**TABLE OF CONTENTS**

**CHAPTER TITLE NO** **PAGE NO**

**1 ABSTRACT**

**2 TABLE OF CONTENTS**

**3**  **LIST OF FIGURES**

**4 LIST OF ABBREVIATION**

**1 INTRODUCTION 1 2 EXISTING SYSTEM**

**2.1 DISADVANTAGES 3 PROPOSED SYSTEM 3.1 ADVANTAGE**

**4 MODULES 4.1 USER REGISTRATION 4.2 USER LOGIN 5 4.3 ADMINISTRATOR 4.4 OFFICER 5 SYSTEM REQUIREMENT 5.1 HARDWARE REQUIREMENT 5.2 SOFTWARE REQUIREMENT 6 TESTING 6.1 UNIT TESTING**

**6.2 VALIDATION TESTING 6.3 DATA CENTRIC TESTING 6.4 INTEGRATION TESTING 7 TEST CASES 15 7.1 TEST CASE FOR LOGIN 7.2 TEST CASE FOR REGISTRATION**

**7.3 TEST CASE FOR COMPLAINT**

**7.4 TEST CASE FOR OFFICER**

**8 SOURCE CODE 8.1 BACK END**

**8.2 FRONT END 9 RESULT 10 CONCLUSION&FUTURE ENHANCEMENT 51**

**FIGURE NO FIGURE NAME PAGE NO 1 PROPOSED SYSTEM ARCHITECTURE 6 2 ARCHITECTURAL DIAGRAM OF ADMIN 7 3 ARCHITECTURAL DIAGRAM OF OFFICER 8 4 ARCHITECTURAL DIAGRAM OF USER 9 5 ARCHITECTURE OF JSP 11 6 HOME PAGE 34 7 ABOUT US PAGE 35 8 ADMIN LOGIN PAGE 36 9 ADMIN PAGE 37 10 OFFICER LOGIN PAGE 38 11 OFFICER PAGE 39 12 REGISTRATION PAGE 40 13 USER LOGIN 41 14 USER PAGE 42 15 ADD PANCHAYAT 43 16 VIEW PANCHAYAT PAGE 44 17 COMPLAINT PAGE 45 18 SOLVE COMPLAINT PAGE 46 19 VIEW COMPLAINT REPORT PAGE 47 20 USER INVALID PAGE 48 21 CONTACT US 49 22 SCHEME REPORT 50**

**LIST OF ABBREVIATION**

**JSP : Java server page**

**HTML : Hyper text Markup**

**XML : Extensible Markup Language**

**ABSTRACT**

**This project provides solution relating to issue in Power, Water, Health care, etc. It is useful for village people and various problems (road, water, school) related to people. They have to first post their complaint by using the website then the system will automatically send an email to taluk officer. All the complaint details will be stored to the database and later a message will be sent to the user, if the particular actions were taken. The “Smart Village” concept aims to realize its goal through providing good service on village people and the challenges of village development. Initiative focuses on improved resource-use efficiencies, empowered local self-governance, access to assured basic amenities and responsible for individual and community behavior to build a vibrant and happy society.**

**CHAPTER 1**

**INTRODUCTION**

* 1. **COMPANY PROFILE**

Sandfits Technologies is a software development company established in 2005. The company is one of the top Software and website development company in Chennai and most creative and ingenious among our peers, in this industry. It provides the latest technology and most innovative solutions, allowing us to give the customers the best service possible. It believes in working with the client, rather than for the client. The company also develop web-sites and web based applications, the area of expertise includes Web Design & Maintenance, Flash based Websites, Web application development, 2D-3D animation, Mobile Phone application development, Domain Name Registration, Search Engine Optimization (SEO). The key technological expertise include, but is not limited to, Database Administration(DBA), Dot Net, PHP, C#, DHTML, CSS, Flash, Photoshop, JAVA & Android.

Sandfits Compant location . Sf No. 191/2, Salakarichal road, appanaickenpatti post Coimbatore , Tamilnadu 641402

**1.2 OBJECTIVE**

**The “Smart Village” concept provides services such as Power, Water, Buildings, Retail, Health care, etc. It allows the village person posting the complaint by the website the system automatically sends an email to taluk officer. All the complaint details will be stored to the database and all complaint projects complete sending a message to the user. New designs, technologies and management models should be used to upgrade the existing ones and in building the new ones. Requires strategy, integrated planning and above all monitoring and execution of the activities using appropriate governance models.**

**The " Smart Village concept” provide local needs and Analysis of the villages on various economic parameters at a micro as well as macro level. Improving the literacy rate of the villages by reducing the dropout rate. The “Smart Village " concept aims to realize its goal through providing policymakers with insightful, bottom–up analyses of the challenges of village development. Initiative focuses on improved resource-use efficiencies, empowered local self-governance, access to assured basic amenities and responsible individual and community behavior to build a vibrant and happy society.**

**CHAPTER 2**

**SYSTEM ANALYSIS**

**2**.**1 EXISTING SYSTEM**

Many researchers believe that the existing technologies developed for the smart city may be useful for the smart village concept. If the Village people have any social problem (road, water, food ect…..). They need to go taluk office in person. The existing system is implemented as manual system. The village people find officer though phone call and direct go to taluk office. All complaint stored in paper records. Researchers reported that the Smart village system can be developed on the lines of smart city model.

**Drawbacks**

* One of the disadvantages of an ethical compliance program is that it requires the comprehensive support of management to be effective.
* If members of the management team decide to apply their own version of corporate ethics to the way they manage their departments, then this clash of principles can cause confusion in the workplace.
* Developing, implementing and maintaining an ethics compliance program within the organization can be expensive and time-consuming,

**2.2 PROPOSED SYSTEM**

Providing quality utility services like power, water, sanitation, and essential services such as education, healthcare, transportation, infrastructure (roads, railways, buildings, equipment) The primary strategy for the development of every village. Some of the utility services can be managed at a district level and others such as health care, schooling etc need to be managed at village level for proximity and accessibility reasons. Investment climate of the village is also impacted to a very large extent on the availability of the above mentioned utility and other services in the villages.

**Advantages**

* User can post their compliant efficiently through this system.
* This system helps to reach higher official to take action in problematic areas.
* Higher official can search a specific area to view compliant of the people directly.
* Identify a department and send a complaint to the department head.
* Efficient and Time Consuming.

**CHAPTER 3**

**SYSTEM SPECIFICATION**

**3.1 HARDWARE REQUIREMENT**

Processor : PIV

Ram : 512 Mb

Hard Disk : 10 GB Space

Monitor : VGA Color (256)

**3.2 SOFTWARE REQUIREMENT**

Operating System : Windows 2000 or Higher

Language : Java

Technologies : Netbeans

Backend : MySql Server

Back End Tool : SQL Yog

Web Server : Apache Tomcat

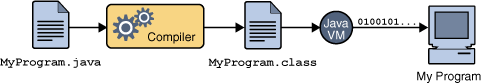
Build Tool : Apache Ant

**CHAPTER 4**

**SOFTWARE DESCRIPTION**

**4.1 FRONT END**

In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes — the machine language of the Java Virtual Machine (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

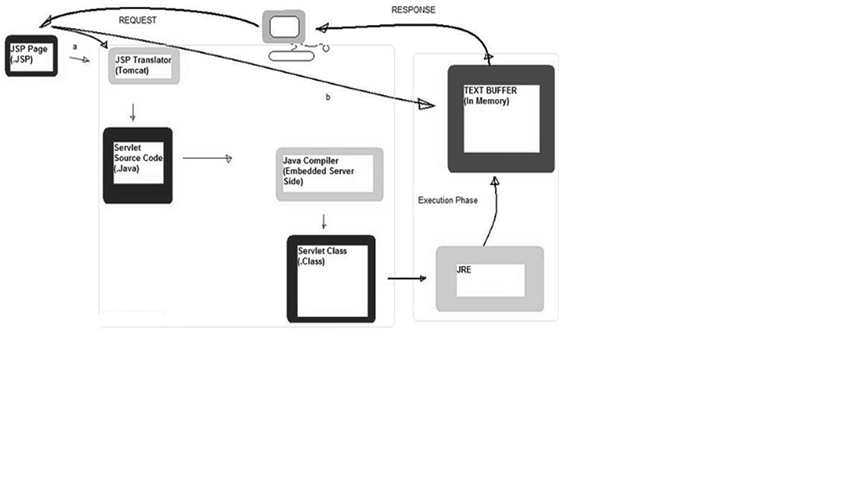


**Figure 4.1 software development process**

**JSP**

**Java Server Pages** (**JSP**) is a Java technology that allows software developers to dynamically generate HTML, XML or other types of documents in response to a Web client request. The technology allows Java code and certain pre-defined actions to be embedded into static content. The JSP syntax adds additional XML-like tags, called JSP actions, to be used to invoke built-in functionality. Additionally, the technology allows for the creation of JSP tag libraries that act as extensions to the standard HTML or XML tags. Tag libraries provide a platform independent way of extending the capabilities of a Web server. JSPs are compiled into Java Servlets by a JSP compiler. A JSP compiler may generate a servlet in Java code that is then compiled by the Java compiler, or it may generate byte code for the servlet directly. JSPs can also be interpreted on-the-fly reducing the time taken to reload changes Java Server Pages (JSP) technology provides a simplified, fast way to create dynamic web content. JSP technology enables rapid development of web-based applications that are server- and platform-independent.

**Architecture OF JSP**

 **Figure 4.2 Architecture of JSP**

**4.2 BACK END**

**MYSQL**

The MySQL Reference Manual covers most areas of MySQL use. This manual is for both MySQL Community Server and MySQL Enterprise Server. If you cannot find the answer(s) from the manual, you can get support by purchasing MySQL Enterprise, which provides comprehensive support and services. MySQL Enterprise also provides a comprehensive knowledge base library that includes hundreds of technical articles resolving difficult problems on popular database topics such as performance, replication, and migration.

MySQL AB develops and supports a family of high-performance, affordable database products. The company's flagship offering is 'MySQL Enterprise', a comprehensive set of production-tested software, proactive monitoring tools, and premium support services. MySQL is the world's most popular open source database software. Many of the world's largest and fastest-growing organizations use MySQL to save time and money powering their high-volume Web sites, business-critical systems and packaged software -- including industry leaders such as Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube and Booking.com. With headquarters in the United States and Sweden -- and operations around the world -- MySQL AB supports both open source values and corporate customers' needs.

**JDBC**

Java Database Connectivity (JDBC) is a programming framework for Java developers writing programs that access information stored in databases, spreadsheets, and flat files. JDBC is commonly used to connect a user program to a "behind the scenes" database, regardless of what database management software is used to control the database. In this way, JDBC is cross-platform . This article will provide an introduction and sample code that demonstrates database access from Java programs that use the classes of the JDBC API, which is available for free download from Sun's site .

**CHAPTER 5**

**PROJECT DESCRIPTION**

**5.1 PROBLEM DEFINITION**

* One of the disadvantages of an ethical compliance program is that it requires the comprehensive support of management to be effective.
* If members of the management team decide to apply their own version of corporate ethics to the way they manage their departments, then this clash of principles can cause confusion in the workplace.
* Developing, implementing and maintaining an ethics compliance program within your organization can be expensive and time-consuming.

**5.2 OVERVIEW OF THE PROJECT**

**PRODUCT PERSPECTIVE**

This website can be used by anyone those who have account. The users can post the complaints and it gives quick action to the problematic area. The authority can manage and solve the problematic area in authority process. The data can be stored in the database and retrieved later for future reference.

Some of the service chains like the Water, Power, and Health care’s are standard and could be part of larger services. Affordable housing, Retail, Education, and Skill based training, rural employment, Farm to market could be specific to the Village Mapping each of these and identifying strategic players and coordinating their actions could be critical for success. These services could redesign to be smart using cloud, ICT and data analytics. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant.

**PRODUCT FEATURES**

The smart village concept on make the village markets will provide important sources of future growth for many industries. One such village which has provided better quality of life to its village people. This web application is used to complaint post. Complaint details stores in the database as well as send email to the inbox. Reference management carried by the administrator.

The features of the project are analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

**DESIGN AND IMPLEMENTATION CONSTRAINTS**

* This application sends the details of complaint and solve through e-mail.
* This application acts like a village people coordinator and provide comfortable arrangement system
* This application automatically takes reference request and create reference portal

**5.3 MODULE DESCRIPTION**

Administrator manages all information and has access rights to view and solve the data related to places. This module provides information related to various complaints. User first register and post the complaint and later can user login and view the complaint

The module maintains the details of all reservations made so far and allows administrator to either confirm or reject the complaint. Reference module is the government sector of the website. It provides an interface to register new client. After registration admin should accept the request then provide a coupon to the agent.

**There are four modules namely**

* User Registration
* Public Module
* Authority Module
* Administrator

**http://localhost:8084/SmartVillage/**

[**http://localhost/svd/index.html**](http://localhost/svd/index.html)

**http://localhost:8084/SmartVillage/index.jsp**

**POST THE COMPLIANT**

**PI:**Message(name,aera…)

**Post the compliant**

**RI:** post compliant successfully

REQUIRED INTERFACE

**PI**:Logins(Username,Password)

**Officer**

**RI**:Logged Successfully

**PI:**Message(compliant)

**Solve the compliant**

**RI:** solve compliant successfully

REQUIRED INTERFACE

**PI**:Logins(Username,Password)

**User**

**RI**:Logged Successfully

**PI**:Logins(Username,Password)

**admin**

**RI**:Logged Successfully

* **Figure 5.2 Architectural Diagram of user**

**Figure 5.1 Architectural Diagram of smart village system**

**USER REGISTRATION**

User can register the details such as username, password, name, address, taluk name, district name, and mobile no, email id and post their complaints. Registered users can only post their complaints. Another users dost post their comlaints. Use registration forms to sign up users for subscription, and to post their complaints

**USER LOGIN**

User want to create the new account after the user login to the account and then the user can post compliant. The login credential of the user is permanent. Whetever the user can use that to login verified user can only post the compliant. After all login methods are invoked, the authentication is continued by invoke of the commit method on each login module.

**Add Complaint**: This module help the user to post online complaints.

**USER**

**OFFICER**

**PI** :Registers(name,address…)

**User registration**

**RI**:Registered Successfully

REQUIRED INTERFACE

**PI:**Message(name,aera…)

**Post the compliant**

**RI:** post compliant successfully

REQUIRED INTERFACE

**PI**:Logins(Username,Password)

**Login**

**RI**:Logged Successfully

**Figure 5.2 Architectural Diagram of user**

**AUTHORITY MODULE**

Municipal authorities can log-in to their accounts as created by administrator. Authorities can access all the complaints, suggestions from users. All the complaints with automatically create reports. The authority person can view the complaint and solves the problem.

**Mail**: This module helps the user to send mail to the authorities person.

**OFFICER**

**USER**

**PI**: Message(compliant)

**Views the compliant**

**RI**: Checks the compliant

**PI**:Message(compliant)

**Solve the compliant**

**RI**: solve successfully

**PI**: Login(username, password)

**Login**

**RI**: login successfully

**REQUIRED INTERFACE**

**Figure 5.3 Architectural Diagram of officer**

**ADMINISTRATOR**

To Create, and monitor accounts ofadministrator. This module helps the admin to add, delete and view authority person details.Administration Module to be a helpful tool in reviewing and managing the work of their users within the database. Create a user activity report for an individual user or for all users.

**ADMIN**

**OFFICER**

**PI**:Message(username,password)

**Add officer**

**RI**:Added successfullly

**PI**:Logins(username,password)

**Blocking the officer**

**RI**: Disables the officer

**PI** :Login(username,password)

**Login**

**RI**:Logged Successfully

**REQUIRED INTERFACE**

**Figure 5.3 Architectural Diagram of Admin**

**Figure 5.4 Architectural Diagram of Admin**

**5.4 DATA FLOW DIAGRAM**

A DFD is a structured analysis and design tool that can be used for flowcharting in place of or in association with information oriented and process oriented system flow charts. A DFD is a network that describes flow of data and the processes that change of transform data through the system.

Level 0

Admin

adminlogin

officer

**Figure 5.5 DFD diagram**

Level 1

User

registration

applyscheme

**Figure 5.6 DFD diagram**

Level 2

Login officer

officer

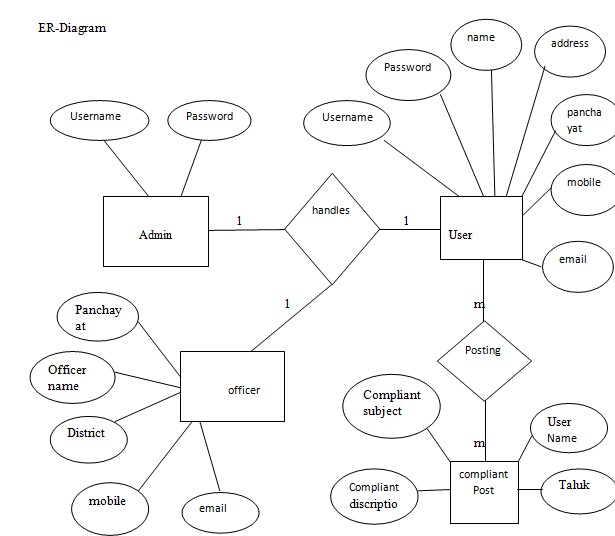
applyscheme

View report

**Figure 5.7 DFD diagram**

**5.5 ER DIAGRAM**

In software engineering, an entity–relationship model (ER model) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities (things) and the relationships that can exist among them.



**Figure 5.8 ER diagram**

**5.6 DATABASE DESIGN**

It is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a Data Definition Language, which then can be used to create a database. A fully attributed data model contains detailed attributes for each entity.

**T**ablename :adminlogin  
Purpose of table :This table is login admin  
primary key : username NOT NULL check(‘A-Z’,’a-z’,’0-9’)  
gin the password : password NOT NULL check(‘A-Z’,’a-z’,’0-9’)

**QUARY**

CREATE TABLE ` adminlogin ` (

`username` varchar(50) NOT NULL check (‘A-Z’,‘a-z’,’0-9’)

`password` varchar(50) NOT NULL check (‘A-Z’,‘a-z’,’0-9’) ,

) ENGINE=InnoDB DEFAULT

**Table 5.1 Admin Login**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Data type** | **Length** | **Description** |
| Username | Varchar | 10 | This field specifies the username  Eg:user123 |
| Password | Varchar | 10 | This field specifies the password  Eg:12345 |

**T**able name :user  
Purpose of table:This table is register the user

User\_name : NOT NULL check(‘A-Z’,’a-z’,’0-9’)

password : NOT NULL check(‘A-Z’,’a-z’,’0-9’)

name :NOT NULL check(‘A-Z’,’a-z’)

address : NOT NULL check(‘A-Z’,’a-z’,’0-9’)

panchayatname : NOT NULL check(‘A-Z’,’a-z’)

taluk : NOT NULL check(‘A-Z’,’a-z’)

mobile : NOT NULL check(’0-9’)

mail\_id : NULL check(‘A-Z’,’a-z’,’0-9’,’@’)

**QUARY**

CREATE TABLE ` user ` (

` user\_name ` varchar(10) NOT NULL check(‘A-Z’,’a-z’,’0-9’),

`password` varchar(10) NOT NULL check(‘A-Z’,’a-z’,’0-9’),

`name` varchar(20) NOT NULL check (‘A-Z’,’a-z’),

`address` varchar(30) NOT NULL check (‘A-Z’,’a-z’,’0-9’),

`panchayatname ` varchar(20) NOT NULL check (‘A-Z’,’a-z’),

` taluk ` varchar(20) NOT NULL (‘A-Z’,’a-z’),

`mobile` varchar(10) NOT NULL check(‘A-Z’,’a-z’,’0-9’),

`mailid` varchar(30) NULL check(‘A-Z’,’a-z’,’0-9’,’@’),

) ENGINE=InnoDB DEFAULT CHARSET=latin1

**Table 5.2 Registration**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Data type** | **Length** | **Description** |
| User\_name | Varchar | 20 | This field specifies the registered name of the user  eg: ananth |
| Password | Varchar | 10 | This field specifies the registered password of the user  Eg:anan123 |
| Name | Varchar | 20 | This field specifies the registered name of the user  eg: ananth |
| Address | Varchar | 50 | This field specifies the registered address of the user  Eg:3/71 SR kulam |
| Panchayatname | Varchar | 20 | This field specifies the Panchayat name  Eg:pallai |
| Taluk | Varchar | 25 | This field specifies the taluk  Eg: taluk |
| Mobile\_no | Int | 10 | This field specifies the registered mobile of the user  Eg:9524722402 |
| Mail\_id | Varchar | 20 | This field specifies the registered mail id of the user  Eg:ananth1@email.com |

Table name : officer

Purpose of table : This table is store officer details

Panchayat name : NOT NULL check(‘A-Z’,’a-z’)

Officer name : NOT NULL check(‘A-Z’,’a-z’)

District name : NOT NULL check(‘A-Z’,’a-z’)

mobile : NOT NULL check(’0-9’)

mailid : NOT NULL check(‘A-Z’,’a-z’,’0-9’,’@’)

**QUARY**

CREATE TABLE `officer` (

` panchayatname ` varchar(20) NOT NULL (‘A-Z’,’a-z’),

` officername ` varchar(20) NOT NULL (‘A-Z’,’a-z’),

` taluk ` varchar(20) NOT NULL (‘A-Z’,’a-z’),

` districtname ` varchar(30) NOT NULL (‘A-Z’,’a-z’),

`mobile` varchar(10) NOT NULL check(’0-9’),

`mailid` varchar(30) NOT NULL check(‘A-Z’,’a-z’,’0-9’,’@’),

) ENGINE=InnoDB DEFAULT CHARSET=latin1

**Table 5.3 Officer Details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Data type** | **Length** | **Description** |
| Panchayatname | Varchar | 20 | This field specifies the Panchayat name  Eg:pallai |
| officer name | Varchar | 20 | This field specifies the officer name  Eg: ananth |
| Taluk | Varchar | 25 | This field specifies the taluk  Eg: taluk |
| District name | Varchar | 20 | This field specifies district name  Eg:nallai |
| Mobile | Int | 10 | This field specifies head of dept mobile no  Eg:9524722402 |
| Mailid | Varchar | 20 | This field specifies the mail id  Eg: [ananth@email.com](mailto:ananth@email.com) |

**T**able name :applyscheme

Purpose of table :This table compliant the user

Username : NOT NULL check(‘A-Z’,’a-z’)

Complian tsubject :NOT NULL check(‘A-Z’,’a-z’)

Comliant discription : NOT NULL check(‘A-Z’,’a-z’)

Taluk : NOT NULL check(‘A-Z’,’a-z’)

District : NOT NULL check(‘A-Z’,’a-z’)

**QUARY**

CREATE TABLE `applyscheme` (

`username` varchar(20) NULL (‘A-Z’,’a-z’),

`compliantsubject` varchar(20) NULL (‘A-Z’,’a-z’),

`comliantdiscription` varchar(30) NULL (‘A-Z’,’a-z’),

`taluk` varchar(50) NULL (‘A-Z’,’a-z’),

`district ` varchar(30) NULL (‘A-Z’,’a-z’),

) ENGINE=InnoDB DEFAULT CHARSET=latin1

**Table 5.4 Compliant**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Data type** | **Length** | **Description** |
| Username | Varchar | 30 | This field specifies the user name  Eg :ananth |
| Compliant subject | Varchar | 20 | This field specifies the complaint of user  Eg: water problem |
| Compliant discription | Varchar | 200 | This field specifies the complaint of user  Eg: please solve |
| Taluk | Varchar | 20 | This field specifies the taluk name  Eg:palai |
| District | Varchar | 20 | This field specifies the districtof user  Eg: nallai |

**5.7 INPUT DESIGN**

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

**There are four modules namely:**

* Login
* Registration
* Complaint
* Officer details

**Login details**

Login page allow the user to give input values such as user name, password. (**ReferFigure9.3**)

**Registration details**

Registration page allow the user to enter the values such as name, address, mobile no, panchayat, taluk , username, password .**(ReferFigure9.5)**

**Complaint details**

Complaint page allow the user to enter the values such as username, address, mobile no, complaint subject, complaint discription and full the contents. (**ReferFigure9.6**)

**Officer details**

Officer details page allow the admin to enter the values such as username, password, panhayat name, address, mobile no, email and full the contents. (**ReferFigure9.9**)

**5.8 OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

**Login**

The login pagecheck the username nd password entered is valid. If invalid username or password is provided then an warning message will be dsplaed as “enter correct username and password”. (**ReferFigure9.3**)

**Registration details**

All the user details will be stored to the database. Details can be view by end user and managed by admin. (**ReferFigure9.5**)

**Complaint details**

The complaint details will be stored to the database. Details can be view by officer and automatically generated report . (**ReferFigure9.6**)

**View all complaint details**

All the complaint details will be stored to the database and all complaint details view the officer. Details can be view by officer. It is generated by automatically all the complaint. (**ReferFigure9.7**)

**CHAPTER 6**

**SYSTEM TESTING**

**TESTING**

Testing is a set of activities that can be planned in advance and conducted systematically.

**6.1 UNIT TETING**

In this project each service can be thought of a module. There are so many modules like Login, Registration,compliant,officer details,report generation. Giving different sets of inputs has tested each module. When developing the module as well as finishing the development so that each module works without any error. The inputs are validated when accepting from the user.For example, admin entering valid username and password to enter into the serverpage.**(referFigure9.9 )**,user entering valid username and password to enter into the service page**(refer Figure9.7 )**

Form name: User login form

Description: This form is tested by giving the username and password

Input: username: admin

Password:admin

Output: Navigate from Login Page to Admin page

**6.2 VALIDATION TESTING**

The validation testing is test for verification and validation is the process of checking that a software to username and password. The validation testing is validating the username and password. For example: Displaying an error message for the invalid username / password. **(Refer Figure 9.3)**

Form name : User login form

Description : This form tested by giving the username and password is correct or not

Input : Enter the user name and password

Output : Displaying an error message for the invalid username password

**6.3 DATA CENTRIC TESTING**

Data centric testing is to test the quality of data that has to be entered into the database is check. For example: Saving data in the database when all the fields are correctly entered. **(Refer Figure 9.9)**

Form name : officer form

Description : This form tested by giving the officer details

Input : Enter the officer name,username, password, address, mobile no, mail id

Output : Displaying officer form

**6.4 INTEGRATION TESTING**

This integration testing is the integrated to one form to another one form. The integration is the phase in software testing in which individual software modules are combined and tested as a group. For example registration forms to login form integration. **(Refer figure 9.11)**

Form name : Registration form

Description : this form tested by navigation from one page to another

Input : clicking one login to complaint

Output : navigation login page to complaint page

**6.7 TEST CASES**

**Test case Table 6.1 Login**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test condition id** | **T Test id** | **Test condition** | **Test Description** | **Test data** | **Actual Result** | **Expected Result** | **Final result** |
| TC\_LG\_01 | TC\_LG\_01 | Check whether the username textbox is value  (**Refer figure 9.2)** | Costumer enter username with combination of alphabet and number | User123 | system accept.(**Refer figure 9.2**) | system accept the data | pass |
| TC\_LG01 | TC\_LG\_02 | Check whether the password textbox is value  (**Refer figure 9.2)** | Costumer enter password with combination of alphabet and number | 123456 | system accept.  (**Refer figure 9.2**) | system accept the data | pass |

**Test case table 6.2 Registration**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test condition id** | **T Test id** | **Test condition** | **Test Description** | **Test data** | **Actual Result** | **Expected Result** | **Final result** |
| TC\_RG02 | TC\_RG\_01 | Check whether the user name textbox is value | Costumer enter user name with combination of alphabet | **Ananth123** | system accept.  (**Refer figure 9.8**) | system accept the data | **pass** |
| TC\_RG02 | TC\_RG\_02 | Check whether the password textbox is value | Costumer enter password with combination of alphabet | **12344** | system accept.  (**Refer figure 9.8**) | system accept the data | **pass** |
| TC\_RG02 | TC\_RG\_03 | Check whether the name textbox is value | Costumer enter name with combination of alphabet | **Ananth** | system accept.  (**Refer figure 9.8**) | system accept the data | **Pass** |
| TC\_RG02 | TC\_RG\_04 | Check whether the address textbox is value | Costumer enter address with combination of alphabet | **S r puram** | system accept.  (**Refer figure 9.8**) | system accept the data | **Pass** |
| TC\_RG02 | TC\_RG\_05 | Check whether the taluk textbox is value | Costumer enter taluk with combination of alphabet | **Palayam** | system accept.  (**Refer figure 9.8**) | system accept the data | **Pass** |
| TC\_RG02 | TC\_RG\_06 | Check whether the mobile no textbox is value | Costumer enter mobile no with combination of alphabet | **9812344456** | system accept.  (**Refer figure 9.8**) | system accept the data | **Pass** |
| TC\_RG02 | TC\_RG\_08 | Check whether the email textbox is value | Costumer enter email with combination of alphabet | [**anan@gmail.com**](mailto:anan@gmail.com) | system accept.  (**Refer figure 9.8**) | system accept the data | **Pass** |

**Test case table 6.3 complaint**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test condition id** | **T Test id** | **Test condition** | **Test Description** | **Test data** | **Actual Result** | **Expected Result** | **Final result** |
| TC\_CM\_01 | TC\_CM\_00 | Check whether the user name textbox is value | Costumer enter user name with combination of alphabet | **Ananth** | system accept.  (**Refer figure 9.9**) | system accept the data | **Pass** |
| TC\_CM02 | TC\_CM\_01 | Check whether the complaint textbox is value | Costumer enter complaint with combination of alphabet | **Water problem** | system accept.  (**Refer figure 9.8**) | system accept the data | **Pass** |
| TC\_CM\_03 | TC\_CM\_02 | Check whether the taluk textbox is value | Costumer enter taluk with combination of alphabet | **palayam** | system accept.  (**Refer figure 9.8**) | system accept the data | **Pass** |
| TC\_CM\_04 | TC\_CM\_03 | Check whether the district textbox is value | Costumer enter district with combination of alphabet | **Nallai** | system accept.  (**Refer figure 9.8**) | system accept the data | **Pass** |
| TC\_CM\_05 | TC\_CM\_04 | user should upload photo | customer enter valid photo | Ren.jpeg | system accept data value **(Refer figure9.8)** | system accept the data | pass |

**Test case table 6.4 officer**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test condition id** | **T Test id** | **Test condition** | **Test Description** | **Test data** | **Actual Result** | **Expected Result** | **Final result** |
| TC\_CO\_04 | TC\_CM\_01 | Check whether the user name textbox is value | Costumer enter user name with combination of alphabet | **Ananth** | system accept.  (**Refer figure 9.9**) | system accept the data | **Pass** |
| TC\_C0\_04 | TC\_CM\_02 | Check whether the password textbox is value | Costumer enter password with combination of alphabet | **123** | system accept.  (**Refer figure 9.9**) | system accept the data | **Pass** |
| TC\_CO\_04 | TC\_CM\_03 | Check whether the taluk textbox is value | Costumer enter taluk with combination of alphabet | **palayam** | system accept.  (**Refer figure 9.9**) | system accept the data | **Pass** |
| TC\_CO\_04 | TC\_CM\_04 | Check whether the email textbox is value | Costumer enter email with combination of alphabet | **anan@gmail.com** | system accept.  (**Refer figure 9.9**) | system accept the data | **Pass** |

**CHAPTER 7**

**SYSTEM IMPLEMENTATION**

**7.1 IMPLEMENTATION**

## This project used the NetBeans Platform 8.0 and NetBeans IDE 8.0.This document is an implementation of [Windowing System UI Spec](https://ui.netbeans.org/docs/ui/ws/ws_spec.html) into NetBeans IDE. The functionality of a NetBeansis provided by its Java classes:

* AllPropsNode.java
* PropertiesNotifier.java
* AllPropsChildFactory.java
* OnePropNode.java
* RefreshPropsAction.java

**Java SE**. Supports all standard Java SE development features as well as support for NetBeans Platform development and JavaFX 2.2 SDK or JavaFX 8 SDK.

**Java EE**. Provides tools for developing Java SE and Java EE applications as well as support for NetBeans Platform development and JavaFX 2.2 SDK or JavaFX 8 SDK. This download option also includes GlassFish Server Open Source Edition 4.1.1, and Apache Tomcat 8.0.27 software.

**HTML5/JavaScript**. Provides tools for HTML5/Javascript development. Includes Java Runtime Environment and does not require a separate Java installation.

**CHAPTER 8**

**CONCLUSION AND FUTURE ENHANCEMENT**

**8.1 CONCLUSION**

The Smart Village concept is easy to solve the social problem. It is necessary for us to have smart village for, sustainable and inclusive future of emerging India. Smart Villages are the need of the hour as development is needed technology itself will offer competent solution. For any overall development, concentration on the basic amenities such as health, drinking water, electricity, primary education, etc., is the first priority. User-friendly Decision Support Information system is developed to help the planners to do planning at village level. Social economic data of available in detail which can be updated in future.

**FUTURE ENHANCEMENT**

One such village which has provided better quality of life to its village people. This system helps to reach a more people. This project is very useful or the user.

**CHAPTER 9**

**APPENDICES**

**9.1 SOURCE CODE**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.sql.\*;

import java.sql.\*;

import java.util.\*;

public class ComplaintBox extends HttpServlet{

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException,IOException{

response.setContentType("text/html");

PrintWriter out = response.getWriter();

Connection conn = null;

String url = "jdbc:mysql://localhost:3306/";

String dbName = "gov";

String driver = "com.mysql.jdbc.Driver";

String userName = "root";

String password = "password";

String solved="solved";

Statement st;

try {

Class.forName(driver).newInstance();

conn = DriverManager.getConnection(url+dbName,userName,password);

System.out.println("Connected to the database");

ArrayList al=null;

ArrayList userList =new ArrayList();

String query = "select \* from applyscheme";

System.out.println("query " + query);

st = conn.createStatement();

ResultSet rs = st.executeQuery(query);

while(rs.next())

{

al = new ArrayList();

al.add(rs.getString(1));

al.add(rs.getString(2));

al.add(rs.getString(3));

al.add(rs.getString(4));

al.add(rs.getString(5));

al.add(rs.getString(6));

al.add(rs.getString(7));

al.add(rs.getString(8));

System.out.println("al :: "+al);

userList.add(al);

}

request.setAttribute("userList",userList);

RequestDispatcher rd=request.getRequestDispatcher("/viewallstatus1.jsp");

rd.forward(request,response);

conn.close();

System.out.println("Disconnected from database");

} catch (Exception e) {

e.printStackTrace();

}

}

import java.io.\*;

import java.sql.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.util.\*;

public class addpanchayat extends HttpServlet {

Connection con=null;

Statement st=null;

ResultSet rs=null;

RequestDispatcher rd=null;

HttpSession sn=null;

PrintWriter out=null;

//response.setContentType("text/html");

// PrintWriter pw = response.getWriter();

public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException,ServletException {

res.setContentType("text/html");

out = res.getWriter();

HttpSession sn = req.getSession(true);

//String level= req.getParameter("level");

String panchayat= req.getParameter("panchayat");

String officer= req.getParameter("officer");

String taluk= req.getParameter("taluk");

String district= req.getParameter("district");

String population= req.getParameter("population");

String illiterates= req.getParameter("illiterates");

String literates= req.getParameter("literates");

RequestDispatcher rd;

try {

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/gov","root","password");

st = con.createStatement();

Random generator = new Random();

int r = generator.nextInt(9999);

String ran="DM"+r;

sn.setAttribute("ran",ran);

int add=st.executeUpdate("insert into panchayat values('"+panchayat+"','"+officer+"','"+taluk+"','"+district+"','"+population+"','"+illiterates+"','"+literates+"')");

rd=req.getRequestDispatcher("success.jsp");

rd.forward(req,res);

} catch(Exception e2)

{

rd=req.getRequestDispatcher("failure.jsp");

}

}

import java.io.\*;

import java.sql.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class adminlogin extends HttpServlet {

String username="";

String password="";

Connection con=null;

Statement st=null;

ResultSet rs=null;

RequestDispatcher rd=null;

HttpSession sn=null;

PrintWriter out=null;

public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException,ServletException {

username = req.getParameter("username");

password = req.getParameter("password");

res.setContentType("text/html");

out = res.getWriter();

HttpSession sn = req.getSession(true);

sn.setAttribute("username",username);

sn.setAttribute("password",password);

RequestDispatcher rd;

try {

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/gov","root","password");

st = con.createStatement();

rs = st.executeQuery("select \* from adminlogin where username='"+username+"' && password='"+password+"'");

System.out.println(username+password);

if(rs.next())

{

rd=req.getRequestDispatcher("/admin.jsp");

}

else {

rd=req.getRequestDispatcher("/failure.jsp");

// out.println("welcome");

}

rd.forward(req,res);

} catch(Exception e2) {

//System.out.println("Exception : "+e2.toString());

out.println(e2);

}

}

import java.io.\*;

import java.sql.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.util.\*;

import javax.mail.Message;

import javax.mail.PasswordAuthentication;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

public class applyscheme extends HttpServlet {

Connection con=null;

Statement st=null;

ResultSet rs=null;

RequestDispatcher rd=null;

HttpSession sn=null;

PrintWriter out=null;

//response.setContentType("text/html");

// PrintWriter pw = response.getWriter();

public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException,ServletException {

res.setContentType("text/html");

out = res.getWriter();

HttpSession sn = req.getSession(true);

String username= req.getParameter("username");

String scheme= req.getParameter("scheme");

String level= req.getParameter("level");

String panchayat= req.getParameter("panchayat");

String taluk= req.getParameter("taluk");

String district= req.getParameter("district");

String status= "process";

String host = "smtp.gmail.com";

String from="propsessionmail@gmail.com";

String subject = "New Complaint";

RequestDispatcher rd;

try {

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/gov","root","password");

st = con.createStatement();

Random generator = new Random();

int r = generator.nextInt(9999);

String ran="DM"+r;

sn.setAttribute("ran",ran);

String user1 = "";

st = con.createStatement();

ResultSet rs1 = st.executeQuery("select \* from panchayat where taluk='"+taluk+"'");

//System.out.println(username+password);

if(rs1.next())

{

user1 = rs1.getString(2);

}

String mailid="";

st = con.createStatement();

ResultSet rs2 = st.executeQuery("select \* from officer where username='"+user1+"'");

//System.out.println(username+password);

if(rs2.next())

{

mailid = rs2.getString(7);

}

String messageText = "New Complaints received";

boolean sessionDebug = true;

// Create some properties and get the default Session.

//System.setProperty("smtp.protocols", "SSLv3");

Properties props = System.getProperties();

props.put("mail.smtp.starttls.enable","true");

props.setProperty("mail.transport.protocol","smtp");

props.setProperty("mail.host",host);

props.put("mail.smtp.auth", "true");

props.put("mail.smtp.port", "587"); //port is 587 for TLS if you use SSL then port is 465

props.put("mail.debug", "true");

props.put("mail.smtp.socketFactory.port", "465");

props.put("mail.smtp.socketFactory.fallback", "false");

props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");

Session mailSession = Session.getInstance(props, new javax.mail.Authenticator()

{

protected PasswordAuthentication getPasswordAuthentication() {return new PasswordAuthentication("UserName", "Password");

}

});

// Set debug on the Session

// Passing false will not echo debug info, and passing True will.

mailSession.setDebug(sessionDebug);

// Instantiate a new MimeMessage and fill it with the

// required information.

Message msg = new MimeMessage(mailSession);

msg.setFrom(new InternetAddress(from));

InternetAddress[] address = {new InternetAddress(mailid)};

msg.setRecipients(Message.RecipientType.TO, address);

msg.setSubject(subject);

//msg.setSentDate(new Date());

msg.setText(messageText);

// Hand the message to the default transport service

// for delivery.

Transport transport = mailSession.getTransport("smtp");

transport.connect(host, "testmailselva", "javaselva");

transport.sendMessage(msg, msg.getAllRecipients());

int add=st.executeUpdate("insert into applyscheme(username,complaintsubject,complaintdes,panchayat,taluk,district,status1) values('"+username+"','"+scheme+"','"+level+"','"+panchayat+"','"+taluk+"','"+district+"','"+status+"')");

rd=req.getRequestDispatcher("success2.jsp");

rd.forward(req,res);

} catch(Exception e2)

{

rd=req.getRequestDispatcher("failure.jsp");

}

}

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.sql.\*;

import java.sql.\*;

import java.util.\*;

public class appstatus extends HttpServlet{

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException,IOException{

response.setContentType("text/html");

PrintWriter out = response.getWriter();

Connection conn = null;

String url = "jdbc:mysql://localhost:3306/";

String dbName = "gov";

String driver = "com.mysql.jdbc.Driver";

String userName = "root";

String password = "password";

String username2= request.getParameter("username2");

Statement st;

try {

Class.forName(driver).newInstance();

conn = DriverManager.getConnection(url+dbName,userName,password);

System.out.println("Connected to the database");

ArrayList al=null;

ArrayList userList =new ArrayList();

String query = "select \* from applyscheme where username='"+username2+"'";

System.out.println("query " + query);

st = conn.createStatement();

ResultSet rs = st.executeQuery(query);

while(rs.next())

{

al = new ArrayList();

al.add(rs.getString(1));

al.add(rs.getString(2));

al.add(rs.getString(3));

al.add(rs.getString(4));

al.add(rs.getString(5));

al.add(rs.getString(6));

al.add(rs.getString(7));

System.out.println("al :: "+al);

userList.add(al);

}

request.setAttribute("userList",userList);

RequestDispatcher rd=request.getRequestDispatcher("/viewstatus.jsp");

rd.forward(request,response);

conn.close();

System.out.println("Disconnected from database");

} catch (Exception e) {

e.printStackTrace();

}

}

import java.io.\*;

import java.sql.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class userlogin extends HttpServlet {

String username="";

String password="";

Connection con=null;

Statement st=null;

ResultSet rs=null;

RequestDispatcher rd=null;

HttpSession sn=null;

PrintWriter out=null;

public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException,ServletException {

username = req.getParameter("username");

password = req.getParameter("password");

res.setContentType("text/html");

out = res.getWriter();

HttpSession sn = req.getSession(true);

sn.setAttribute("username",username);

sn.setAttribute("password",password);

RequestDispatcher rd;

try {

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/gov","root","password");

st = con.createStatement();

rs = st.executeQuery("select \* from user where username='"+username+"' && password='"+password+"'");

System.out.println(username+password);

if(rs.next())

{

rd=req.getRequestDispatcher("/userpage.jsp");

}

else {

rd=req.getRequestDispatcher("/failure1.jsp");

// out.println("welcome");

}

rd.forward(req,res);

} catch(Exception e2) {

//System.out.println("Exception : "+e2.toString());

out.println(e2);

}

}

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.sql.\*;

import java.sql.\*;

import java.util.\*;

public class viewallstatus extends HttpServlet{

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException,IOException{

response.setContentType("text/html");

PrintWriter out = response.getWriter();

Connection conn = null;

String url = "jdbc:mysql://localhost:3306/";

String dbName = "gov";

String driver = "com.mysql.jdbc.Driver";

String userName = "root";

String password = "password";

String solved="solved";

HttpSession sn = request.getSession(true);

String user1 = sn.getAttribute("username").toString();

String area="";

Statement st;

try {

Class.forName(driver).newInstance();

conn = DriverManager.getConnection(url+dbName,userName,password);

System.out.println("Connected to the database");

st = conn.createStatement();

ResultSet rs1 = st.executeQuery("select \* from panchayat where officer='"+user1+"'");

//System.out.println(username+password);

if(rs1.next())

{

area = rs1.getString(3);

}

ArrayList al=null;

ArrayList userList =new ArrayList();

String query = "select \* from applyscheme where taluk='"+area+"' && status1!='"+solved+"'";

System.out.println("query " + query);

st = conn.createStatement();

ResultSet rs = st.executeQuery(query);

while(rs.next())

{

al = new ArrayList();

al.add(rs.getString(1));

al.add(rs.getString(2));

al.add(rs.getString(3));

al.add(rs.getString(4));

al.add(rs.getString(5));

al.add(rs.getString(6));

al.add(rs.getString(7));

al.add(rs.getString(8));

System.out.println("al :: "+al);

userList.add(al);

}

request.setAttribute("userList",userList);

RequestDispatcher rd=request.getRequestDispatcher("/viewallstatus.jsp");

rd.forward(request,response);

conn.close();

System.out.println("Disconnected from database");

} catch (Exception e) {

e.printStackTrace();

}

}

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.sql.\*;

import java.sql.\*;

import java.util.\*;

public class viewscheme extends HttpServlet{

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException,IOException{

response.setContentType("text/html");

PrintWriter out = response.getWriter();

Connection conn = null;

String url = "jdbc:mysql://localhost:3306/";

String dbName = "gov";

String driver = "com.mysql.jdbc.Driver";

String userName = "root";

String password = "password";

Statement st;

try {

Class.forName(driver).newInstance();

conn = DriverManager.getConnection(url+dbName,userName,password);

System.out.println("Connected to the database");

ArrayList al=null;

ArrayList userList =new ArrayList();

String query = "select \* from scheme";

System.out.println("query " + query);

st = conn.createStatement();

ResultSet rs = st.executeQuery(query);

while(rs.next())

{

al = new ArrayList();

al.add(rs.getString(1));

al.add(rs.getString(2));

al.add(rs.getString(3));

al.add(rs.getString(4));

System.out.println("al :: "+al);

userList.add(al);

}

request.setAttribute("userList",userList);

RequestDispatcher rd=request.getRequestDispatcher("/viewscheme.jsp");

rd.forward(request,response);

conn.close();

System.out.println("Disconnected from database");

} catch (Exception e) {

e.printStackTrace();

}

}

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.sql.\*;

import java.sql.\*;

import java.util.\*;

public class viewuser extends HttpServlet{

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException,IOException{

response.setContentType("text/html");

PrintWriter out = response.getWriter();

Connection conn = null;

String url = "jdbc:mysql://localhost:3306/";

String dbName = "gov";

String driver = "com.mysql.jdbc.Driver";

String userName = "root";

String password = "password";

String panchayat = request.getParameter("panchayat");

String taluk = request.getParameter("taluk");

String district = request.getParameter("district");

String pass = "";

Statement st;

try {

Class.forName(driver).newInstance();

conn = DriverManager.getConnection(url+dbName,userName,password);

System.out.println("Connected to the database");

ArrayList al=null;

ArrayList userList =new ArrayList();

String query = "select \* from user where panchayat='"+panchayat+"' && taluk='"+taluk+"' && district='"+district+"'";

System.out.println("query " + query);

st = conn.createStatement();

ResultSet rs = st.executeQuery(query);

while(rs.next())

{

al = new ArrayList();

al.add(rs.getString(1));

pass= rs.getString(2);

al.add(rs.getString(3));

al.add(rs.getString(4));

al.add(rs.getString(5));

al.add(rs.getString(6));

al.add(rs.getString(7));

al.add(rs.getString(8));

al.add(rs.getString(9));

al.add(rs.getString(10));

al.add(rs.getString(11));

System.out.println("al :: "+al);

userList.add(al);

}

request.setAttribute("userList",userList);

RequestDispatcher rd=request.getRequestDispatcher("/viewuser.jsp");

rd.forward(request,response);

conn.close();

System.out.println("Disconnected from database");

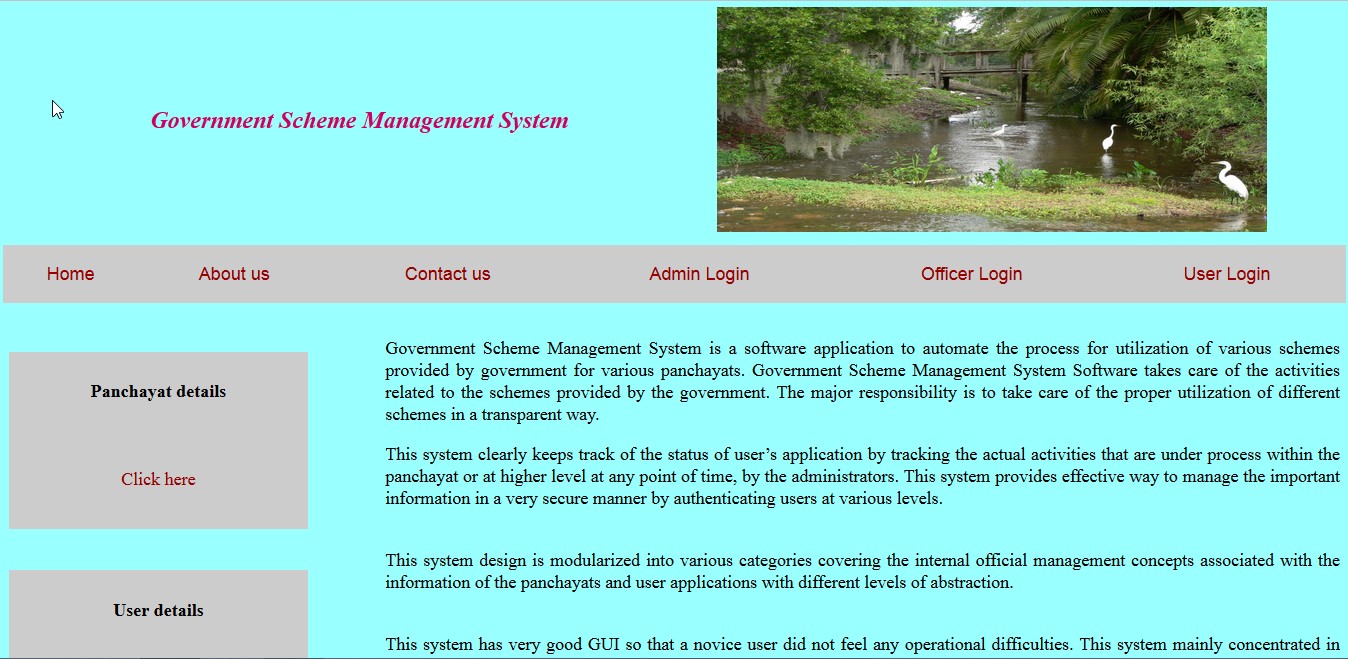
} catch (Exception e) {

e.printStackTrace();

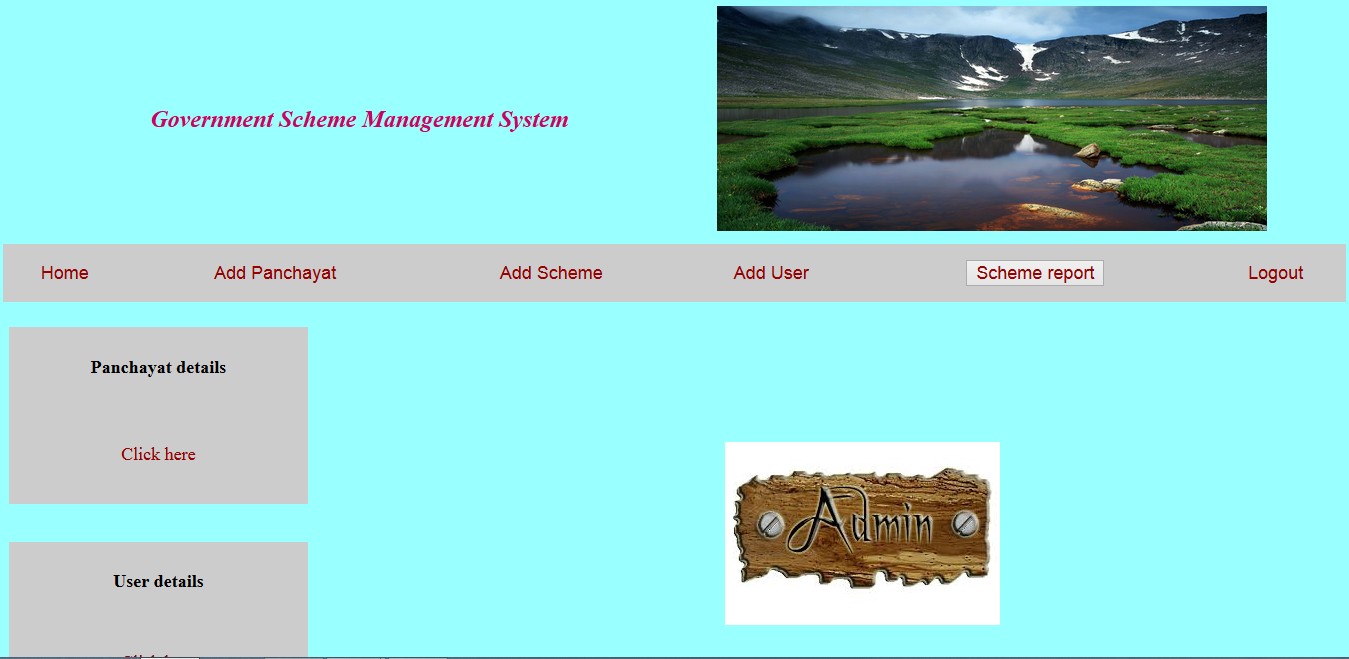
}

}

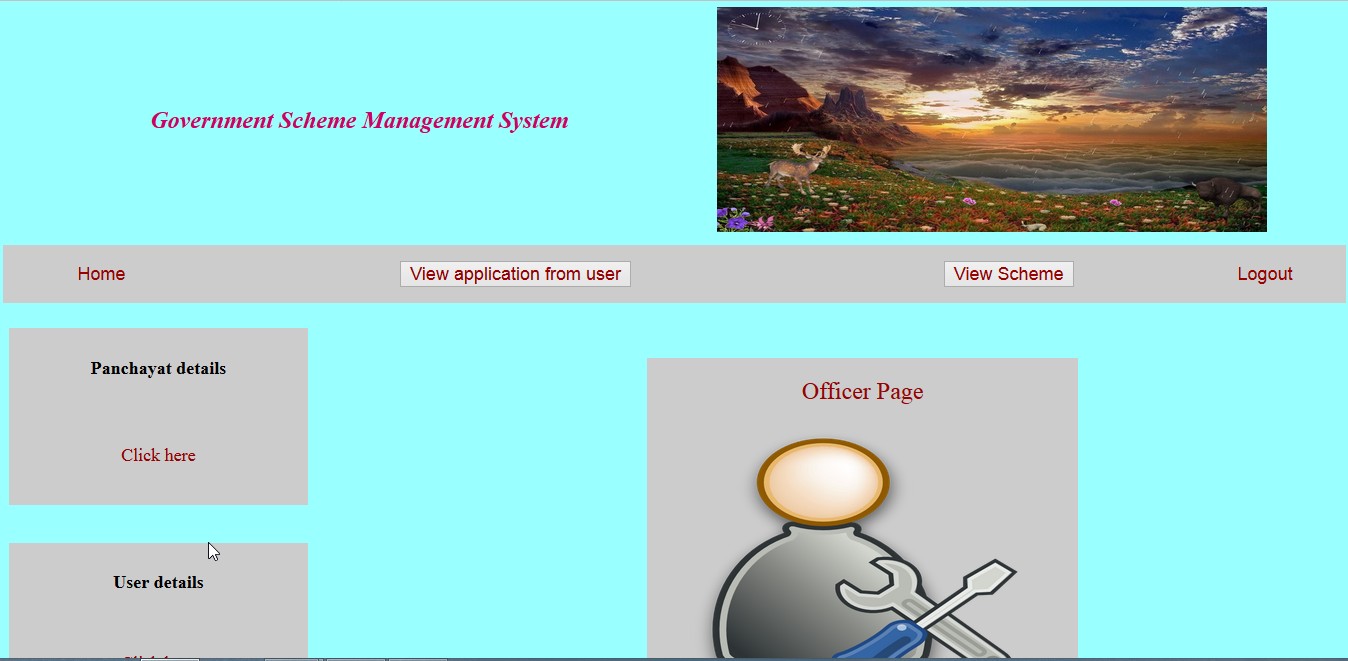
**9.2 SCREEN SHOTS**

****

**Figure 9.1 home page**

****

**Figure 9.2 Admin page**

****

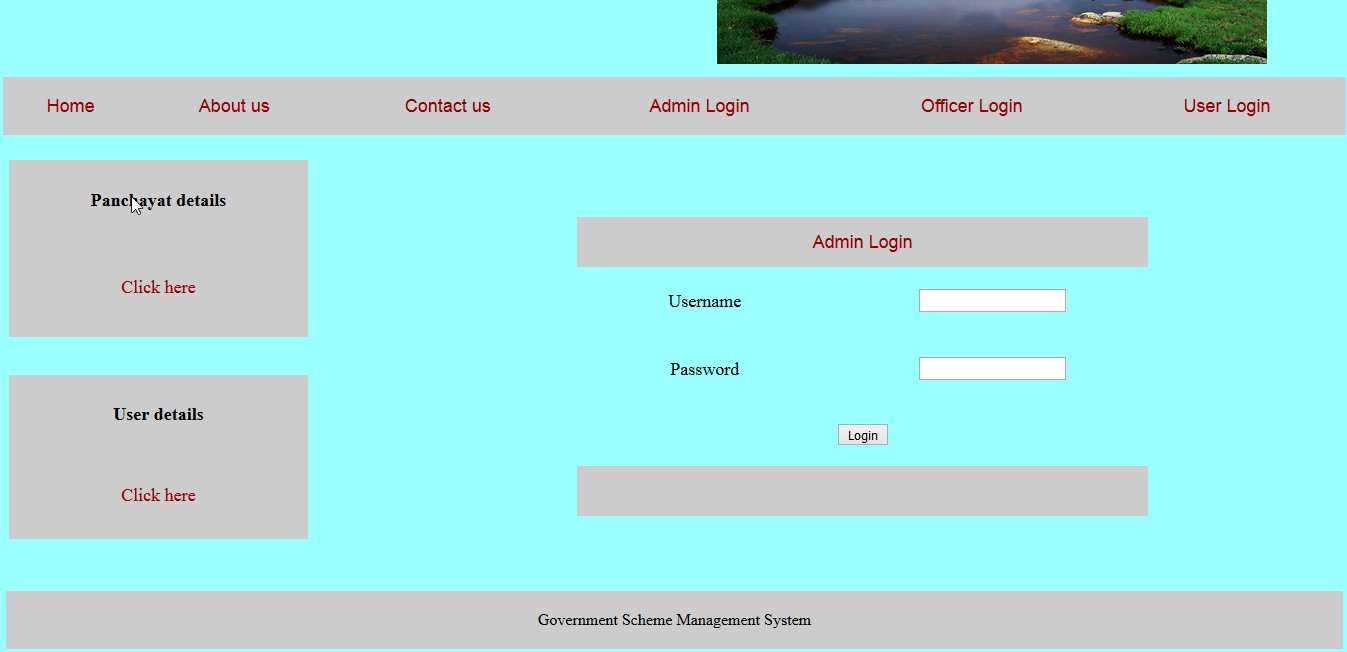
**Figure 9.3 officer page**

****

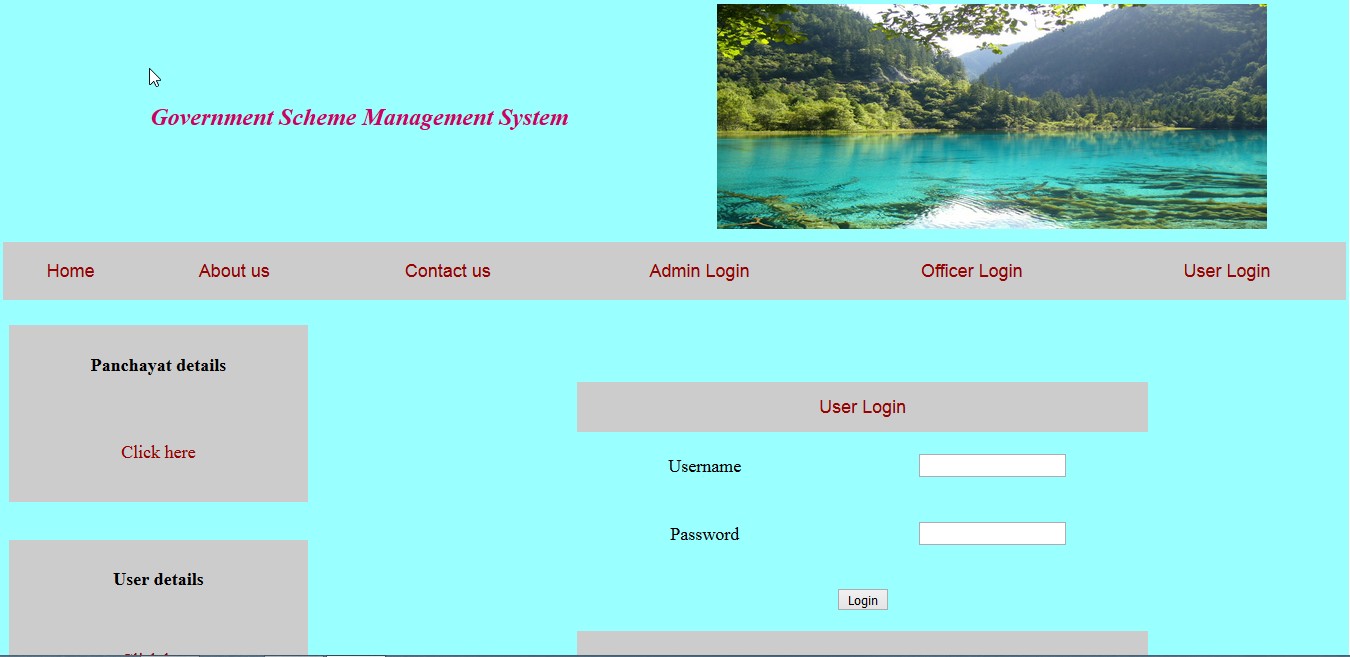
**Figure 9.4 user page**

****

**Figure 9.5 officer login**

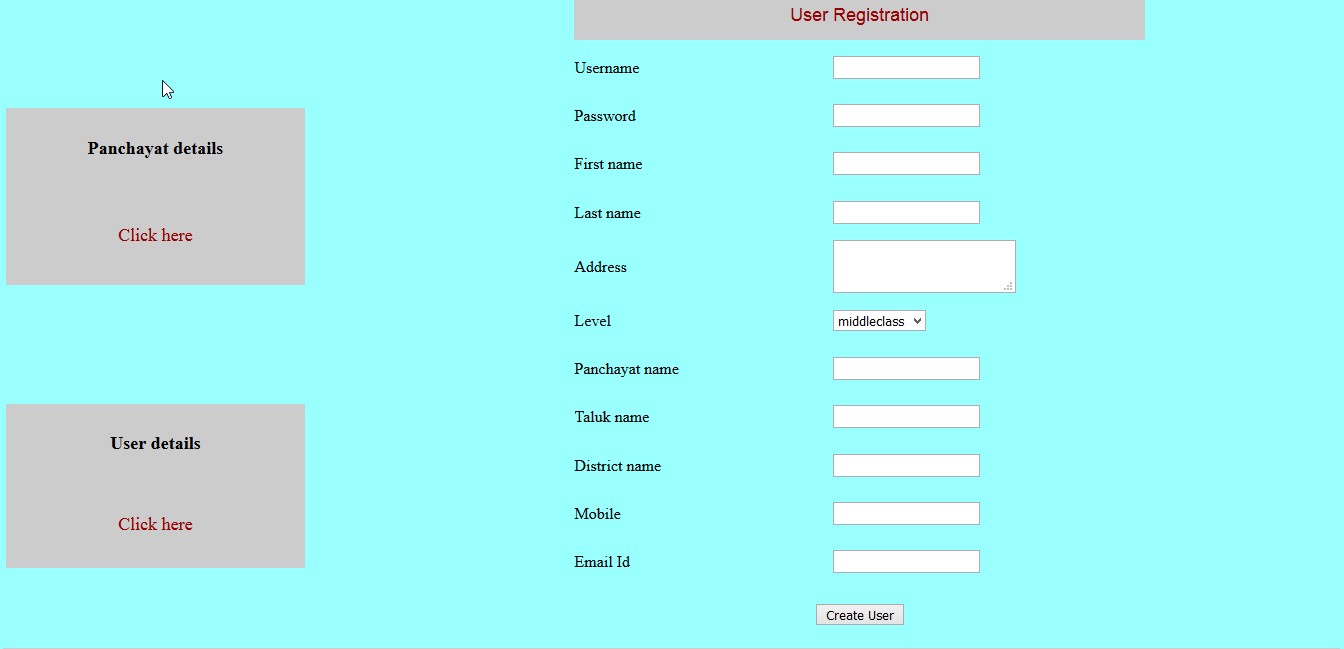
****

**Figure 9.6 admin login page**

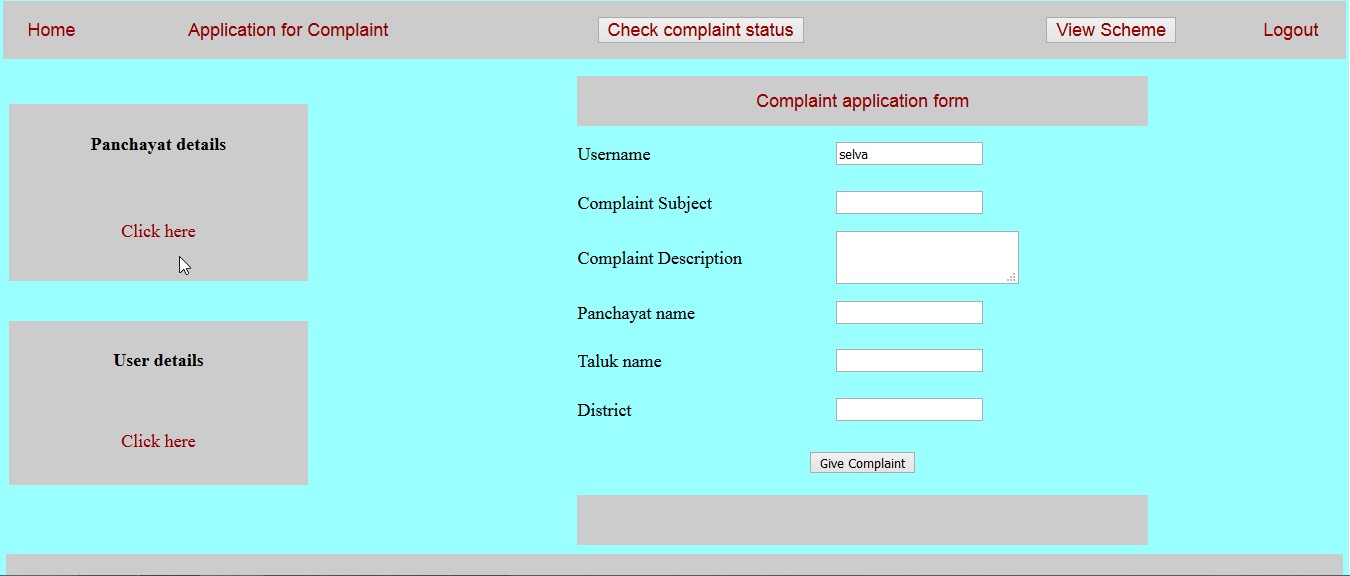
****

**Figure 9.7 user login page**

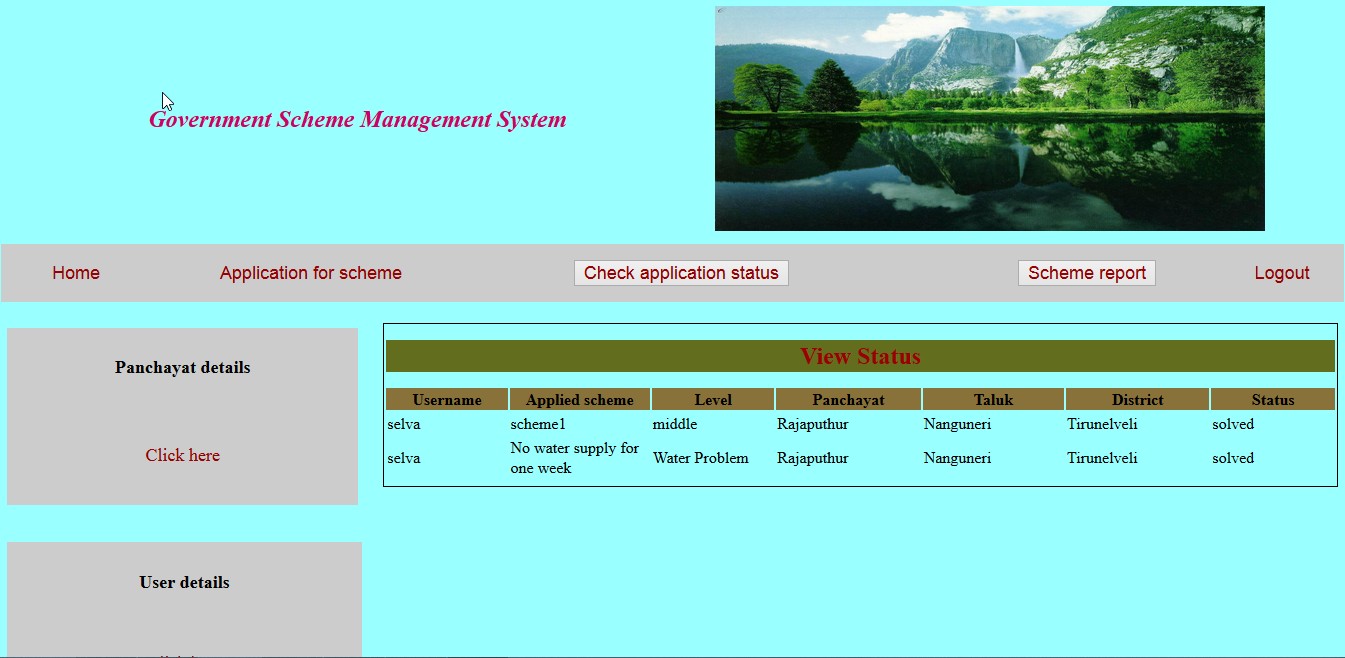
** Figure 9.8 user login invalid page**

****

**Figure 9.9 registration page**

****

**Figure 9.10 complaint page**

****

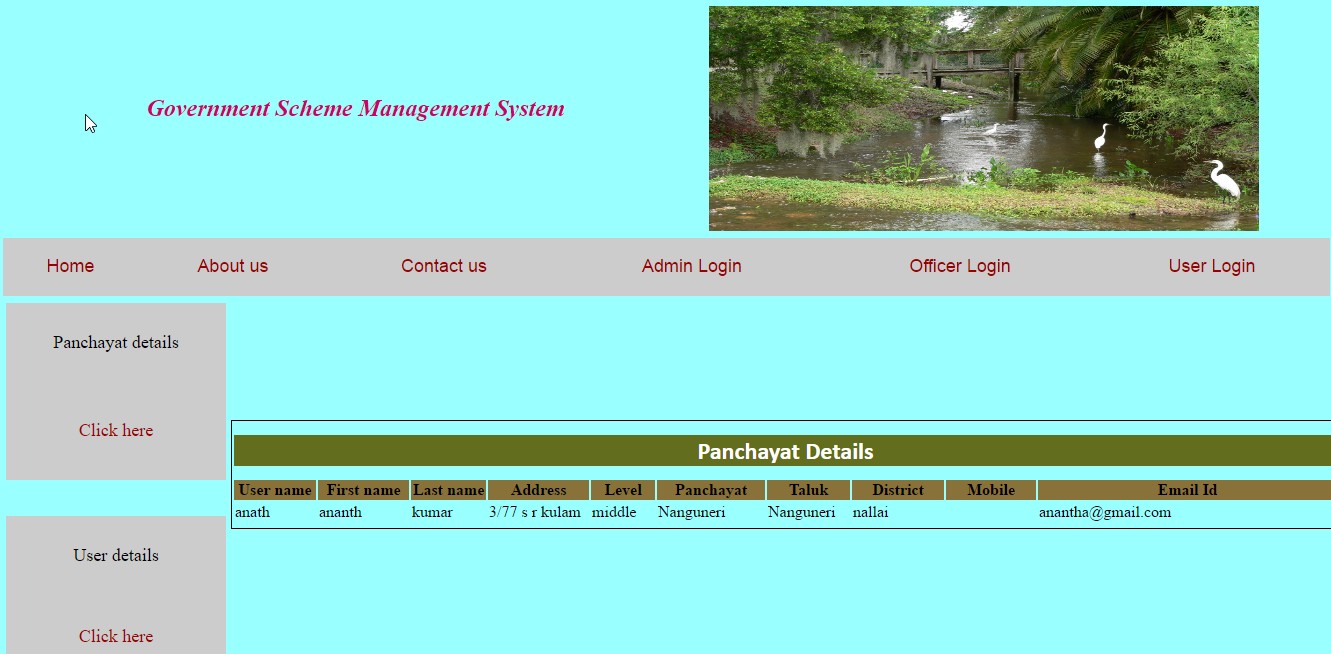
**Figure 9.11 solve complaint page**

****

**Figure 9.12 view complaint report page**



**Figure 9.13 view officer page**



**Figure 9.14 view user page**

****

**Figure 9.15 contact use page**



**Figure 9.16 About use page**

**CHAPTER 10**

**REFERENCES**

**Reference Books**

1. Herbert Schildt, “Java: The Complete Reference”, April 1, 2014.
2. Mark Matthews, “[MySQL and Java Developer’s Guide”](https://www.goodreads.com/book/show/40540.PHP_and_MySQL_Web_Development)2003.
3. George Reese, “MySQL Pocket Reference” September 24, 2016.
4. Sarang and Poornachandra, “[Java Programming (Learn Advanced Skills from a Java Expert)”](https://www.goodreads.com/book/show/4479421-php-object-oriented-solutions) January 14, 2017.

**Online References**

1. http://www.javapoint.com/sql-tutorials/
2. http://www.developer.com/java/
3. https://www.udemy.com/eclipse-java-tutorial-for-beginners/
4. http://www.w3resource.com/mysql/mysql-tutorials.com
5. https://arachnoid.com/MYSQL