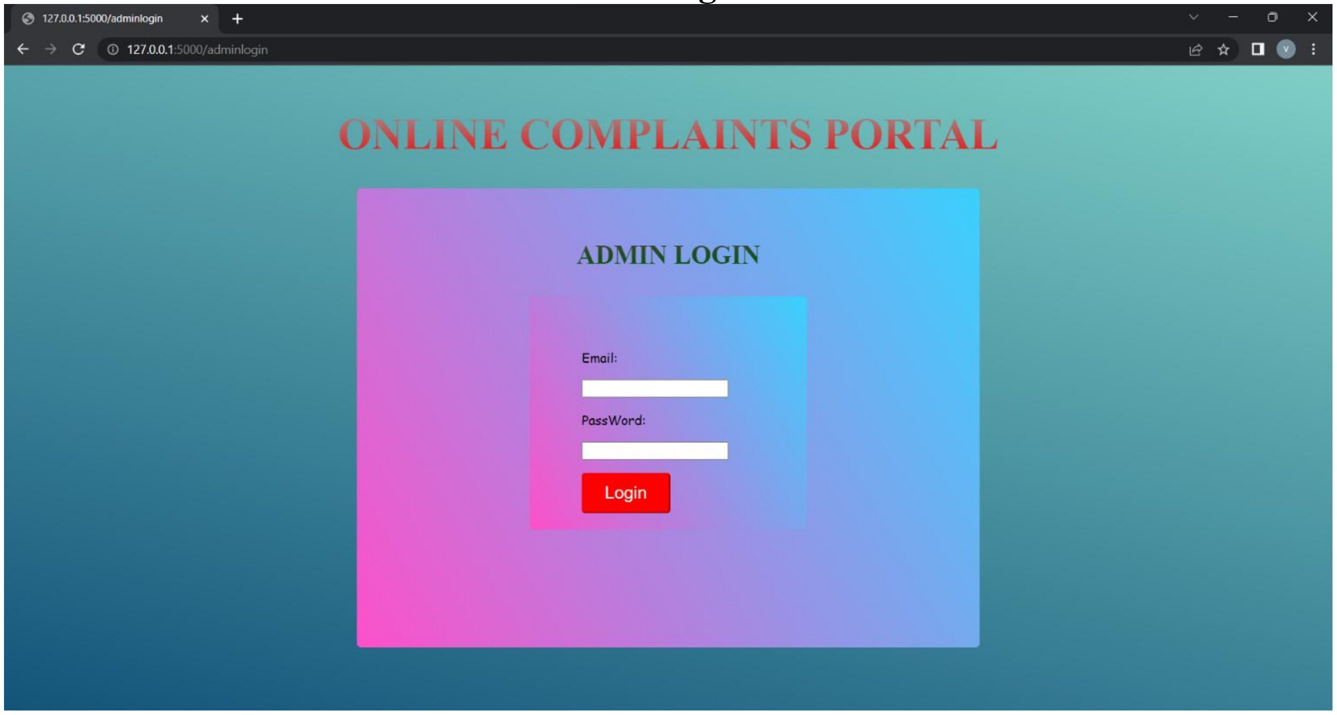
User View after registering the Complaint



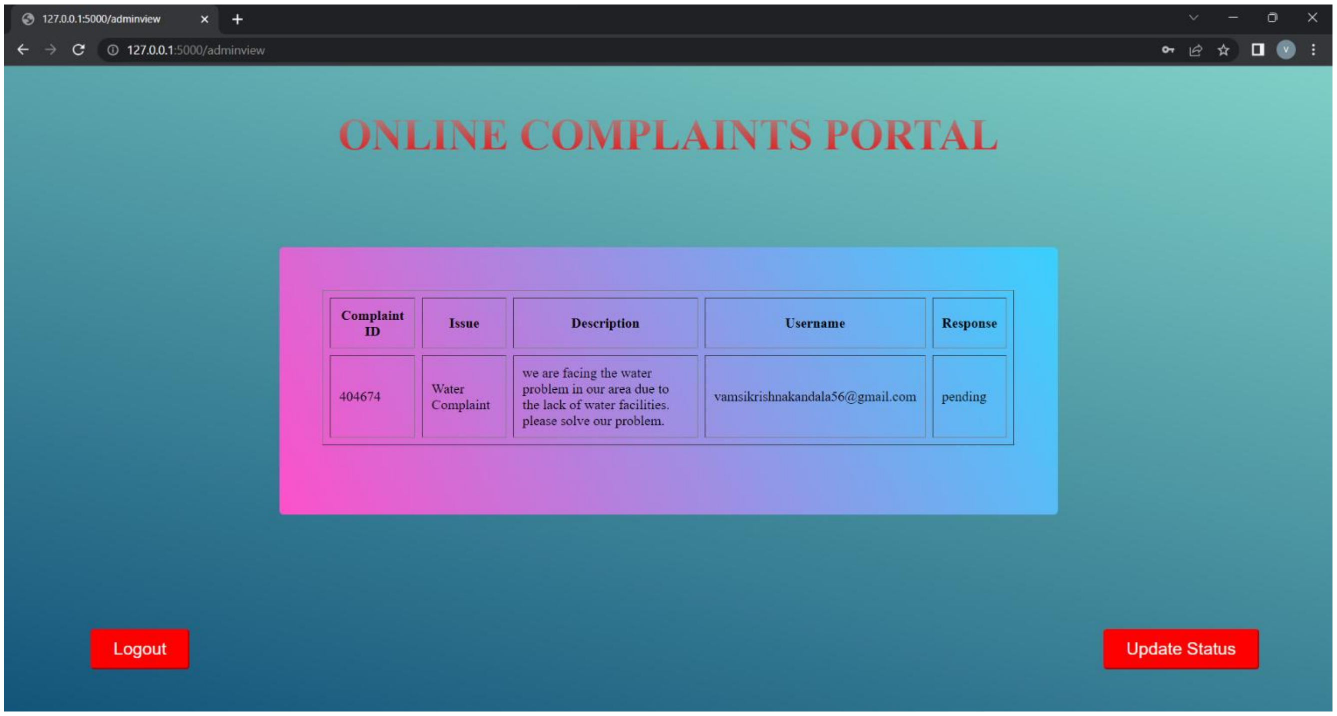
Complaint ID	Issue	Description	Response
404674	Water Complaint	we are facing the water problem in our area due to the lack of water facilities. please solve our problem.	pending

Admin Side:

Admin Login

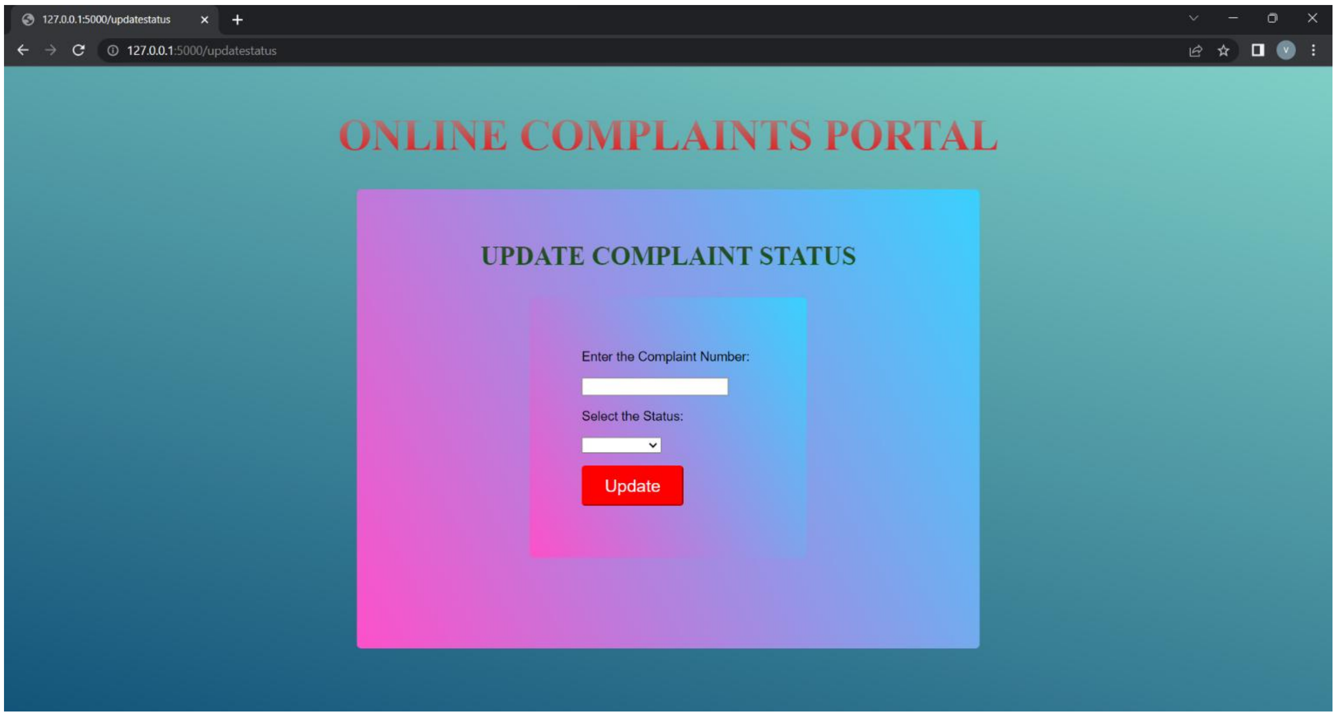


Admin View of Registered Complaints



Complaint ID	Issue	Description	Username	Response
404674	Water Complaint	we are facing the water problem in our area due to the lack of water facilities. please solve our problem.	vamsikrishnakandala56@gmail.com	pending

Update Form to Update the Status of the Complaint



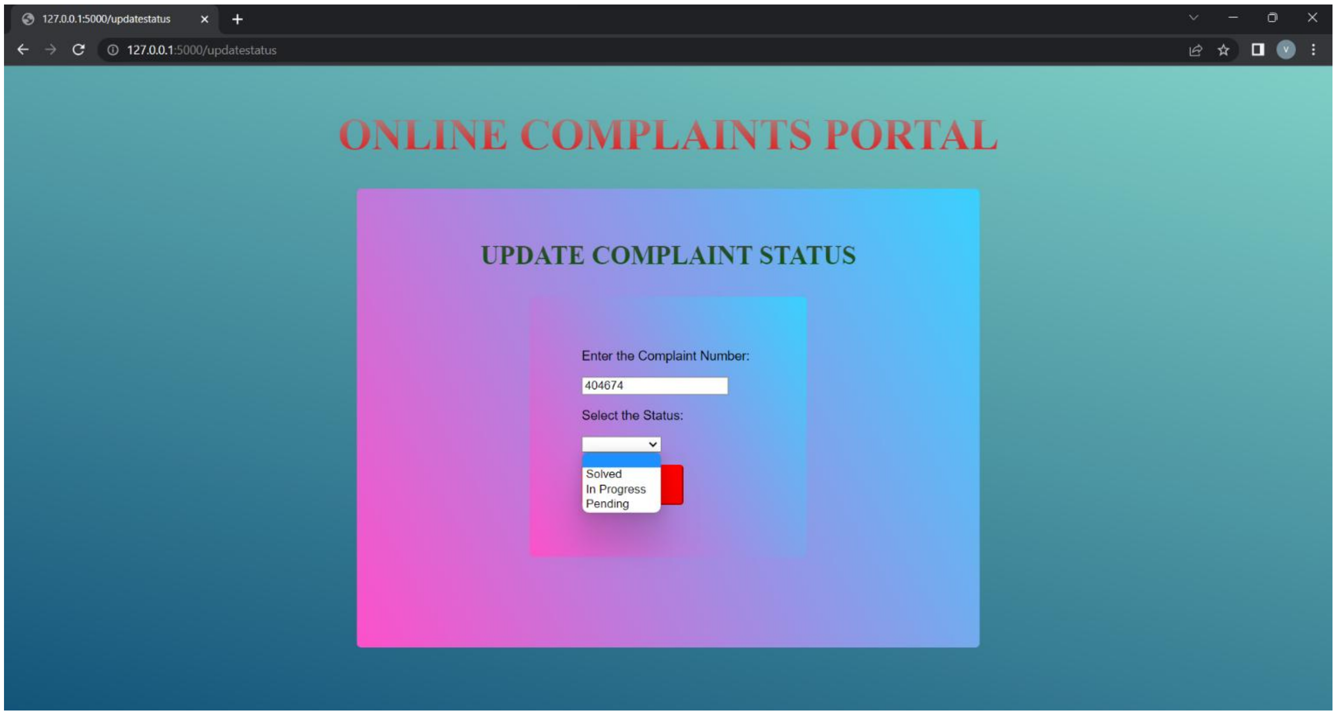
127.0.0.1:5000/updatestatus

ONLINE COMPLAINTS PORTAL

UPDATE COMPLAINT STATUS

Enter the Complaint Number:

Select the Status:



127.0.0.1:5000/updatestatus

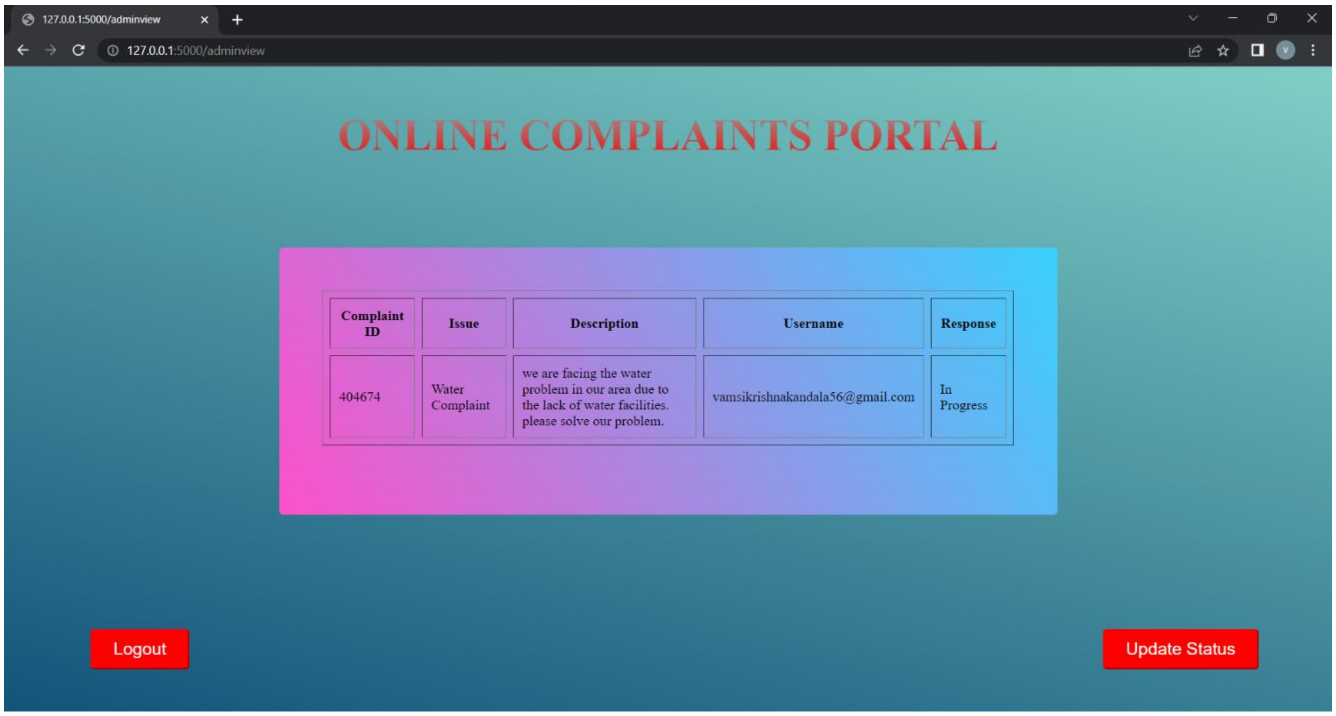
ONLINE COMPLAINTS PORTAL

UPDATE COMPLAINT STATUS

Enter the Complaint Number: 404674

Select the Status:

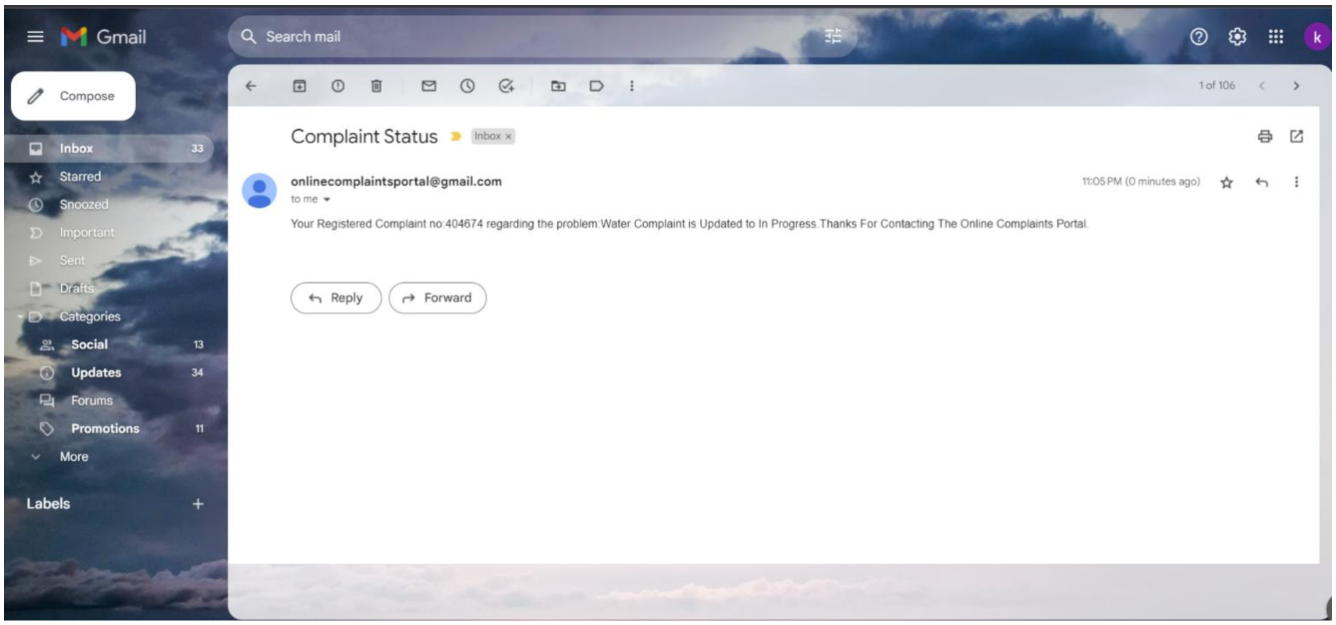
After Updating Status to In Progress


127.0.0.1:5000/adminview

ONLINE COMPLAINTS PORTAL

Complaint ID	Issue	Description	Username	Response
404674	Water Complaint	we are facing the water problem in our area due to the lack of water facilities. please solve our problem.	vamsikrishnakandala56@gmail.com	In Progress

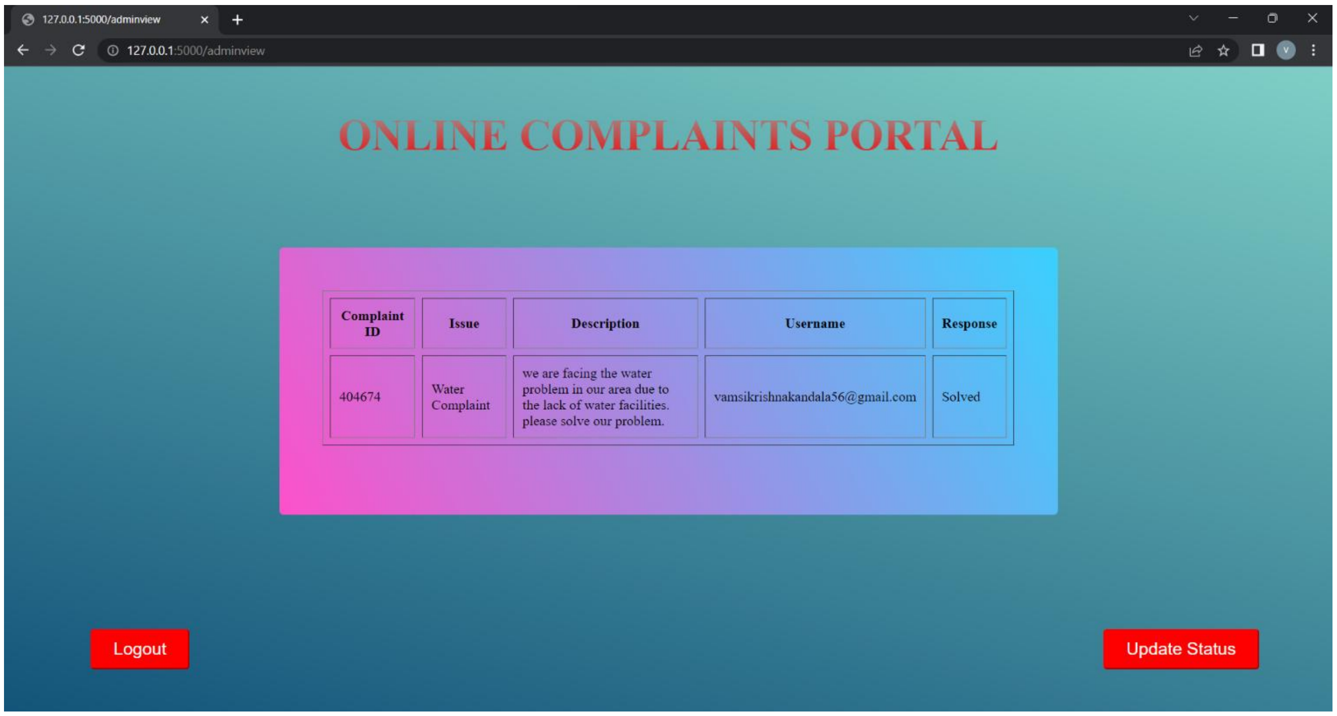
User will get the mail regarding update of status


Complaint Status

onlinecomplaintsportal@gmail.com
to me
11:05 PM (0 minutes ago)

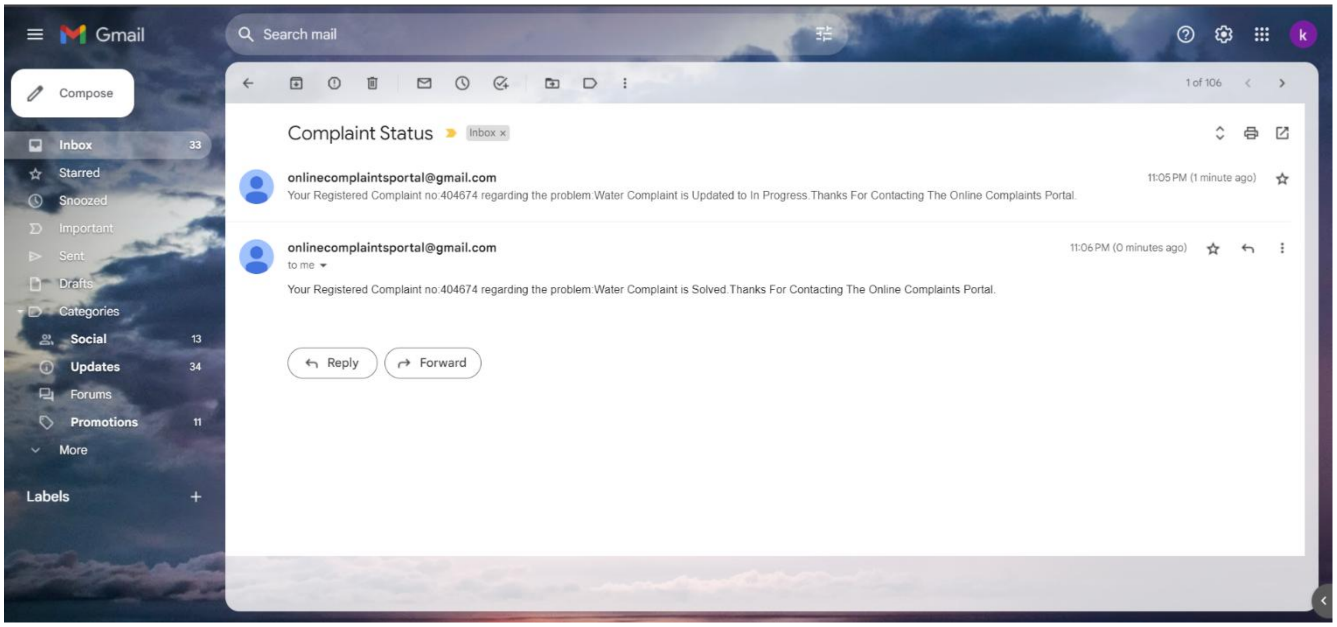
Your Registered Complaint no:404674 regarding the problem:Water Complaint is Updated to In Progress.Thanks For Contacting The Online Complaints Portal.

After Updating Status to Solved



Complaint ID	Issue	Description	Username	Response
404674	Water Complaint	we are facing the water problem in our area due to the lack of water facilities. please solve our problem.	vamsikrishnakandala56@gmail.com	Solved

User will get the mail regarding update of status



Complaint Status Inbox ×

onlinecomplaintsportal@gmail.com 11:05 PM (1 minute ago)
Your Registered Complaint no:404674 regarding the problem:Water Complaint is Updated to In Progress.Thanks For Contacting The Online Complaints Portal.

onlinecomplaintsportal@gmail.com 11:06 PM (0 minutes ago)
to me
Your Registered Complaint no:404674 regarding the problem:Water Complaint is Solved.Thanks For Contacting The Online Complaints Portal.

6.TESTING

Software testing is a critical element of the software quality assurance and represents the ultimate review of specification, design and coding. Testing is the exposure of the system to trial input to see whether it produces correct output.

Testing Phases: Software testing phases include the following:

- Test activities are determined and test data selected.
- The test is conducted and test results are compared with the expected results.

There are various types of testing:

Unit Testing: Unit testing is essentially for the verification of the code produced during the coding phase and the goal is test the internal logic of the module/program. Int the Generic code project, the unit testing is done during coding phase of data entry forms whether the functions are working properly or not. In this phase all the drivers are tested they are rightly connected or not.

Integration Testing: All the tested modules are combined into subsystems, which are then tested. The goal is to see if the modules are properly integrated, and the emphasis being on the testing interfaces between the modules. The generic code integration testing is done mainly on table creation module and insertion module.

System Testing: It is mainly used if the software meets its requirements. The reference document for this process is the requirement document. **Acceptance Testing:** It is performed with realistic data of the client to demonstrate that the software is working satisfactorily.

Testing Methods: Testing is a process of executing a program to find out errors. If testing is conducted successfully, it will uncover all the errors in the software. Any testing can be done basing on two ways:

White Box Testing: It is a test case design method that uses the control structures of the procedural design to derive the test cases. Using this testing a software engineer can derive the following test cases: Exercise all the logical decisions on either true or false sides. Execute all loops at their boundaries and within their operational boundaries. Exercise the internal data structures to assure their validity.

Black Box Testing: It is a test case design method used on the functional requirements of the software. It will help a software engineer to derive sets of input conditions that will exercise all the functional requirements of the program.

Black box Testing attempts to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures
- Performance errors
- Initialization and termination errors

By Black box testing we derive a set of test cases that satisfy the following criteria:

- Test cases that reduce by a count that is greater than one, the number of additional test cases that must be designed to achieve reasonable testing.
- Test cases that tell us something about the presence or absence of classes of errors rather than errors associated only with a specific test at hand.

TEST APPROACH:

Testing can be done in two ways:

- Bottom up approach
- Top down approach

Bottom up approach: Testing can be performed starting from smallest and lowest level modules and proceeding one at a time. For each module in bottom up testing a short program executes the module and provides the needed data so that the module is asked to perform the way it will when embedded with in the larger system. When bottom level modules are tested attention turns to those on the next level that use the lower level ones they are tested individually and then linked with the previously examined lower level modules.

Top down approach: This type of testing starts from upper level modules, since the detailed activities usually performed in the lower level routines are not provided stubs are written. A stub is a module shell called by upper level module and that when reached properly will return a message to the calling module indicating that proper interaction occurred. No attempt is made to verify the correctness of the lower level module.

7. Conclusion:

The Web Application Using Python Flask for the registering complaints using CRUD Operation with the Connection to Database Using MySQL was successful in creating a secure and user-friendly experience for users. The documentation provides a comprehensive overview of the project and can be used as a reference for future development or maintenance.

8. BIBLIOGRAPHY

WEBSITES REFERRED:

https://www.python.org/
https://en.wikipedia.org/wiki/Python_(programming_language)
https://www.learnpython.org/
https://realpython.com/
https://flask.palletsprojects.com/en/2.3.x/
https://www.tutorialspoint.com/flask/index.htm
https://www.fullstackpython.com/flask.html