|  |  |  |
| --- | --- | --- |
| **Table of Content** | | |
| **Chapter no.** | **Topic** | **Page. no.** |
| **1.** | **Chapter 1 - Introduction**  1.1 Objectives and Scope of Project  1.2 Theoretical Background  1.3 Definition of the Directory  1.3.1 Need for System  1.3.1 Proposed System  1.4 Key Tasks of the Project | **9**  **10**  **11-12**  **13** |
| **2.** | **Chapter 2 - System Analysis**  2.1 System Analysis Introduction  2.2 Software Analysis Specification  2.3 Feasibility Studies  2.3.1 Technical Feasibility  2.3.2 Economical Feasibility  2.3.3 Operational Feasibility  2.4 System Security | **15-16**  **17**  **17-19**  **19** |
| **3.** | **Chapter3 - Software Requirement Specification**  3.1 Software Requirement Analysis Introduction  3.2 System Requirements  3.2.1 Hardware Specification  3.2.2 Software Specification    3.3 Features of Software  3.3.1 Windows 10  3.3.2 Front End Tool(PHP)  3.3.3 Back End Tool(MySQL)  3.4 Methodology Adopted  3.5 Architectural Design  3.6 Database Design | **21**  **21-22**  **22-26**  **26-27**  **28**  **29-30** |
| **4.** | **Chapter 4 - System Design And Development**  4.1 System Design  4.2 Dataflow Diagrams  4.2.1 0-level DFD  4.2.2 1-level DFD  4.2.3 2-level DFD  4.3 E.R. Diagram | **32-33**  **34-39**  **40** |
| **5.** | **Chapter 5 - Snapshots Of Outputs** | **42-63** |
| **6.** | **Chapter 6 - System Testing And Implementation**  6.1 Testing  6.1.1 Unit Testing  6.1.2 Integration Testing  6.1.3 Component Interface Testing  6.1.4 System Testing  6.1.5 Operational Acceptance Testing  6.2 System Implementation  6.3 Post Implementation Review | **65-68**  **69**  **69-70** |
| **7.** | **Chapter7 - Scope And Conclusion**  7.1 Future Scope  7.2 Limitation  7.3 Conclusion | **72**  **72**  **73** |
| **8.** | **Chapter8 – References** | **75** |

CHAPTER 1

**Introduction**

* 1. **OBJECTIVES AND SCOPE OF THE PROJECT**

An effective **GRIEVANCE** management system is integral to providing quality public service. It helps to measure public satisfaction and is a useful source of information and feedback for improving services. In Online Public Grievance Redressal System we can report new complains, see the status of existing complains and also the administrator or the director of the organization can access all grievance and take suitable actions for each grievances.

**The following objectives have been set:**

1) Smooth flow of data without any hurdles.

2) Adequate validation checks for data entry.

3) Adequate security of data.

4) Facility to update data from time to time.

5) Prompt and specific retrieval of data.

6) Flexibility in the system according to the changing environment.

7) Controlling redundancy in storing the same data multiple times.

8) Accuracy, timeliness and comprehensiveness of the system output.

9) Stability and operability by people of average intelligence.

10) Enhancement in the completion of work within the constraints of time.

* 1. **THEORETICAL BACKGROUND**

In this era of technological advancement where everything is being computerized and the use of paperwork has reduced to minimal extend, and most of the works nowadays are done on internet. So, I decided to create a user friendly website based to PHP as front end and MySQL as back end tool to help general public to have online public grievances & redressal system.

This website system is a user friendly system with integration of HTML, CSS, PHP and MySQL. The webpage is divided into two parts i.e. PUBLIC SECTION and ADMINISTRATION SECTION, which you can notice in the webpage. The user’s section is further divided into two subparts: Lodge Grievance Section & Grievance Status Section, where the users/ public can lodge their grievance by entering their personal details and grievance details and the system generates a unique Grievance ID where users can see their status of their lodged grievance.

The second section is the administration section, which is also divided into two subparts: Verification Cell & Admin Login Section. The verification cell verifies the lodged grievance whether the details of the user/public is genuine or not, further this department sends the genuine grievances to the Administration department so that the administrator could verify the grievance. The administrator also sends email and SMS to the users whose grievance has been resolved. The administration department can also see the reports of the grievances which are further categorized into four parts: Monthly Reports, Age-Wise Reports, District-Wise Reports, and Custom Date-Wise Reports.

All precautions have been taken in creation of this website so that it is easy for the users to use it. Every page of the website is created after detailed understanding on every minute aspect which would make the user’s work more free and easy.

Hope this website would ease the work experience of our esteemed us

* 1. **DEFINATION OF GRIEVANCE REDRESSAL**

**Grievance Redressal** is a management- and governance-related process used commonly in [India](https://en.wikipedia.org/wiki/India). While the term "Grievance Redressal" primarily covers the receipt and processing of complaints from citizens and consumers, a wider definition includes actions taken on any issue raised by them to avail services more effectively.

The traditional approach to Grievance Redressal, handled through letters and complaint forms, has very little appeal and its usage rarely reflects the actual state of [customer satisfaction](https://en.wikipedia.org/wiki/Customer_satisfaction) or lack thereof. However, new Internet-based approaches used by the government and more by private organizations, such as [Public Grievance Portal (Govt. of India)](http://pgportal.gov.in/) and [ActPlease.com](http://www.actplease.com/).

Grievance Redressal mechanism is mandated in Government agencies and departments that are directly involved with serving citizens and organizations. Usually a [Public Relations Officer (PRO)](https://en.wikipedia.org/wiki/Chief_communications_officer) is designated with the role of receiving complaints and initiating corrective action, but this mechanism often fails on account of lack of authority vested in the PRO over officers of various capacities. The Government of India has made effort to systematize the nature of grievance Redressal through legislation, being driven by civil society agitations under leadership of [Anna Hazare](https://en.wikipedia.org/wiki/Anna_Hazare) and [Arvind Kejriwal](https://en.wikipedia.org/wiki/Arvind_Kejriwal) for enactment of the [Jan Lokpal Bill](https://en.wikipedia.org/wiki/Jan_Lokpal_Bill) into law.

To define the directory we have to study the needs of system, After this I will explain the proposed system.

**Following Points are defined for the definition of problem:**

1. Needs of the system

2. Proposed system

**1.3.1. NEED FOR SYSTEM**

The package that I have designed can handle the Grievance details without any difficulty & with a little bit of effort. As the work is done manually before, so it will be very time consuming & required a large efforts to maintain the files. By computerizing the system these files can be handled with a small effort & in less time.

The chances of duplicity of complaints are negligible. The Public Grievance Report can be generated easily by getting the information without any problem from all the related files. The package is designed by using GUI concept there for it is very user friendly & easy to use.

**1.3.2. PROPOSED SYSTEM**

The proposed system has a user friendly homepage in which the main page is being divided into two parts i.e. for the public and for the administrator/ director of the organization.

The public users can lodge a grievance from this portal in which they have to provide their personal details which includes their Name, D.O.B., Address, Locality, Phone No. etc. and grievance details in which they have to enter their grievance and have to mention in which department and officer the grievance pertains to. After lodging the grievance the users can see the status of their grievance by entering their grievance id.

The administrator or the director of the organization can open their homepage by entering the user id and password in which they can see the list of applications or grievances lodged by the public. The list is being categorized into two parts i.e. Pending List and Resolved List. After reviewing the list the director can take action for the grievance and make effective change and state reason for each grievance. The director can send email confirmation to the public describing that their grievance has been resolved and can check their status on the website. A SMS portal has also being integrated with this project in which the director can send SMS to the public users. For the future reference I have added a reports page for the administrator/ director of the organization in which they can see the statistics of reports which is being categorized into four types i.e. Monthly Reports, District Wise Reports, Age Wise Reports & Custom Date Wise Reports

**1.4 KEY TASKS OF THE PROJECT:**

i. To prepare database.

ii. To prepare web pages.

iii. To handle the control navigation from other pages

iv. Writing the Code Behind pages for the web pages.

v. Database Design

CHAPTER 2

**SYSTEM**

**ANALYSIS**

**2.1 SYSTEM ANALYSIS INTRODUCTION**

System analysis is the process of studying the grievance system and procedures, generally referred to as complain systems, to see how they can operate and whether improvement is needed. This may involve examining data movement and storage, machines and technology used in the system, programs that control the machines, people providing inputs, doing the processing and receiving the outputs.

**INVESTIGATION PHASE**

The investigation phase is also known as the fact-finding stage or the analysis of the current system. This is a detailed study conducted with the purpose of wanting to fully understand the existing system and to identify the basic information requirements. Various techniques may be used in fact -finding and all fact obtained must be recorded. A thorough investigation was done in every effected aspect when determining whether the proposed system is feasible enough to be implemented.

**INVESTIGATION**

As it was essential for us to find out more about the present system, we used the following methods to gather the information: -

**1. Observation: -** Necessary to see the way the system works first hand.

**2. Document sampling: -** These are all the documents that are used in the system. They are necessary to check all the data that enters and leaves the system.

**3. Questionnaires:-**  These were conducted to get views of the other employees who are currently employed in the system.

**ANALYSIS OF THE INVESTIGATION**

**STRENGTHS OF SYSTEM**

**1. No complex equipment: -** The equipment that is used is very simple and no special skills have to be mastered to be able to operate the system. Therefore no training is required for the employees.

**2. Low cost: -** There is little money spent in maintaining the present system other than buying the necessary office equipment.

**2.2 SOFTWARE ANALYSIS SPECIFICATION**

The software analysis specification is produced at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by establishing a complete information description, a detailed functional description, a representation of system behavior, an indication of performance requirement and design constraints appropriate validation criteria, and other information pertinent to requirement.

The introduction to software requirements specification states the goals and objectives of the software, describing it in the context of the computer based system. The Information Description provides a detailed description of the problem that the software must solve. Information content, flow and structure are documented.

A description of each function required to solve the problem is presented in the Functional Description. Validation Criteria is probably the most important and ironically the most often neglected section of the software requirement specification. Software requirement specification can be used for different purpose.

**2.3 FEASIBILITY STUDIES**

Prior to stating whether the system we have to develop is feasible or not we believe that we should emphasize on what is implied by the word “Feasibility”. Feasibility is the measure of how beneficial or practical the development of the system will be to the organization. It is a preliminary survey for the systems investigation. It aims to provide information to facilitate a later in-depth investigation.

**TYPES**

There are various measures of feasibility that helps to decide whether a particular project is feasible or not. These measures include –

1. Operational Feasibility

2. Technical Feasibility

3. Economical Feasibility

Each of these types will be explained in detail throughout the project report

**2.3.1 OPERATION FEASIBILITY**

A proposed system is beneficial only if it can be turned into an information system that will meet the operational requirements of an organization. A system often fails if it does not fit within existing operations and if users resist the change. Important issues a systems developer must look into are:

* Will the new system be used if implemented in an organization?
* Are there any major barriers to implementation or is proposed system accepted without destructive resistance?

The whole purpose of computerizing the Complaint Management is to handle the work much more accurately and efficiently with less time consumption. There will be additional work to be completed, because now the cellular company will have to maintain database of both their employees as well as their Customers.

Compared to the semi-computerized system the chances of avoiding errors in a computerized system is much higher because the user need not stress himself unnecessarily resulting in recklessness. Unlike the semi-computerized system there would be backup data for all the information concerning the daily transactions occurred within the organization.

Another important fact to be regarded is the security control, which is handled by the system. Since data regarding each Customer and the Organization is confidential, security is a key issue. Information falling into the wrong hands could jeopardize the entire organization. Unlike in semi-computerized systems The proposed system offers adequate control to protect the organization against fraud and embezzlement and guarantees the accuracy and Security of data and information. This is handled by the system providing individuals with separate login names and passwords.

The new system is user-friendlier, which enables the end-user to complete his/her work efficiently and accurately with interest. After taking the above fact into consideration we can state the operating of the proposed system within the organization is feasible.

**2.3.2 TECHNICAL FEASIBILITY**

Based on the outline design of the system requirements in terms f inputs, output, Procedures, the technical issues raised during technical feasibility include:

* Does the necessary technology exist to do what is proposed?
* Does the proposed equipment have the technical capacity to hold the data required to use in the new system?
* Adequate responses provided by the proposed system?
* Is the system flexible enough to facilitate expansion?
* Is there any technical guarantee of accuracy, reliability, ease of access and data security?

The system developer’s task is to view needed capabilities in light of currently available technology. Our site works hand in hand with high technology. A database has to be maintained in order to update and backup data whenever required. To create databases we use SQL server. After taking the above facts into consideration we can state that the new proposed system is technically feasible.

**2.3.3 ECONOMICAL FEASIBILITY**

In making recommendations a study of the economics of the proposed system should be made. Even though finding out the costs of the proposed project is difficult we assume and estimate the costs and benefits as follows. According to the computerized system we propose, the costs can be broken down in two categories.

1. Costs associated with the development of the system.

2. Costs associated with operating the system.

**2.4 SYSTEM SECURITY**

System security is a vital aspect when it comes to developing a system. The system should ensure the facility of preventing unauthorized personnel from accessing the information and the data within the system. The system should provide total protection for each user’s information so that the integrity of data is sustained and also prevent hackers from hacking the system.

The proposed system ensures the security and the integrity of data. This is done by providing a password login system for each authorized users. And for example the System Administrator has access to all kinds of information.

By providing this facility information is properly managed and information is protected. For example the system administrator’s day to day tasks are lessened and easier because he doesn’t have to have a constant eye on the system and worry about hackers hacking the system.

CHAPTER 3

**SOFTWARE**

**REQUIREMENT**

**SPECIFICATION**

**3.1 SOFTWARE REQUIREMENT SPECIFCATION INTRODUCTION**

A software requirements specification (SRS) is a description of a [software system](https://en.wikipedia.org/wiki/Software_system) to be developed, laying out [functional](https://en.wikipedia.org/wiki/Functional_requirement) and [non-functional requirements](https://en.wikipedia.org/wiki/Non-functional_requirements), and may include a set of [use cases](https://en.wikipedia.org/wiki/Use_case) that describe interactions the users will have with the software.

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers (in market-driven projects, these roles may be played by the marketing and development divisions) on what the software product is to do as well as what it is not expected to do. Software requirements specification permits a rigorous assessment of requirements before design can begin and reduces later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules.

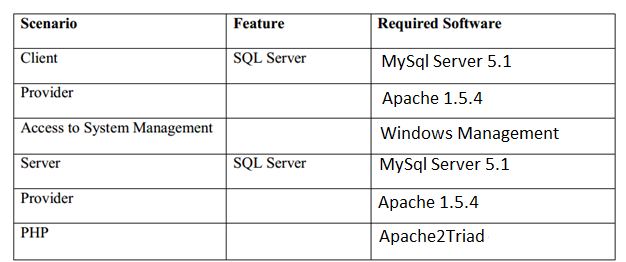
The software requirements specification document enlists enough and necessary requirements that are required for the project development. To derive the requirements we need to have clear and thorough understanding of the products to be developed or being developed. This is achieved and refined with detailed and continuous communications with the project team and customer till the completion of the software.

**3.2 SYSTEM REQUIREMENTS**

**3.2.1 HARDWARE REQUIREMENTS**

****

**3.2.2 SOFTWARE REQUIREMENTS**



**3.3 FEATURES OF SOFTWARES**

**3.3.1 Windows 10**

**Windows 10** is a [personal computer](https://en.wikipedia.org/wiki/Personal_computer) [operating system](https://en.wikipedia.org/wiki/Operating_system) released by [Microsoft](https://en.wikipedia.org/wiki/Microsoft) as part of the [Windows NT](https://en.wikipedia.org/wiki/Windows_NT) family of operating systems. Officially unveiled in September 2014 following a brief demo at [Build 2014](https://en.wikipedia.org/wiki/Build_%28developer_conference%29), the operating system entered a public beta testing process in October 2014, leading up to and continuing through the consumer release of Windows 10 on July 29, 2015, and its release to volume licensing on August 1, 2015. To encourage its adoption, Microsoft announced that during its first year of availability, Windows 10 would be made available free of charge to users of [genuine](https://en.wikipedia.org/wiki/Windows_Genuine_Advantage) copies of eligible editions of [Windows 7](https://en.wikipedia.org/wiki/Windows_7) or [Windows 8.1](https://en.wikipedia.org/wiki/Windows_8.1).

Windows 10 introduces what Microsoft described as a "universal" application architecture; expanding on [Metro-style apps](https://en.wikipedia.org/wiki/Metro-style_apps), these apps can be designed to run across multiple Microsoft product families with nearly identical code—including [PCs](https://en.wikipedia.org/wiki/Personal_computer), [tablets](https://en.wikipedia.org/wiki/Tablet_computer), [smart phones](https://en.wikipedia.org/wiki/Smartphone), [embedded systems](https://en.wikipedia.org/wiki/Embedded_system), [Xbox One](https://en.wikipedia.org/wiki/Xbox_One), [Surface Hub](https://en.wikipedia.org/wiki/Surface_Hub) and [Halo lens](https://en.wikipedia.org/wiki/HoloLens). The Windows user interface was revised to handle transitions between a mouse-oriented interface and a [touch screen](https://en.wikipedia.org/wiki/Touchscreen)-optimized interface based on available input devices—particularly on [2-in-1 PCs](https://en.wikipedia.org/wiki/2-in-1_PC); both interfaces include an updated [Start menu](https://en.wikipedia.org/wiki/Start_menu) that comprises a design similar to Windows 7 with 8's tiles.

**3.3.2 FRONT END TOOL (PHP)**

**PHP: Hypertext Preprocessor** is a widely used, general-purpose scripting language that was originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. As a general-purpose programming language, PHP code is processed by an interpreter application in command-line mode performing desired operating system operations and producing program output on its standard output channel. It may also function as a graphical application. PHP is available as a processor for most modern web servers and as a standalone interpreter on most operating systems and computing platforms.

PHP was originally created by Rasmus Lerdorf in 1995 and has been in continuous development ever since. The main implementation of PHP is now produced by the PHP Group and serves as the *de facto* standard for PHP as there is no formal specification. PHP is free software released under the PHP License.

PHP is a general-purpose scripting language that is especially suited to server-side web development where PHP generally runs on a web server. Any PHP code in a requested file is executed by the PHP runtime, usually to create dynamic web page content. It can also be used for command-line scripting and client-side GUI applications. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems. It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.

PHP primarily acts as a filter, taking input from a file or stream containing text and/or PHP instructions and outputs another stream of data; most commonly the output will be HTML. Since PHP 4, the PHP parser compiles input to produce byte code for processing by the Zend Engine, giving improved performance over its interpreter predecessor.

Originally designed to create dynamic web pages, PHP now focuses mainly on server-side scripting, and it is similar to other server-side scripting languages that provide dynamic content from a web server to a client.

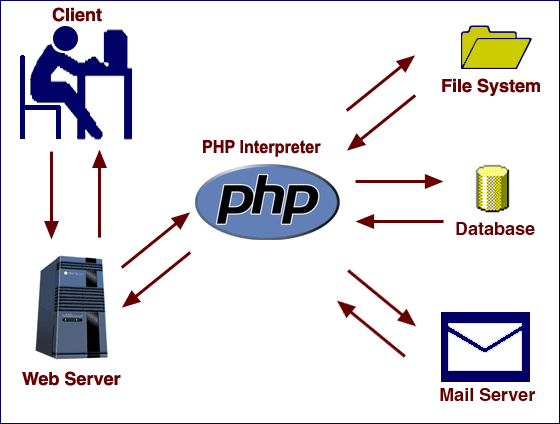
The LAMP architecture has become popular in the web industry as a way of deploying web applications. PHP is commonly used as the *P* in this bundle alongside Linux, Apache and My SQL... WAMP packages (Windows/ Apache/ MySQL / PHP) and MAMP packages (Macintosh / Apache / MySQL / PHP) are also available.

As of April 2007, over 20 million Internet domains had web services hosted on servers with PHP installed and mod php was recorded as the most popular Apache HTTP Server module.

**ADVANTAGES TO PHP:-**

* Open Source, readily available (you can be using it today) and dual-   
  licensed - if you are doing non-profit work or not licensing, there is no cost.
* Very easy to understand Syntax, some really cool features (arrays   
  are something else!)
* Interfaces very easily with Apache/MySQL
* Server side.
* Lots of good source code out there to use and/or learn from, as well   
  as many useful libraries for working with PDFs, graphics, etc.
* Lots of good books and on-line help (php.net is great)
* Platform agnostic can run on Windows Linux or Mac servers. Also   
  very scalable.
* Lots of hosting services have it ready to use, no special   
  configuration (except if you have special security needs)
* Pretty easy to access other web-based tools through PHP (i.e. google   
  maps, etc

**HOW PHP WORKS.**



**3.3.3 BACK END TOOL (MySQL)**

MySQL is an [open-source](https://en.wikipedia.org/wiki/Open-source) [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS);in July 2013, it was the world's second most widely used RDBMS, and the most widely used open-source [client–server model](https://en.wikipedia.org/wiki/Client%E2%80%93server_model) RDBMS. It is named after co-founder [Michael Widenius](https://en.wikipedia.org/wiki/Michael_Widenius)'s daughter, my. The [SQL](https://en.wikipedia.org/wiki/SQL) acronym stands for [Structured Query Language](https://en.wikipedia.org/wiki/Structured_Query_Language). The MySQL development project has made its [source code](https://en.wikipedia.org/wiki/Source_code) available under the terms of the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), as well as under a variety of [proprietary](https://en.wikipedia.org/wiki/Proprietary_software) agreements. MySQL was owned and sponsored by a single [for-profit](https://en.wikipedia.org/wiki/Business) firm, the [Swedish](https://en.wikipedia.org/wiki/Sweden) company [MySQL AB](https://en.wikipedia.org/wiki/MySQL_AB), now owned by [Oracle Corporation](https://en.wikipedia.org/wiki/Oracle_Corporation). For proprietary use, several paid editions are available, and offer additional functionality.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used [LAMP](https://en.wikipedia.org/wiki/LAMP_%28software_bundle%29) open source web application software stack (and other "[AMP](https://en.wikipedia.org/wiki/List_of_AMP_packages)" stacks). LAMP is an acronym for "[Linux](https://en.wikipedia.org/wiki/Linux), [Apache](https://en.wikipedia.org/wiki/Apache_HTTP_Server), MySQL, [Perl](https://en.wikipedia.org/wiki/Perl)/[PHP](https://en.wikipedia.org/wiki/PHP)/[Python](https://en.wikipedia.org/wiki/Python_%28programming_language%29)." [Free-software](https://en.wikipedia.org/wiki/Free_software)-open source projects that require a full-featured database management system often use MySQL. Applications that use the MySQL database include: [TYPO3](https://en.wikipedia.org/wiki/TYPO3), [MODx](https://en.wikipedia.org/wiki/MODx), [Joomla](https://en.wikipedia.org/wiki/Joomla), [WordPress](https://en.wikipedia.org/wiki/WordPress), [phpBB](https://en.wikipedia.org/wiki/PhpBB), [MyBB](https://en.wikipedia.org/wiki/MyBB), [Drupal](https://en.wikipedia.org/wiki/Drupal) and other software. MySQL is also used in many high-profile, large-scale [websites](https://en.wikipedia.org/wiki/Website), including [Google](https://en.wikipedia.org/wiki/Google) (though not for searches), [Facebook](https://en.wikipedia.org/wiki/Facebook), [Twitter](https://en.wikipedia.org/wiki/Twitter), [Flickr](https://en.wikipedia.org/wiki/Flickr), and [YouTube](https://en.wikipedia.org/wiki/YouTube).

**3.4 METHODOLOGY ADOPTED**

Prototyping Model has been used for software development according to which a throwaway prototype of the proposed system, based on the currently known requirements, is given to the user so that he has a fair idea about how the proposed system is going to be like. This will help in deciding the interface, input and output requirements.

It can be easily adjudged that inputs and outputs are big in number, can increase exponentially and may create a big chaos if not restricted properly. As the user spends some time on the prototype, he will become more precise about his own input output Requirements. This prototype will provide him with an environment analogous to the proposed system’s environment. Due to object oriented support, various concepts (like reusability, polymorphism, isolation etc.) are already there but for the efficient management of system components, Component based Software Engineering will also be exercised which will help in a resultant library of components, the benefit of which will be reusability and fast development.

Due to lack of hierarchical structure in object oriented approach, there is no meaning of Bottom-up or Top-down testing. Testing will begin from the rudimentary levels of the system and will move towards higher level components, which will be based on design phase rather than coding phase Words.

**Outline of the prototyping process**

The process of prototyping involves the following steps:

1. **Identify basic** [**requirements**](https://en.wikipedia.org/wiki/Requirement)**-**

Determine basic requirements including the input and output information desired. Details, such as security, can typically be ignored.

1. **Develop Initial Prototype -**

The initial prototype is developed that includes only user interfaces.

1. **Review**

The customers, including end-users, examine the prototype and provide feedback on additions or changes.

1. **Revise and Enhance the Prototype**

Using the feedback both the specifications and the prototype can be improved. Negotiation about what is within the scope of the contract/product may be necessary. If changes are introduced then a repeat of steps #3 and #4 may be needed.

**3.5 ARCHITECTURAL DESIGN**

**Three layers of project-**

**Presentation tier**

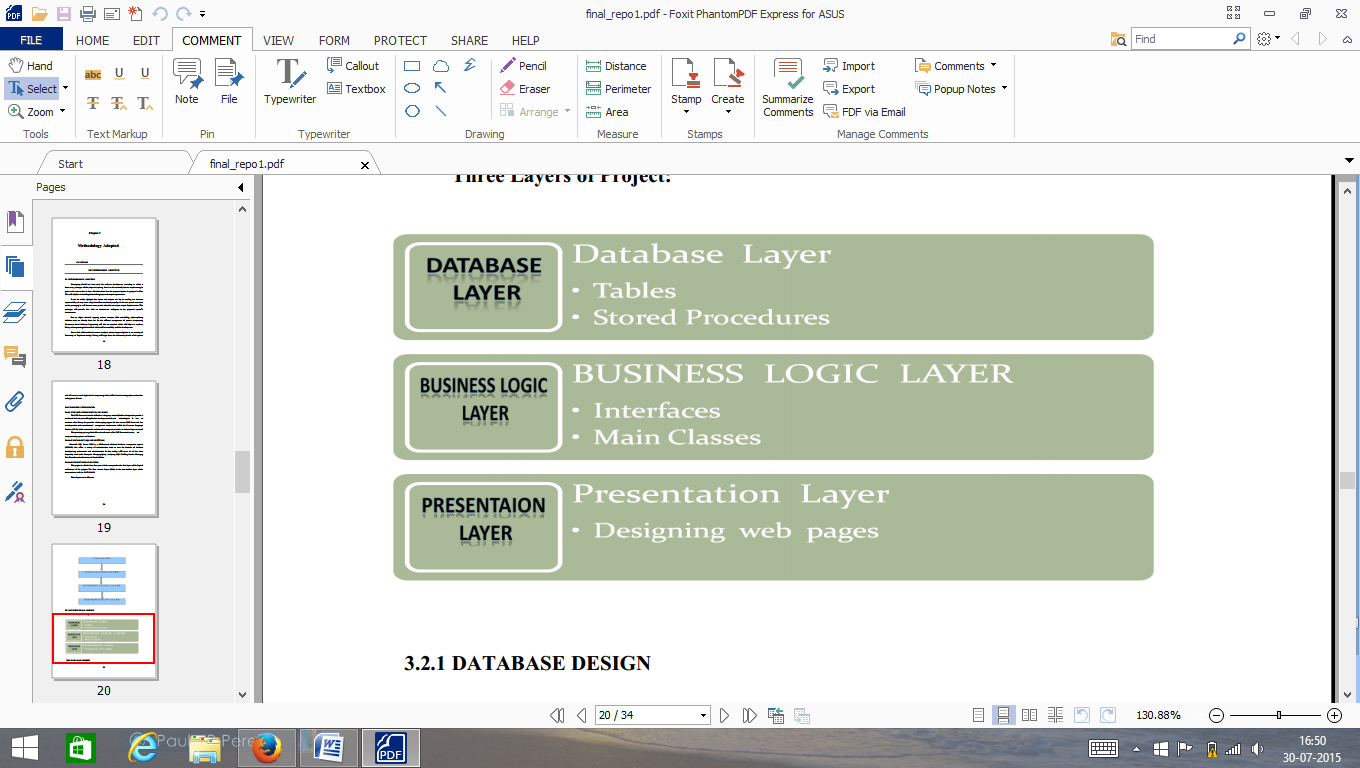
This is the topmost level of the application. The presentation tier displays information related to such services as browsing merchandise, purchasing and shopping cart contents. It communicates with other tiers by which it puts out the results to the browser/client tier and all other tiers in the network. (In simple terms it is a layer which users can access directly such as a web page, or an operating systems GUI)

**Application tier (**[**business logic**](https://en.wikipedia.org/wiki/Business_logic)**, logic tier, or middle tier)**

The logical tier is pulled out from the presentation tier and, as its own layer; it controls an application’s functionality by performing detailed processing.

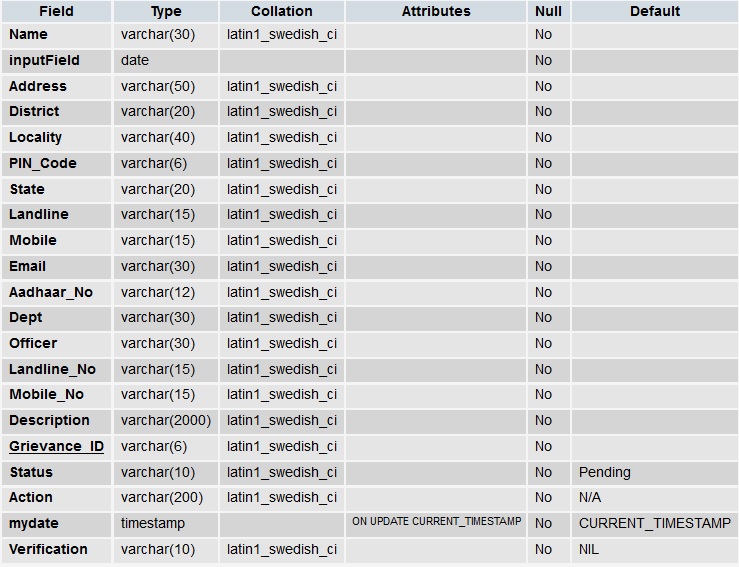
**Data tier**

The data tier includes the data persistence mechanisms (database servers, file shares, etc.) and the data access layer that encapsulates the persistence mechanisms and exposes the data. The data access layer should provide an [Application Programming Interface (API)](https://en.wikipedia.org/wiki/Application_programming_interface) to the application tier that exposes methods of managing the stored data without exposing or creating dependencies on the data storage mechanisms. Avoiding dependencies on the storage mechanisms allows for updates or changes without the application tier clients being affected by or even aware of the change. As with the separation of any tier, there are costs for implementation and often costs to performance in exchange for improved scalability and maintainability.



**3.6 DATABASE DESIGN**

**Lodge Table**

****

**Login Table**

****

**Verification Cell Login Table**

****

CHAPTER 4

**SYSTEM DESIGN**

**AND DEVELOPMENT**

**4.1 SYSTEM DESIGN**

Software design is a process through which requirements are translated in to a representation of software. Initially the representation depicts a holistic view of software. Subsequent refinement leads to a design representation that is very close to source code.

Since, we are following an Object oriented Design technique, the next step towards the development is to identify the classes and their relationships. A class is a description of an object type. Instances of classes are known as Objects. UML also provides tools for designing the system. Class diagrams enable us to establish relationship among various classes of the system. Before proceeding on to develop class diagrams, the next step is to identify the potential classes in the system.

Some of the basic tips in identifying the classes are:

* Analyzing the requirement statement.
* Use Cases.
* Application experts.
* Studying the system.

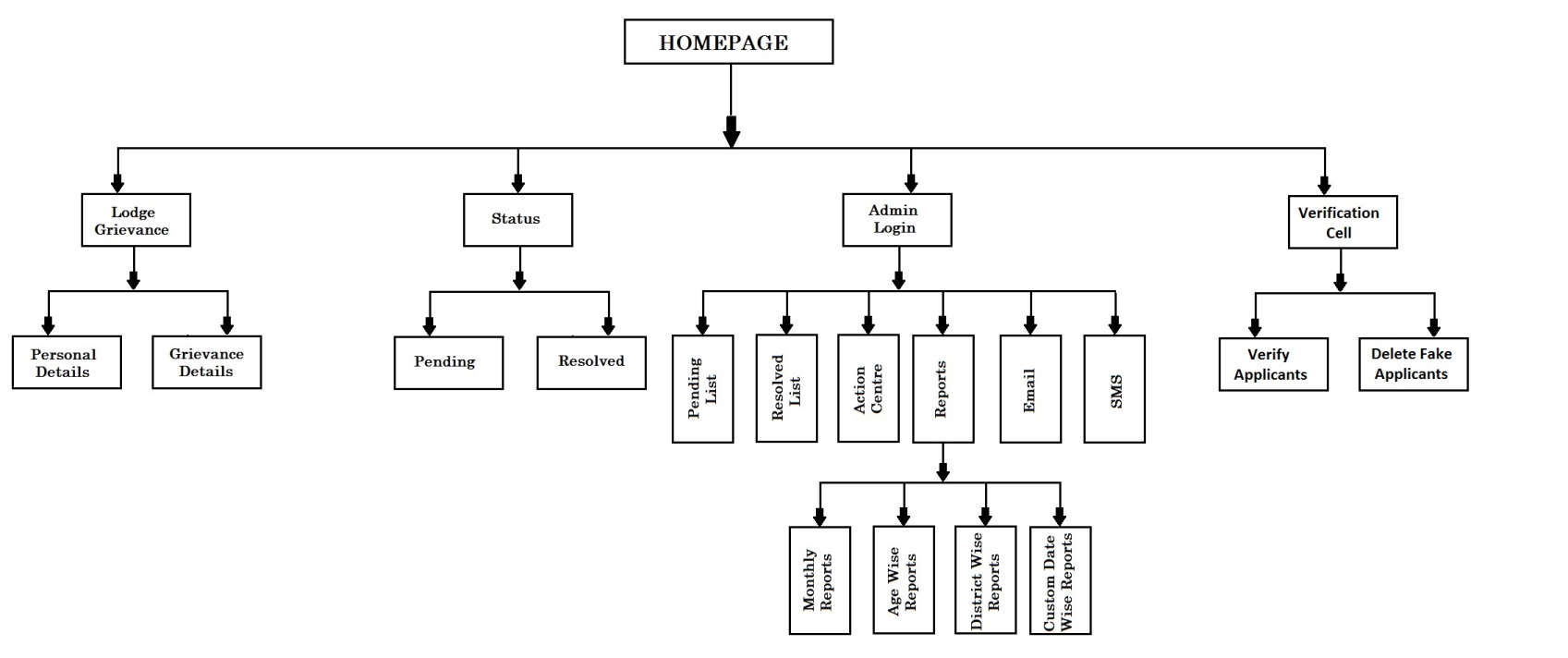
By following these simple rules during the initial process of analyzing, several classes get formulated. These classes are referred to as candidate classes and they represent the possible classes in a given system. It is not essential to incorporate all the identified candidate classes; some of them may also be dropped and are called unfit candidate classes.

A class icon is a rectangle with three sections in it. Horizontal lines across the rectangle divide the sections. The first section is where the class name is mentioned. In the second section the attributes or data members of the class and in the third section the methods or functions of the class are mentioned. A class diagram thus takes the form

In DFD the cardinality or multiplicity can be expressed at the ends of the association at the class where it is applicable. Whenever there is no mention of the cardinality then one is considered.

With the help of DFD, we designed the class diagram of our system, which looks like the following. The cardinality among the relationship is also mentioned.

**System Design Diagram**

****

**4.2 DATA FLOW DIAGRAMS**

In DFD, we give names to data flows, processes, and data stores. Although the names are descriptive of the data, they do not give details. So the following the DFD, our interest is to build some structured place to keep details of the contents of data flow, processes, and data store. A data dictionary is a structured repository of data about data. It is a set of rigorous definition of all DFD data element and data structure.

**DFD Symbols**

In the DFD, there are four symbols,

**1.** A Square defines a source (originator) or destination of system data.

**2.** An Arrow identifies data flow- data in motion .It is pipeline through which information flows.

**3.** A circle or a bubble (or a oval bubble) represents a process that transforms incoming data flow(s) into outgoing data flow(s)

**4.** An Open rectangle is a data store-data at rest, or temporary repository of data.

The DFD was first developed by “Larry Constantine” as a way of expressing system requirements in a graphical form. A DFD, also referred to as a bubble chart has a purpose of clarifying system requirements and identifying major transformations that will become the program in this system design.

* A box defines a source of destination or system data.
* An arrow line identifies the data flow or data in motion. It is a pipeline through which Information flows.
* A circle or bubble represents a process transform incoming data flow in to outgoing data flow.

**LEVEL 0**

Initially in the first level of the Data flow the level 0 explains the basic outline of the system. The end-user sends the packets to the system to determine the source and destination address. The diagram marked as the 0 represents the complete Packet watching system which simply represents the basic operation that is being performed by it in the initial level.

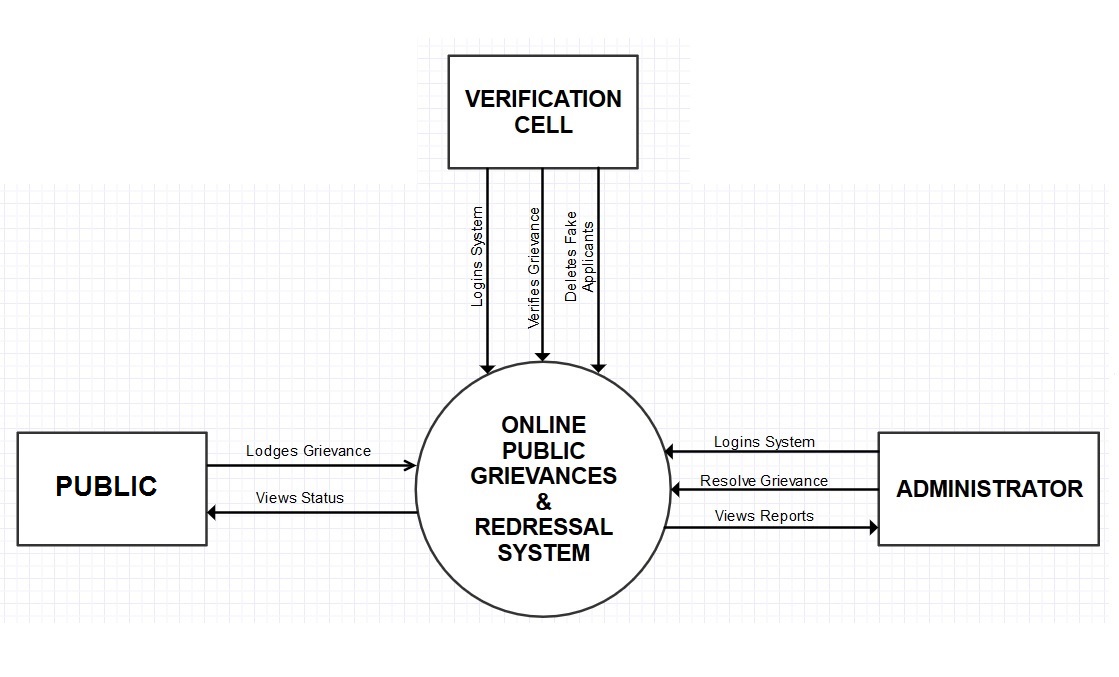
**LEVEL 1**

The level 1 of the Data flow diagram given explains in detail about the Packet watching system which was marked as 0 in the previous level. In this level the end-user who passes the request for the system enters into the first process, the capturing process and then to the processing module. After processing the packets it was send for storing.

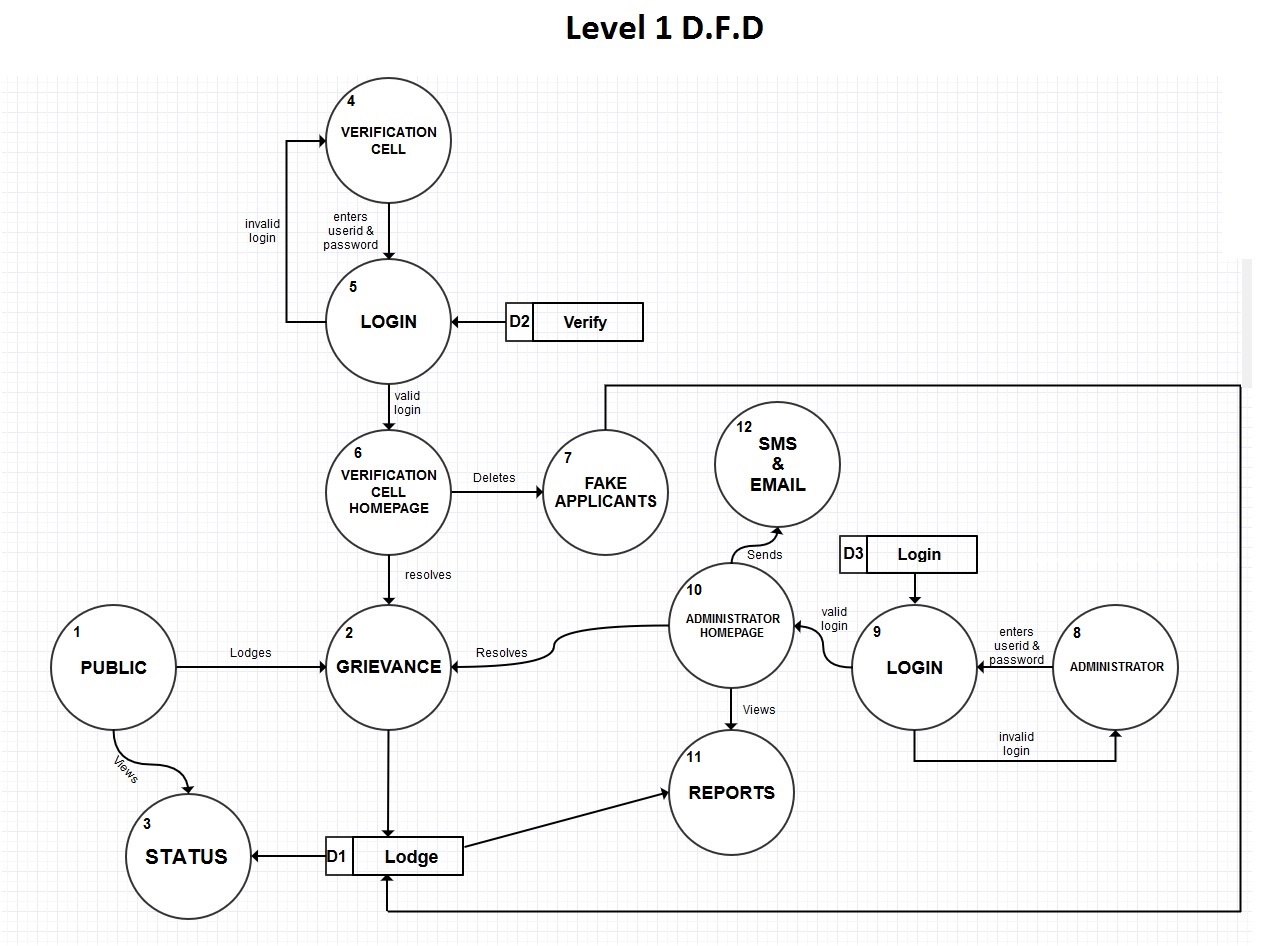
**LEVEL 2**

The level 2 provides the clear explanation about the whole system. In this level first we have to select the packet and perform test over that selected packets. Then identify the end address of the packet and send that packet for processing. After processing the packet it was send to the identity content. Then send the processed packet for storing and display the source and destination addresses.

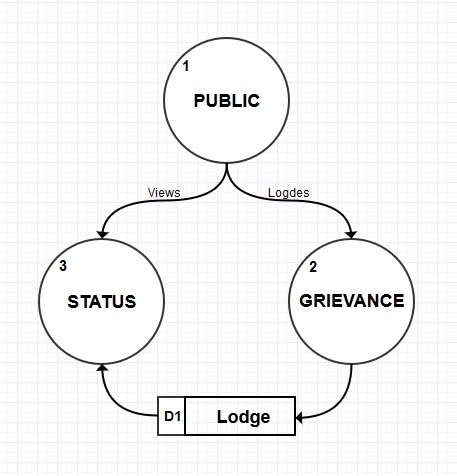
**4.2.1 LEVEL ZERO DFD DIAGRAM**

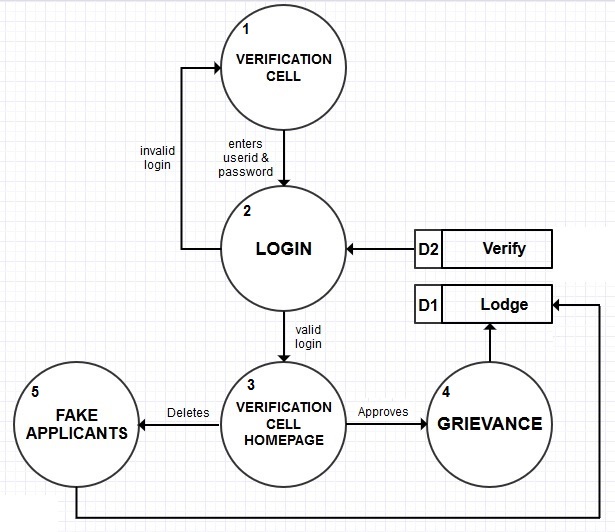


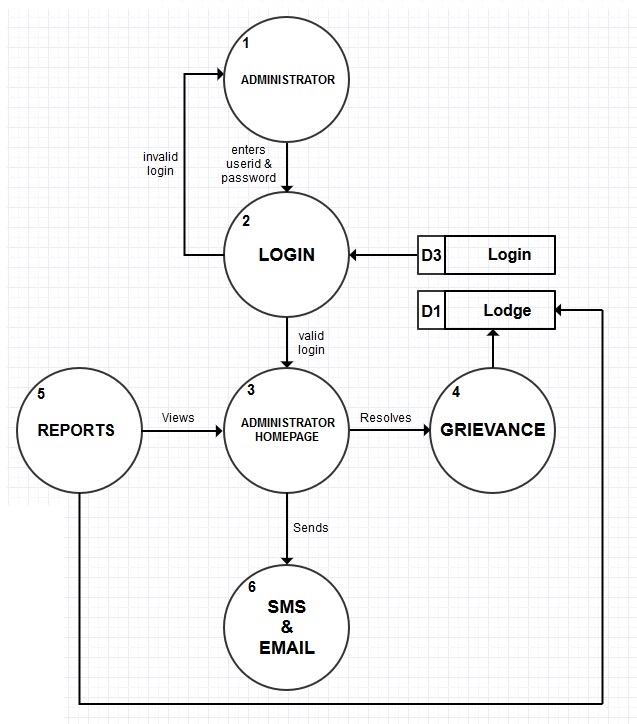
**4.2.2 LEVEL ONE DFD DIAGRAM**

****

**4.2.3 LEVEL TWO DFD DIAGRAM**

****

****

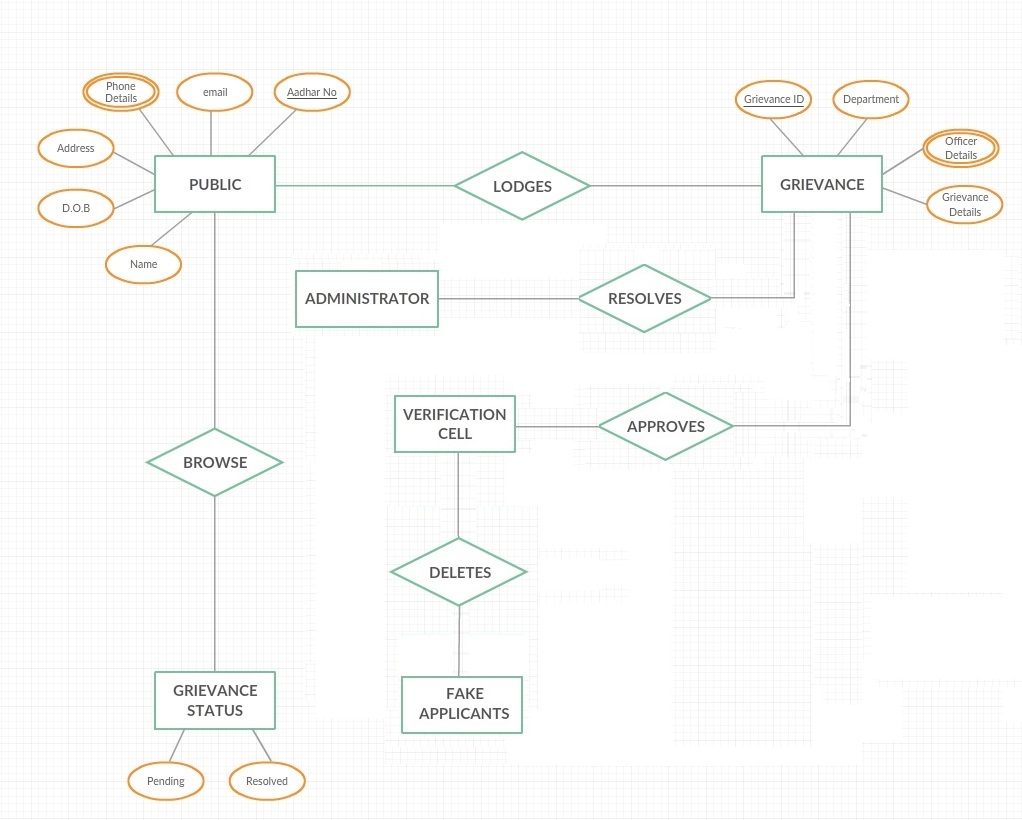
****

**4.3 ENTITY RELATIONSHIP DIAGRAM**

An ER diagram represents the following three elements:

* Entities: An entity is an object with a distinct set of properties that is easily identified.
* Attributes: An attribute is a property of an entity that differentiates it from other entities and provides information about the entity.
* Relationships: A relationship is a crucial part of the design of database.

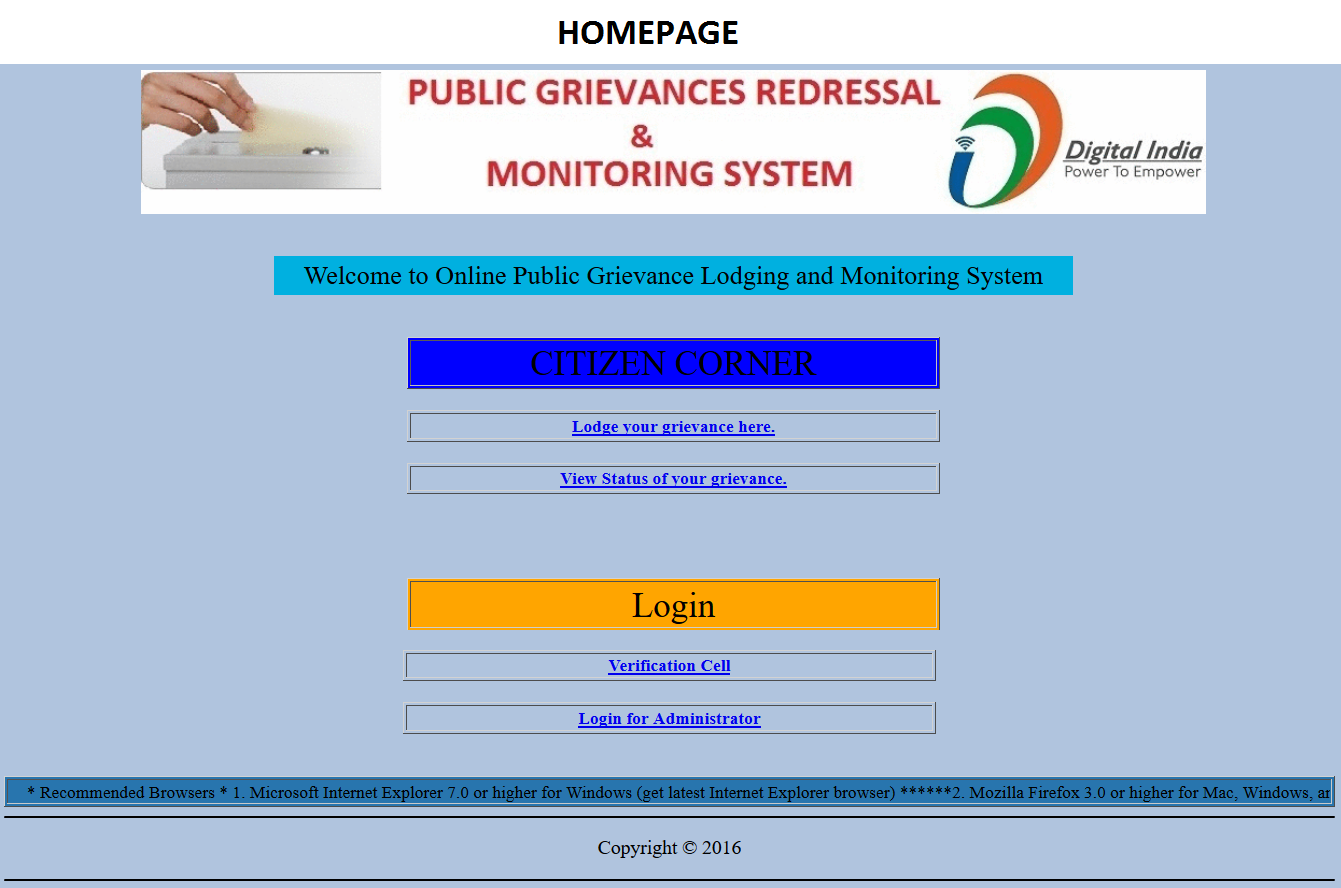
**E.R. Diagram**

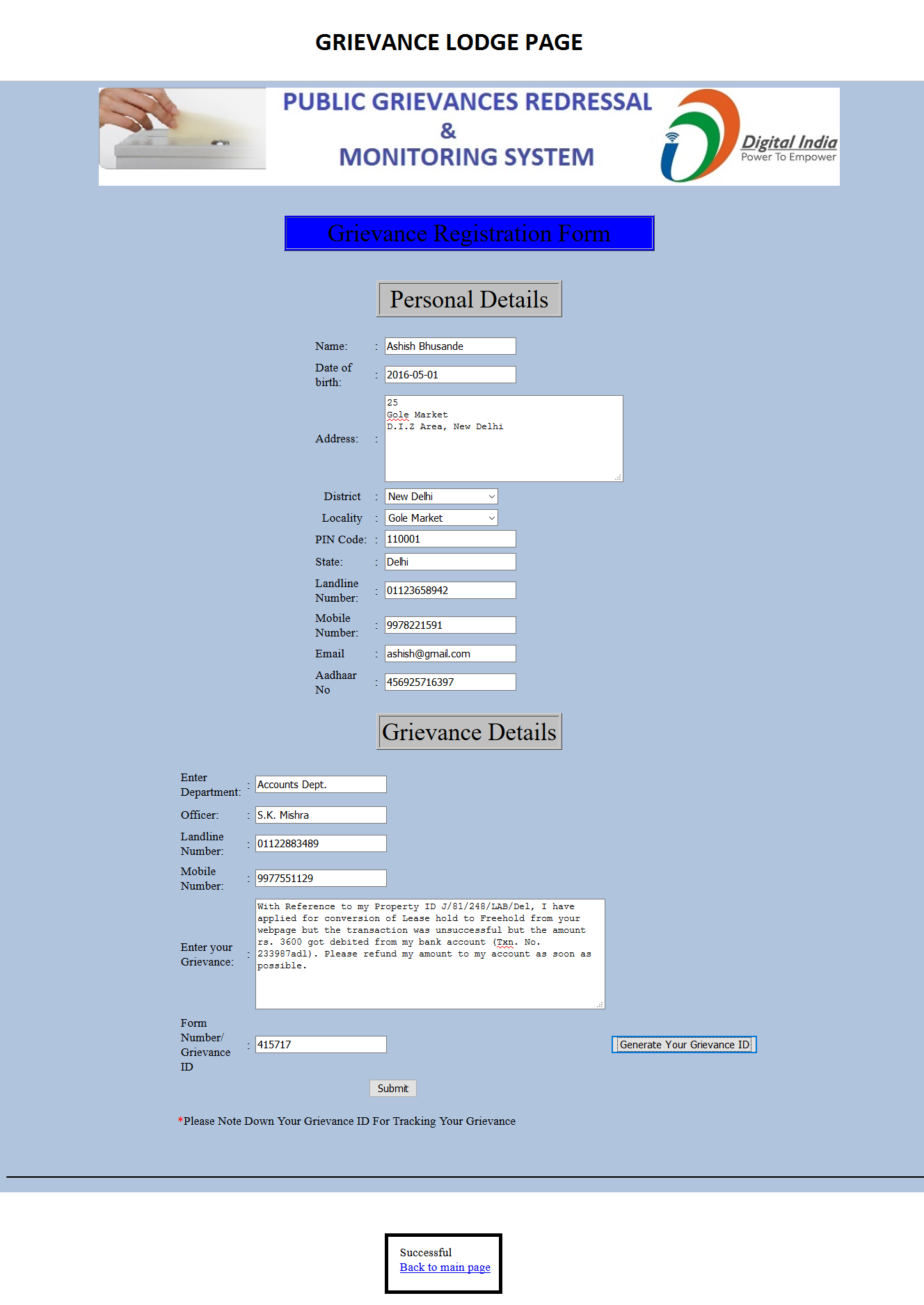


CHAPTER 5

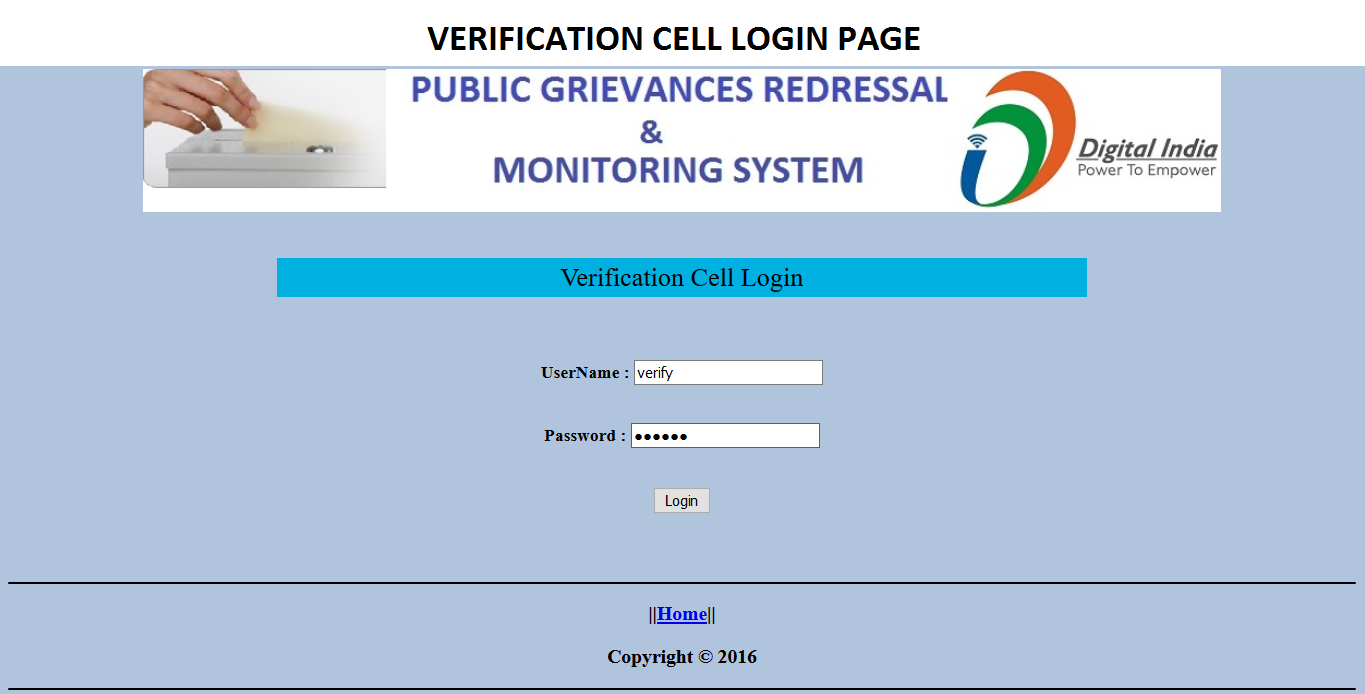
**SNAPSHOTS OF OUTPUTS**

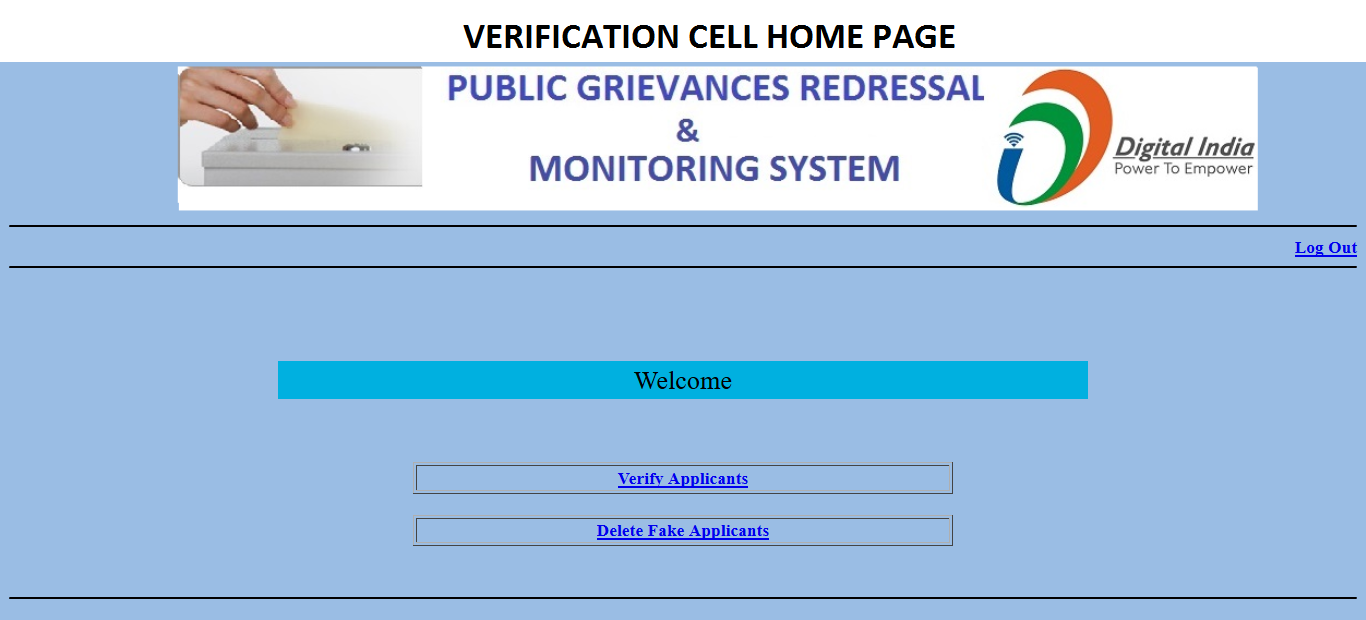
**5.1 Screen Shots**

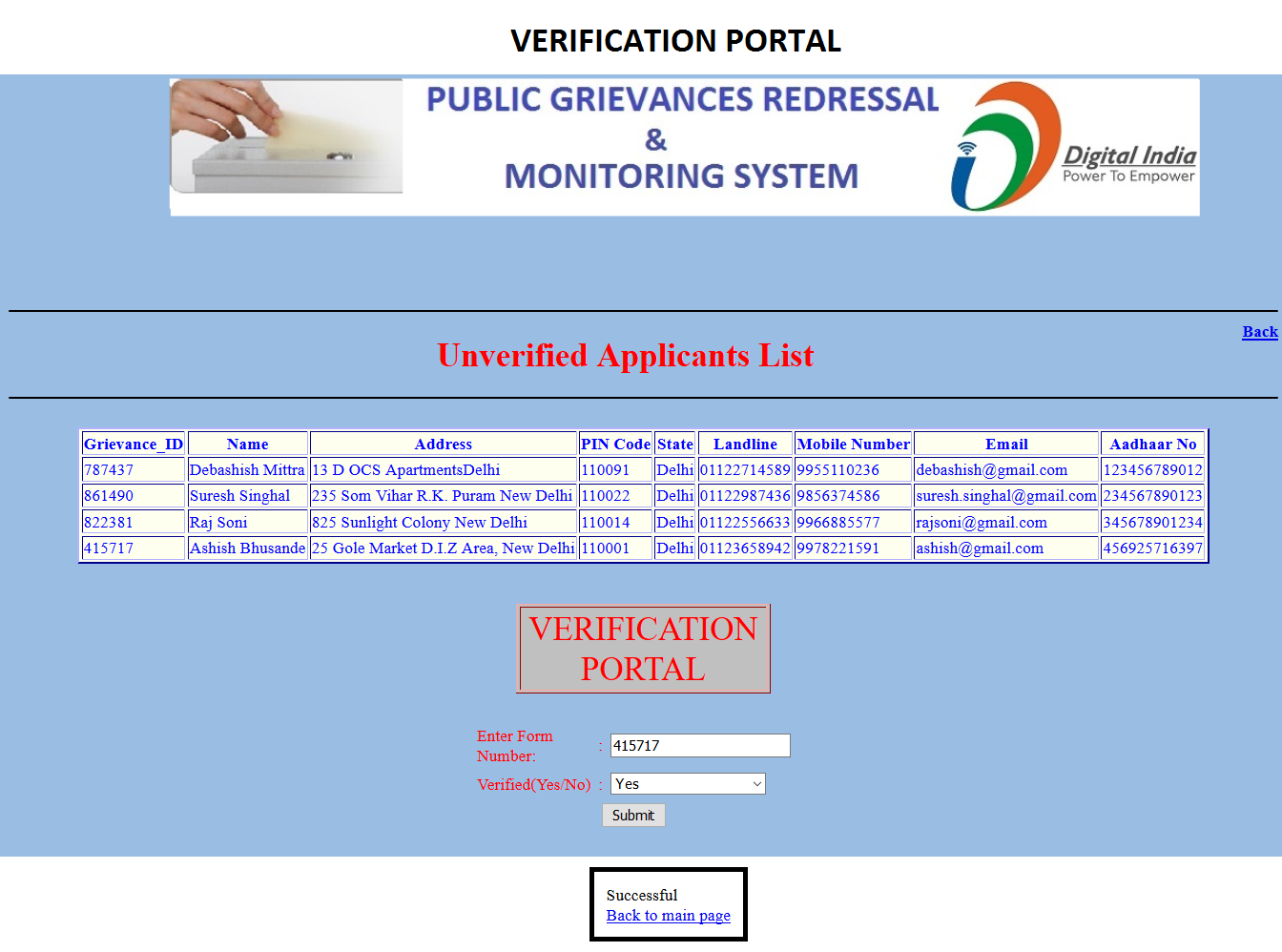
****

****

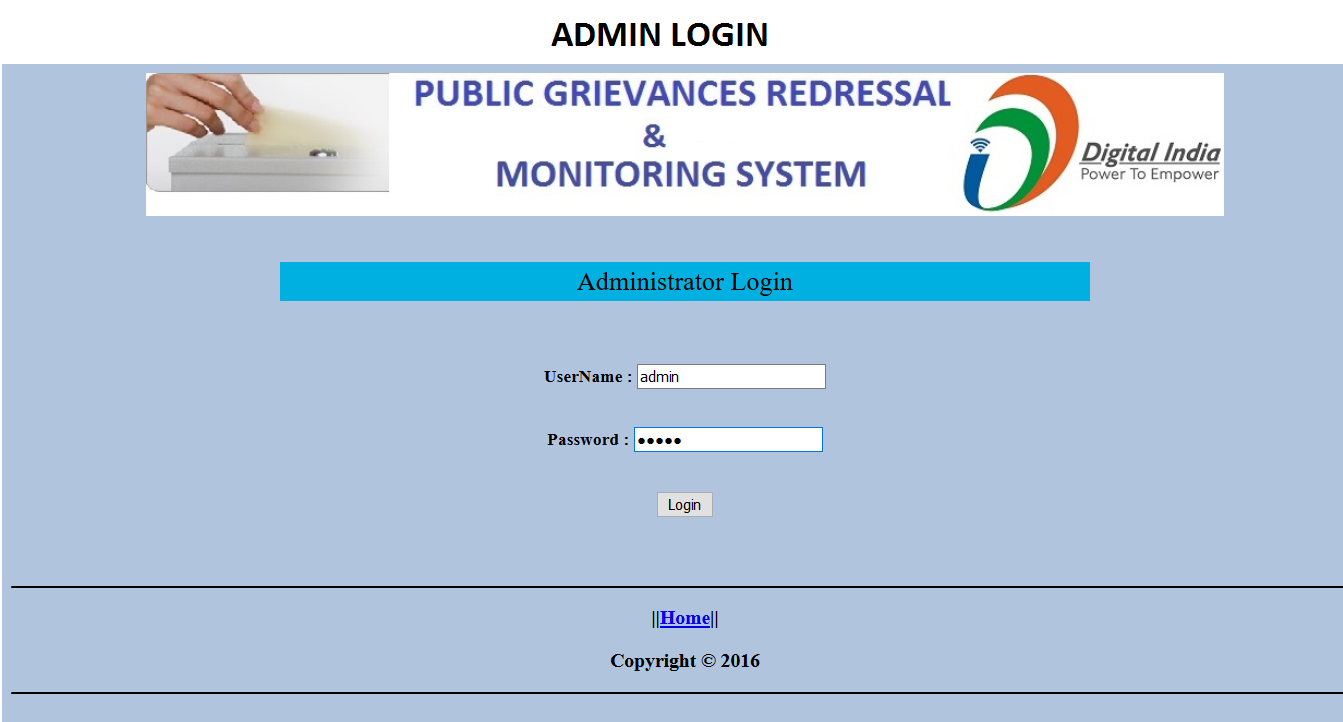
****

****

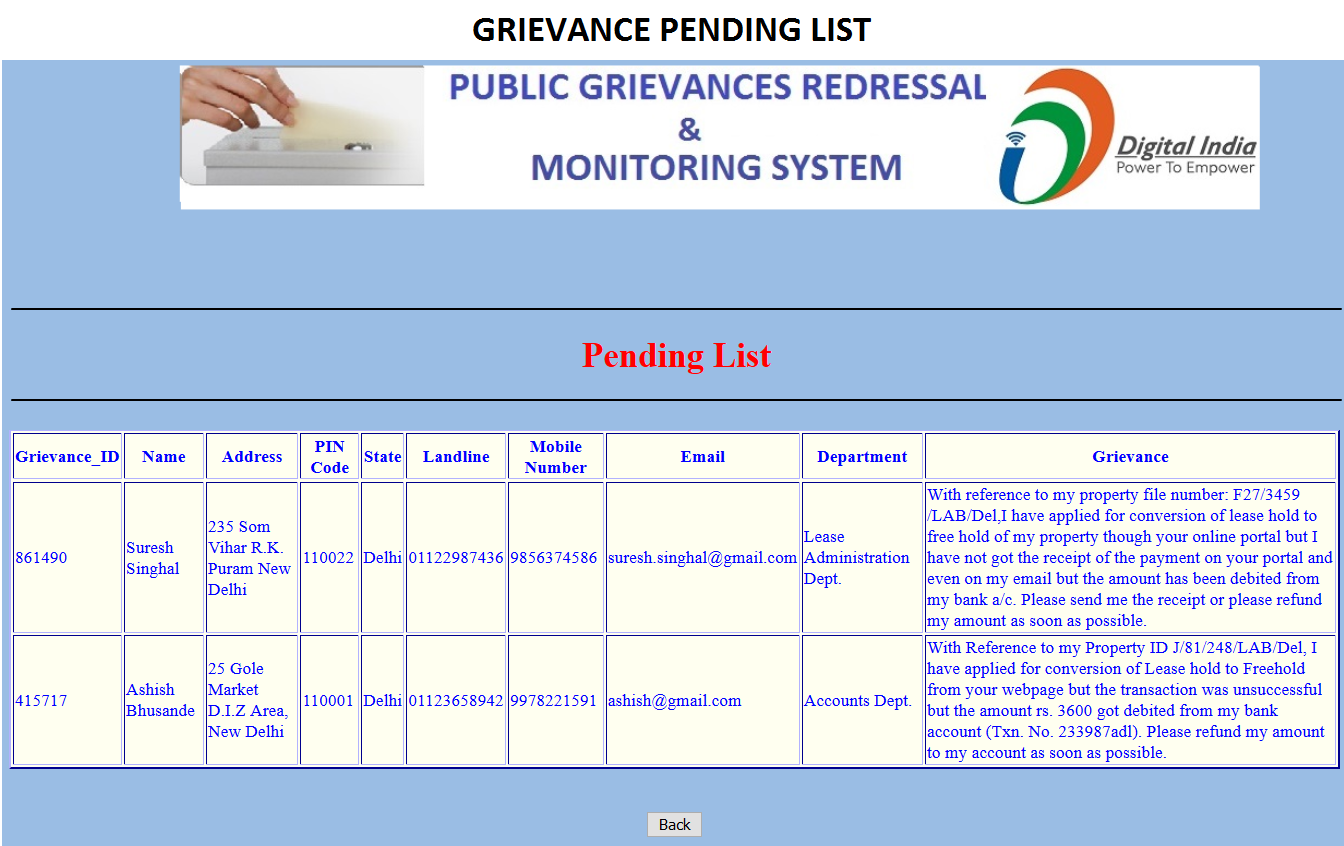
****

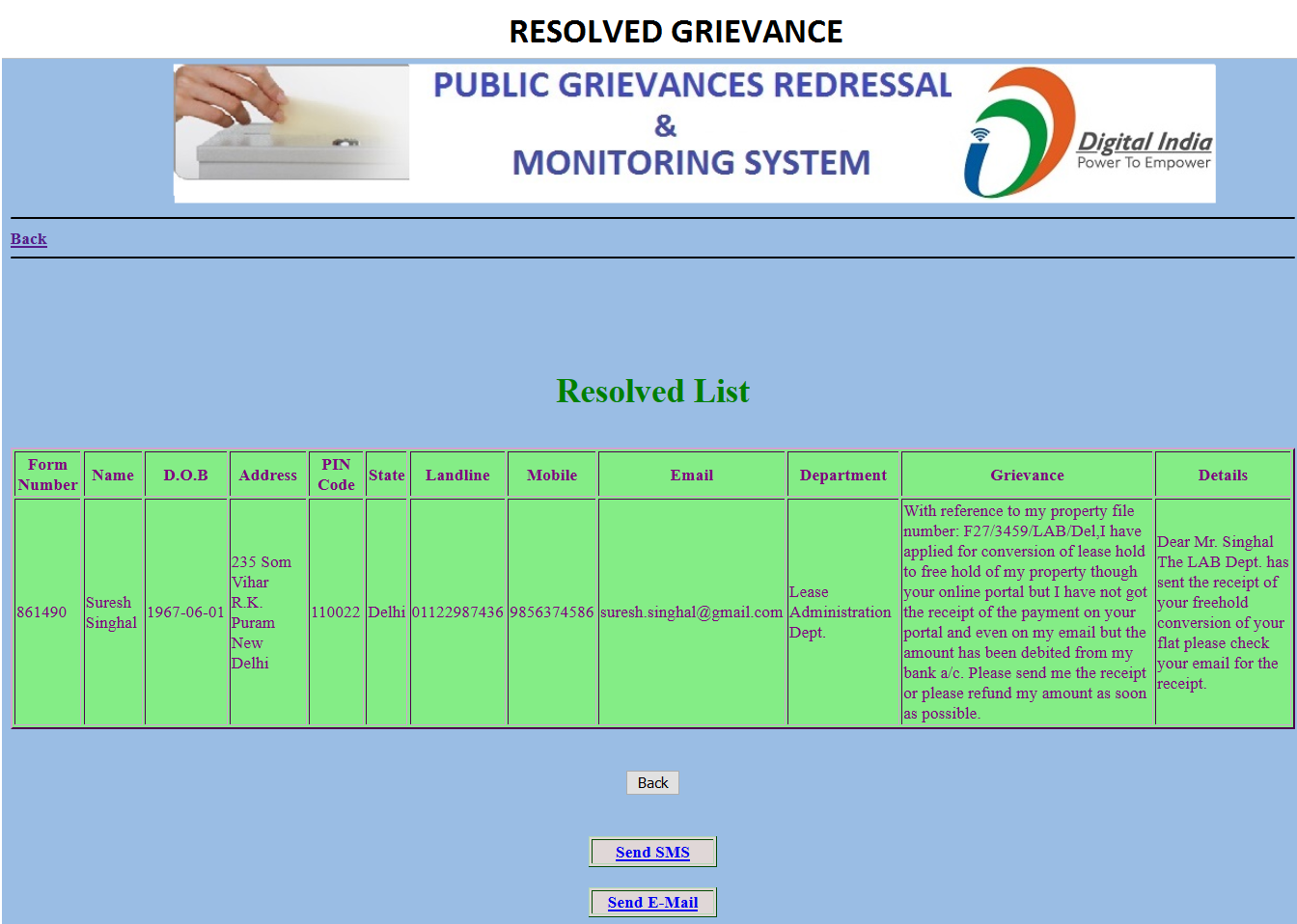
****

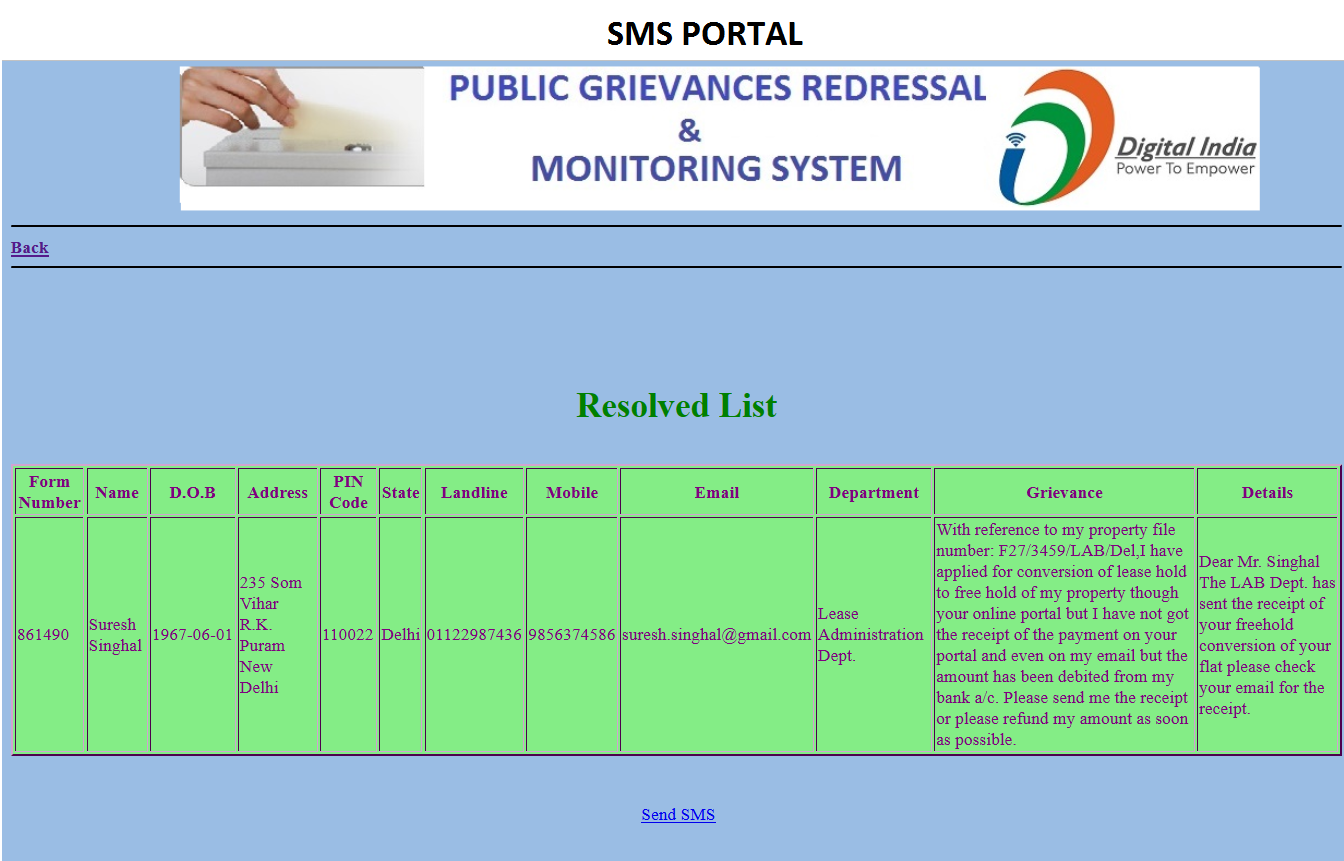
****

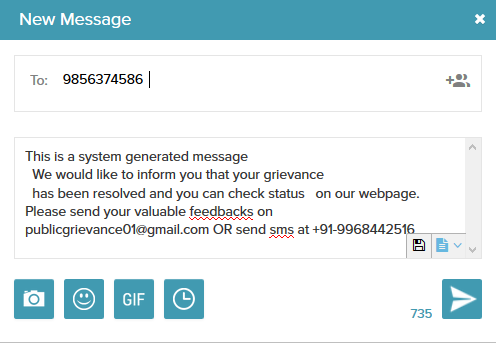
****

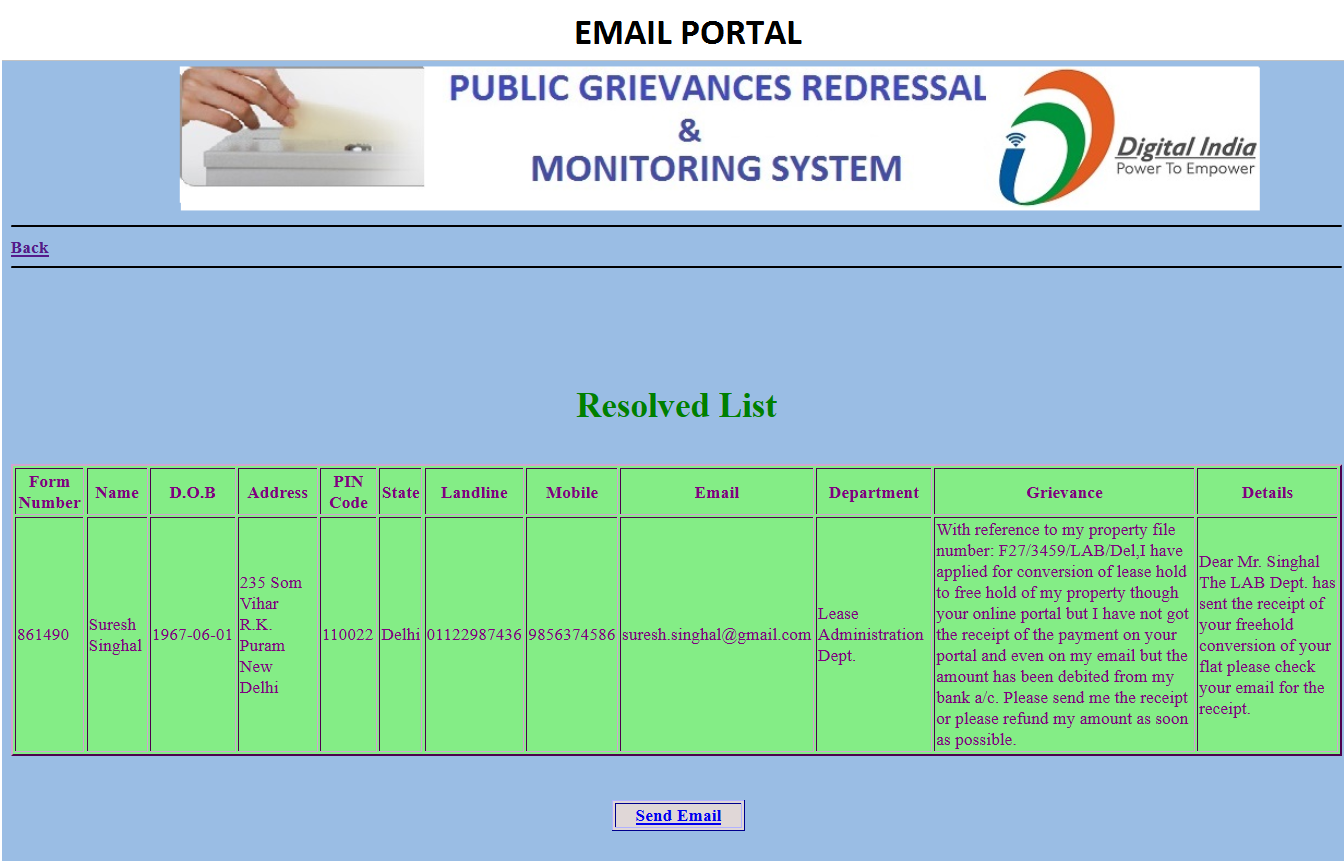
****

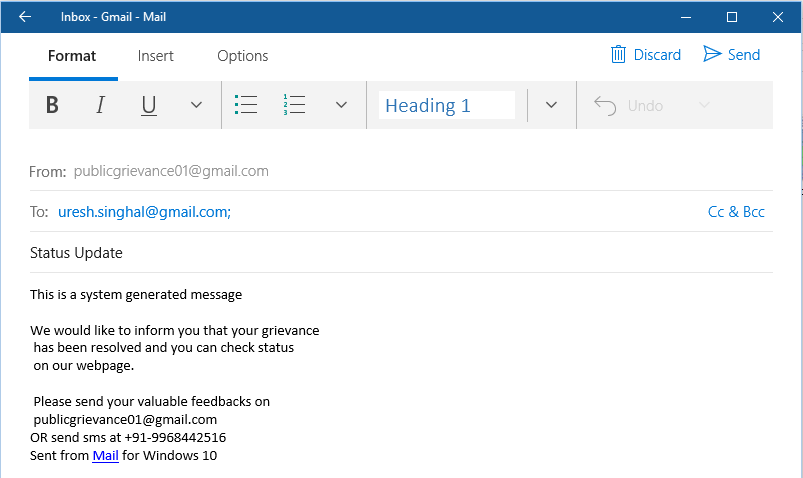
****

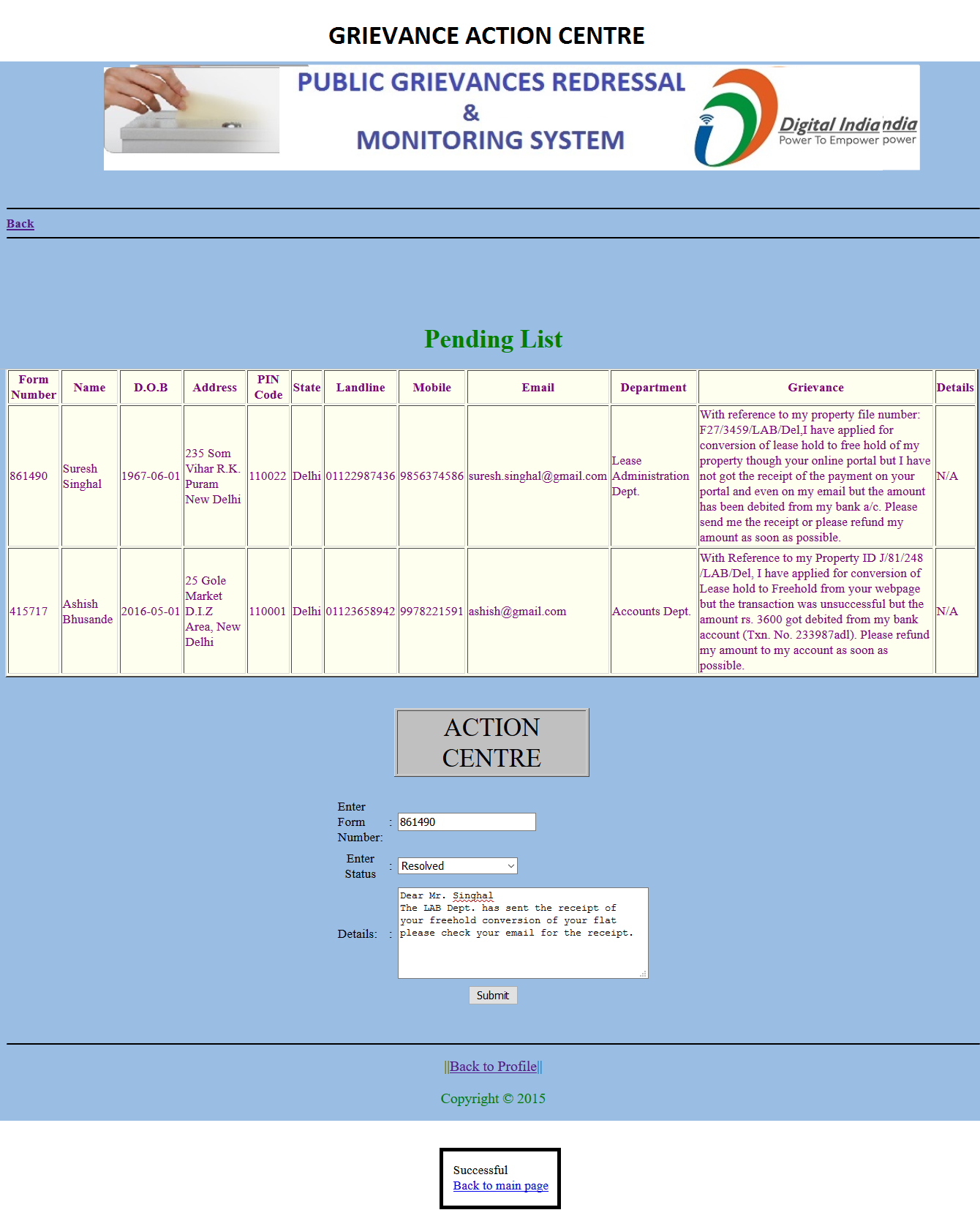
****

****

****

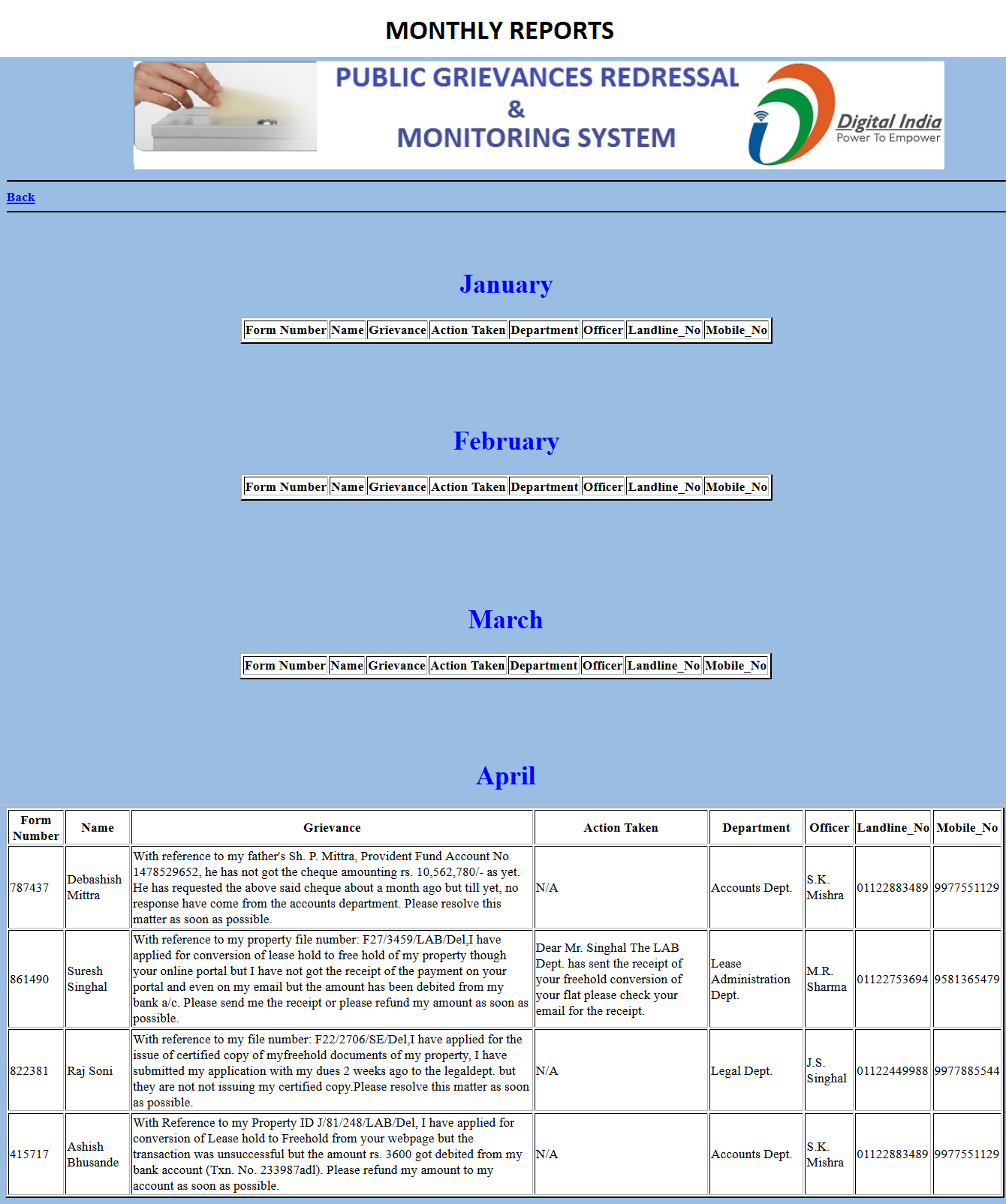
****

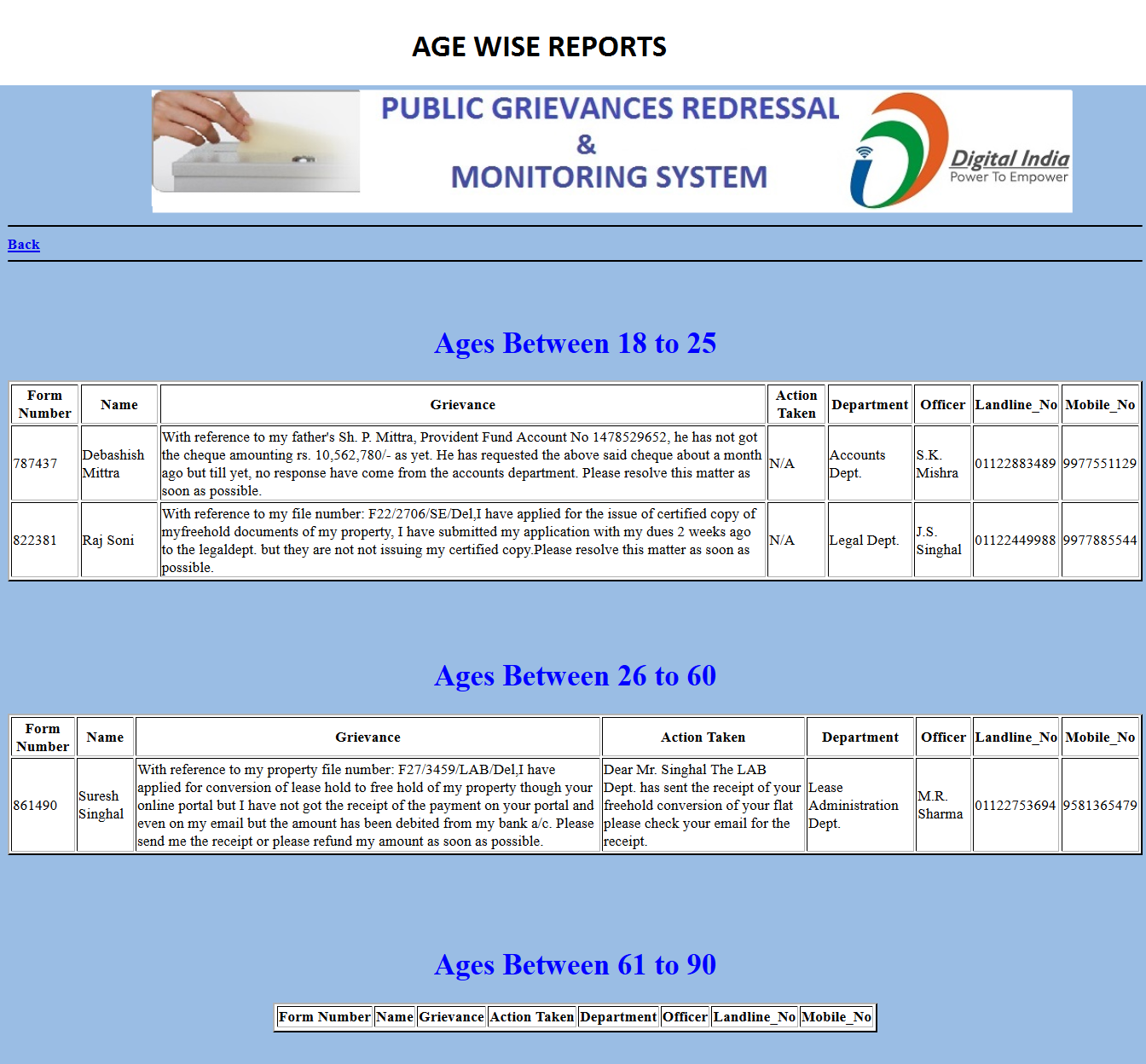
****

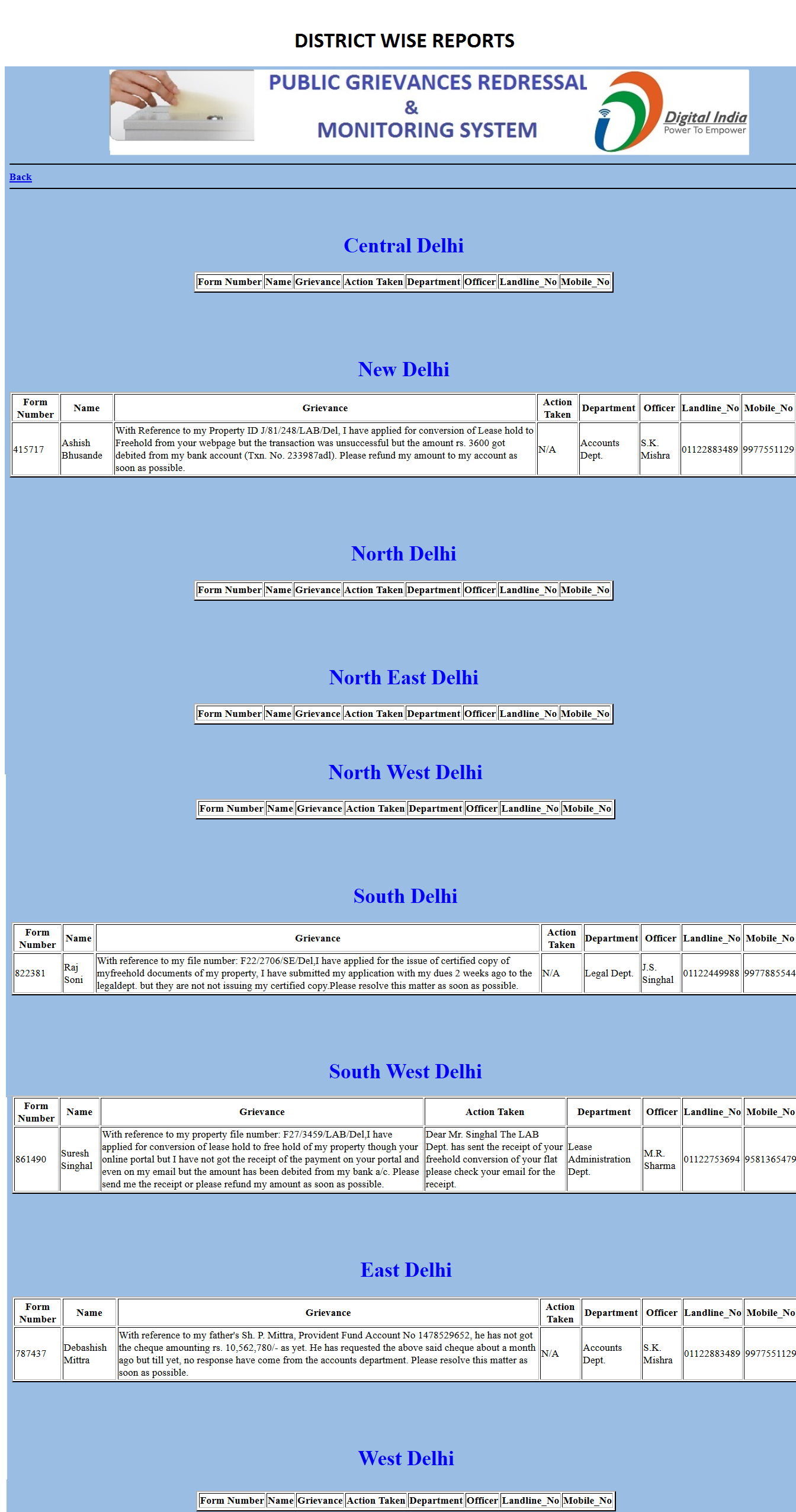


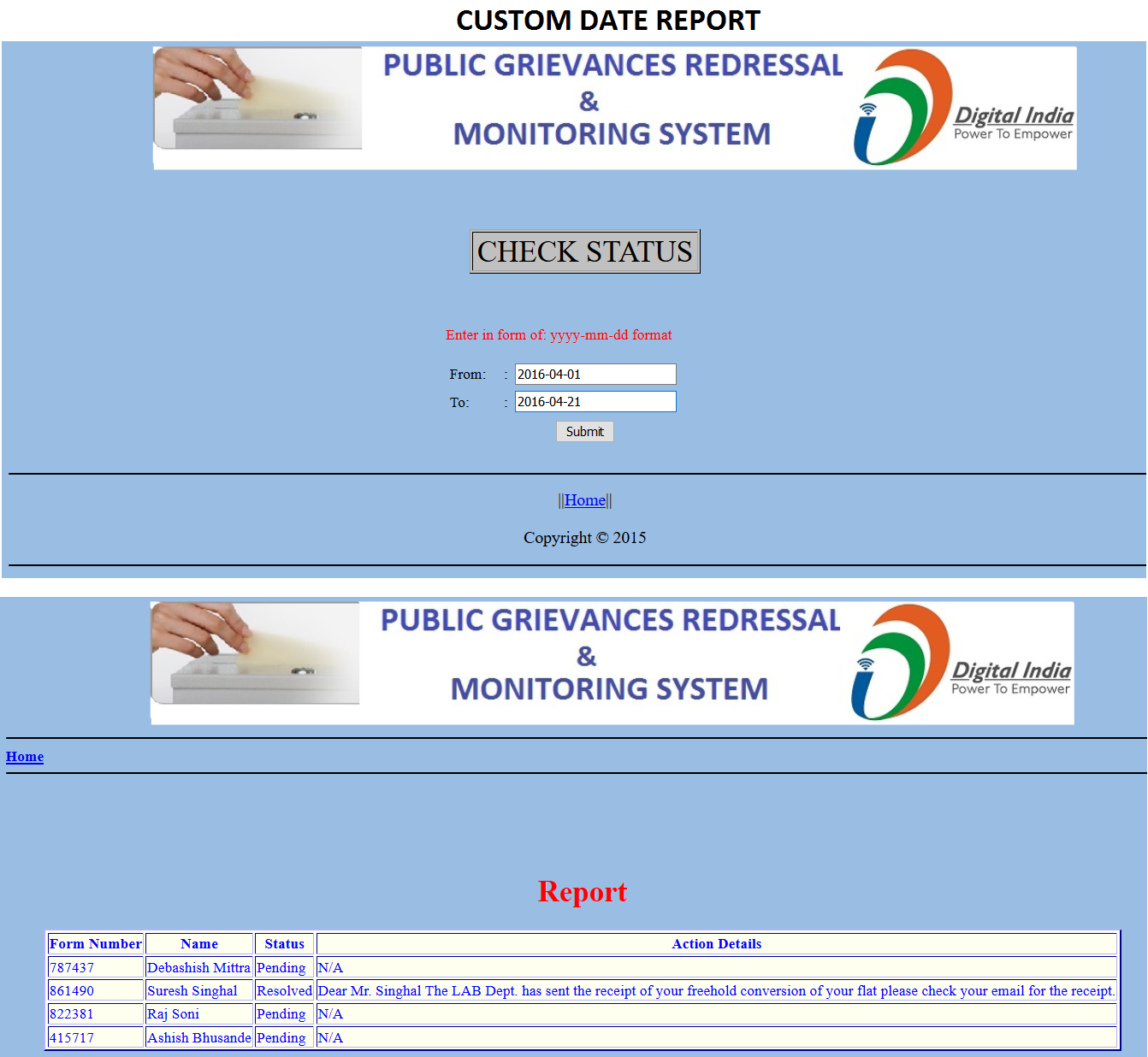
**REPORTS**

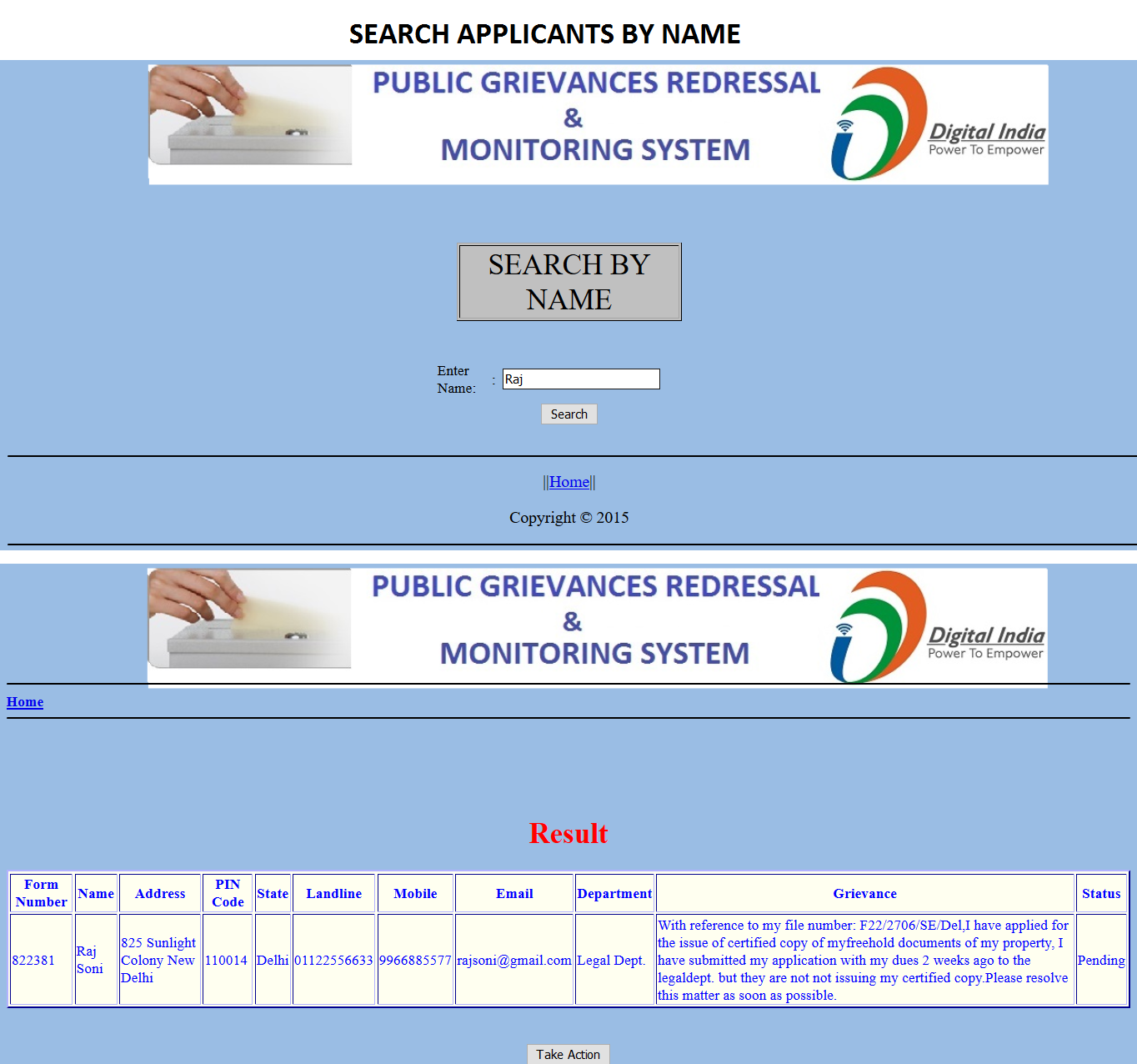




****

****

****

****

CHAPTER 6

**SYSTEM TESTING AND IMPLEMENTATION**

**6.1 TESTING**

The testing and implementation they are important and final phases. All the process that has been done is just a trail or by assumption. All the required hardware & software is prepared for the testing so that some errors or some modifications may be required for further proceeding.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct. The goal will be successfully achieved. There are four steps within, they are,

* Unit Testing
* Integration Testing
* Component Interface Checking
* System Testing
* Operational Acceptance Testing

**6.1.1 UNIT TESTING**

In [computer programming](https://en.wikipedia.org/wiki/Computer_programming), unit testing is a [software testing](https://en.wikipedia.org/wiki/Software_testing) method by which individual units of [source code](https://en.wikipedia.org/wiki/Source_code), sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In [procedural programming](https://en.wikipedia.org/wiki/Procedural_programming), a unit could be an entire module, but it is more commonly an individual function or procedure. In [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming), a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by [white box testers](https://en.wikipedia.org/wiki/White-box_testing) during the development process. It forms the basis for component testing.

Ideally, each [test case](https://en.wikipedia.org/wiki/Test_case) is independent from the others. Substitutes such as [method stubs](https://en.wikipedia.org/wiki/Method_stub), [mock objects](https://en.wikipedia.org/wiki/Mock_object), [fakes](https://en.wikipedia.org/wiki/Mock_object#Mocks.2C_fakes_and_stubs), and [test harnesses](https://en.wikipedia.org/wiki/Test_harness) can be used to assist testing a module in isolation. Unit tests are typically written and run by [software developers](https://en.wikipedia.org/wiki/Software_developer) to ensure that code meets its design and behaves as intended.

**6.1.2 INTEGRATION TESTING**

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in [software testing](https://en.wikipedia.org/wiki/Software_testing) in which individual software modules are combined and tested as a group. It occurs after [unit testing](https://en.wikipedia.org/wiki/Unit_testing) and before [validation testing](https://en.wikipedia.org/wiki/Verification_and_validation_%28software%29). Integration testing takes as its input [modules](https://en.wikipedia.org/wiki/Module_%28programming%29) that have been [unit tested](https://en.wikipedia.org/wiki/Unit_testing), groups them in larger aggregates, applies tests defined in an integration [test plan](https://en.wikipedia.org/wiki/Test_plan) to those aggregates, and delivers as its output the integrated system ready for [system testing](https://en.wikipedia.org/wiki/System_testing).

The purpose of integration testing is to verify functional, performance, and reliability [requirements](https://en.wikipedia.org/wiki/Requirement) placed on major design items. These "design items", i.e. assemblages (or groups of units), are exercised through their interfaces using [black box testing](https://en.wikipedia.org/wiki/Black_box_testing), success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and [inter-process communication](https://en.wikipedia.org/wiki/Inter-process_communication) is tested and individual [subsystems](https://en.wikipedia.org/wiki/Subsystem) are exercised through their input interface. [Test cases](https://en.wikipedia.org/wiki/Test_case) are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e. unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.

Some different types of integration testing are big bang, [top-down, and bottom-up](https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design). Other Integration Patterns are: Collaboration Integration, Backbone Integration, Layer Integration, Client/Server Integration, Distributed Services Integration and High-frequency Integration.

**6.1.3 COMPONENT INTERFACE CHECKING**

The practice of component interface testing can be used to check the handling of data passed between various units, or subsystem components, beyond full integration testing between those units. The data being passed can be considered as "message packets" and the range or data types can be checked, for data generated from one unit, and tested for validity before being passed into another unit. One option for interface testing is to keep a separate log file of data items being passed, often with a timestamp logged to allow analysis of thousands of cases of data passed between units for days or weeks. Tests can include checking the handling of some extreme data values while other interface variables are passed as normal values. Unusual data values in an interface can help explain unexpected performance in the next unit. Component interface testing is a variation of [black-box testing](https://en.wikipedia.org/wiki/Black-box_testing), with the focus on the data values beyond just the related actions of a subsystem component.

**6.1.4 SYSTEM TESTING**

**System testing** of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified [requirements](https://en.wikipedia.org/wiki/Requirements). System testing falls within the scope of [black-box testing](https://en.wikipedia.org/wiki/Black-box_testing), and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all of the "integrated" software components that have passed [integration testing](https://en.wikipedia.org/wiki/Integration_testing) and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called *assemblages*) or between any of the *assemblages* and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

**6.1.5 OPERATIONAL ACCEPTANCE TESTING**

Operational Acceptance Testing (OAT) is used to conduct operational readiness (pre-release) of a product, service or system as part of a [quality management system](https://en.wikipedia.org/wiki/Quality_management_system). OAT is a common type of non-functional [software testing](https://en.wikipedia.org/wiki/Software_testing), used mainly in [software development](https://en.wikipedia.org/wiki/Software_development) and [software maintenance](https://en.wikipedia.org/wiki/Software_maintenance) projects. This type of testing focuses on the [operational readiness](https://en.wikipedia.org/w/index.php?title=Operational_readiness&action=edit&redlink=1) of the system to be supported, and/or to become part of the production environment. Hence, it is also known as operational readiness testing (ORT) or [operations readiness and assurance](https://en.wikipedia.org/wiki/Operations_readiness_and_assurance) (OR&A) testing. [Functional testing](https://en.wikipedia.org/wiki/Functional_testing) within OAT is limited to those tests which are required to verify the *non-functional* aspects of the system.

During OAT changes may be made to environmental parameters which the application uses to run smoothly. For example, with [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) applications with a mixed or hybrid architecture, this may include: [Windows services](https://en.wikipedia.org/wiki/Windows_service), [configuration files](https://en.wikipedia.org/wiki/Configuration_file), [web services](https://en.wikipedia.org/wiki/Web_services), [XML](https://en.wikipedia.org/wiki/XML) files, [COM+](https://en.wikipedia.org/wiki/COM%2B) components, web services, [IIS](https://en.wikipedia.org/wiki/Internet_Information_Services), stored procedures in databases, etc. Typically OAT should occur after each main phase of the development life cycle: design, build, and [functional testing](https://en.wikipedia.org/wiki/Functional_testing). In sequential projects it is often viewed as a *final* verification before a system is released; where in agile and iterative projects, a more frequent execution of OAT occurs providing stakeholders with assurance of continued stability of the system and its operating environment.

**6.2 SYSTEM IMPLEMENTATION**

* + - Training the operating staff
    - Installing hardware
    - Installing terminals
    - Installing telecommunication network before system is up and running.

In the implementation phase**,** the project reached its fruition. After the development phase of the SDLC is complete, the system is implemented. The software, which was designed in design and programmed in development phase of the SDLC, was installed on all the PCs that require it. The persona’s using the program was trained during this phase of the SDLC. Moreover, both the hardware and software are tested. Although we found and fixed many problems, almost invariably, the user’s helped us to uncover problems that we were unable to simulate.

These were the main activities performed by us in the course of the project, which lead to its proper completion.

**6.3 POST IMPLEMENTATON REVEIW**

When computer based systems are built therefore we must develop mechanism for evaluating controlling and making modifications, maintenance issued to improve the case with which the changes can be accommodated and reduce the amount of expended on its maintenance activity occurs because it is unreasonable to assume that software testing will uncover all latest errors in a large software system. The final event in the post implementation flow is review that revalidates all elements of the system configuration and ensures correctness, after the software maintenance, software reviews is being conducted for future maintenance effort and provides feedback, which is important to effectively management of software organization.

**Implementation Tasks**

The major implementation tasks consist of:

* Planning the implementation activities
* Acquiring and laying out facilities and office
* Organizing the personnel for implementation
* Developing procedures for installation and testing
* Developing the training program for operation personnel
* Acquiring required hardware
* Generating files
* Designing forms
* Testing the entire system
* Completing cutover to the new system
* Documenting the system
* Evaluating the MIS
* Providing system maintenance

CHAPTER 7

**SCOPE AND CONCLUSION**

**7.1 FUTURE SCOPE**

Completion of the development process will result in a software package that will provide user-friendly environment, which is very easy to work with, even for people with very little knowledge of computer.

Management of various tasks is incorporated in the package and will deliver the required information in a very easy to use and easy to access manner.

This package will provide accuracy, efficiency, speed and easiness to the end user. Since the system is verified with valid as well as invalid data and is run with an insight into the necessary modifications that may require in the future, it can be maintained successfully without any problems.

This Portal gives Option to Users to lodge their grievance at anytime and anywhere part of this world. At this time every users have shortage of time so Users can lodge their grievance online this helps to save traveling time, fuel, money and other kinds of resources. This project can have so many options to selling and purchasing.

**7.2 LIMITATION**

* Due to unavailability of templates it is impossible to generate different types of framework; here we need to develop them by our own.
* Deployment of the Project on Pocket-PC i.e. Mobile Application.

**7.3 CONCLUSION**

There was a lot of fun in making this project. This project was very useful to us as it provided us the inside view of the planning and implementation of the data base. In this project we had to think about the various options which we can provide to user. The implementation was not easy as we had to look into the minute details in order to achieve my goals. We have tried to make this project user friendly and also interactive by providing many features.

I am satisfied by achieving the goals for which I had planned a lot. A lot of experimental work can be done with this project. I’m looking forward for any advice which can help us to improve the project.

CHAPTER 8

**REFERENCES**

Various sites were referred during making of the project are as follows:

* [www.en.wikipedia.org](http://www.en.wikipedia.org)
* [www.google.com](http://www.google.com)
* <http://www.php.net>
* [www.roseindia.net](http://www.roseindia.net)
* [www.w3cschools.com](http://www.w3cschools.com)
* <http://www.phpreferencebook.com/>

Various books were being referred to for PHP, HTML, XML, CSS & DHTML clarification and documentations as follows:

* PHP the complete reference by Steven Holzner
* [Beginning Web Programming with HTML, XHTML, and CSS](https://books.google.co.in/books?id=cCYDj6vEReAC&printsec=frontcover&dq=html+books&hl=en&sa=X&ved=0CCkQ6AEwAmoVChMI0Nmli-WVyQIVQ1qOCh04pQgj) by Jon Duckett.
* Pro MySQL by [Jay Pipes](https://www.google.co.in/search?biw=1366&bih=657&tbm=bks&tbm=bks&q=inauthor:%22Jay+Pipes%22&sa=X&ved=0CC4Q9AgwAmoVChMI_OvgrOWVyQIVQQeOCh24NA4Y), ‎[Michael Kruckenberg.](https://www.google.co.in/search?biw=1366&bih=657&tbm=bks&tbm=bks&q=inauthor:%22Michael+Kruckenberg%22&sa=X&ved=0CC8Q9AgwAmoVChMI_OvgrOWVyQIVQQeOCh24NA4Y)

The project idea was given to me by Mr. V.S. Tomar, Systems Director of Delhi Development Authority, and he motivated me a lot to shape up this project up to this extend.