Project Report on

ONLINE POTHOLE REPAIRING APPLICATION

Vickyben Patel

ABSTRACT

Online Pothole Repairing Application is a part of e-government that gives facility to citizen that citizen can register any pothole complaint online and it is being solved online. Citizen does not need to go to Pothole Repairing Office and make a physical complaint just simply register a complaint with details regarding Pothole. Citizen can track the status of complaint also by using this system. Inspector and Contractor also use this system to repair the Pothole.

Inspector adds the inspection details to the pothole complaint registered by citizen and attach the inspection image also to it after that generate the order for contractor to repair the pothole. After generating order, Inspector sends it to the Contractor.

Contractor fills up repairing details and invoice details to the Pothole Repair order given by Inspector. After filled up all details, Contractor sends those details back to Inspector and Inspector verify and approve those details.

Citizen can check the status of complaint registered by him/her. Online Pothole Repairing Application helps government to provide a good online service to its Citizen and helps Citizen to solve the pothole problem very easily and effectively.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

1.1 ABOUT THE SYSTEM

This project is a web-based application and it is a live project. 'Online Pothole Repairing Application' gives service to citizen that citizen can directly register a Pothole complaint online. Citizens do not need to go to Pothole Repairing Office and make a physical complaint. Citizen can track the status of complaint also. Inspector and contractor is also uses this system to repair the Pothole.

Potholes are repaired to ensure safe access for all highway users, and to maintain the structure of the road or footway. This system gives benefit to citizen that he/she can register a complaint of pothole online and using this system inspector and contractor will repair the system and acknowledge the citizen about his/her complaint. It's a kind of service that any government should provide to its citizen.

Citizen registers a complaint of pothole and then it will go to Inspector and Inspector inspects it and sends it to the contractor for repairing. Contractor will repair the pothole and then send repair details to inspector and inspector approves it and pothole complaint is solved and citizen is acknowledged.

1.2 PROJECT PROFILE

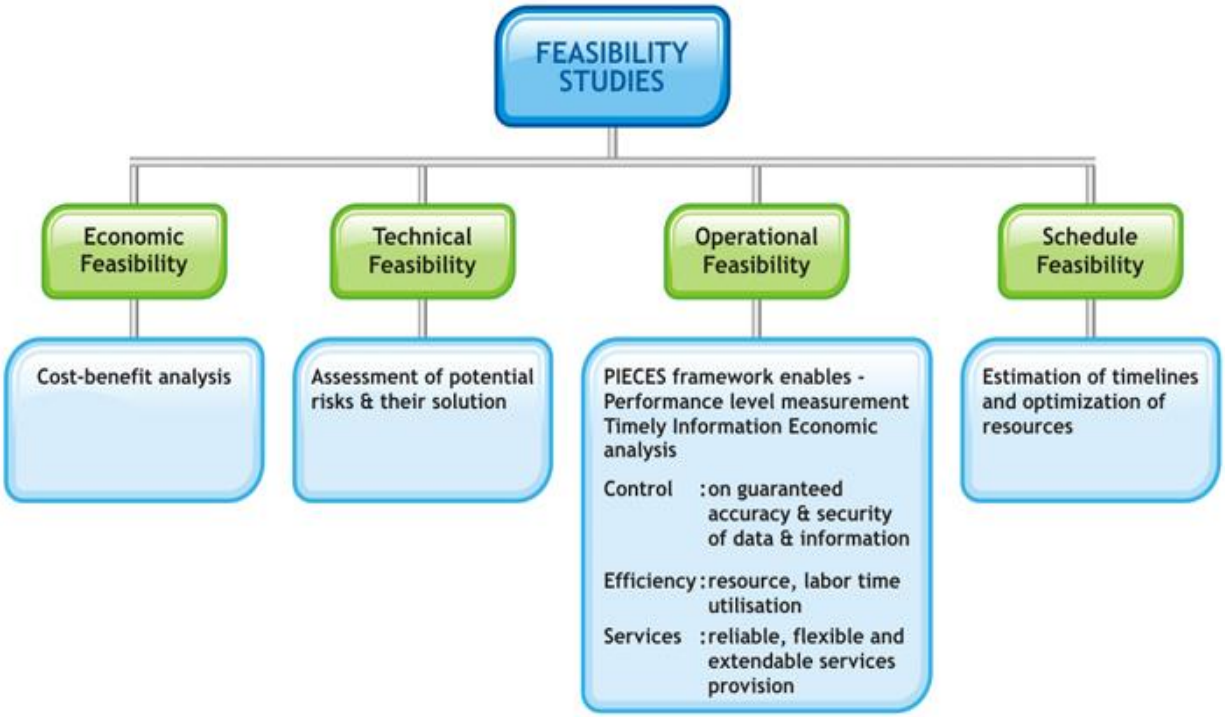
Project title	Online Pothole Repairing Application
Objective	The main objective of this system is to enhance and upgrade the existing documentary system by increasing its efficiency and effectiveness.
Developed for	Tata Consultancy Services Limited, Gandhinagar, Gujarat.
Front-end	Portal Framework, Hibernate, JSP
Back-end	DigiGOV framework, JDBC
Tools/Server	Eclipse Helios, Apache Tomcat, JBoss
Documentation tool	Ms Office 2010, StarUML, EDraw 3

Developed by	Ms. Arpita Soni(10012011117), Ms. Vicky Patel(10012011081)
Internal guide	Prof. Ketan Sarvakar
External guide	Mr. Amit Lakhtaria Mr. Ravish Tank (Project Leader)
Submitted to	U. V. Patel College of Engineering ,CE Department
Team size	Two person
Time duration	8th sem (full time)[31st December 2013 - 16st April 2014]

CHAPTER 2

FEASIBILITY ANALYSIS

The objective of a feasibility study is to find out if an information system project can be done and to suggest possible alternative solutions.



2.1 TECHNICAL FEASIBILITY

The following technical feasibility areas were probed during the technical feasibility study phase:

- The necessary development needs database technology like MySQL and various supporting tools like Eclipse, design tools like J2EE, reporting tools which are already available within the organization.
- Equipment with sufficient capacity to host the proposed system and to hold the data are available.
- Due to disconnected architecture it is possible to expand more centers. System is expandable in many dimensions with respect to addition of more functionality features etc.

2.2 TIME SCHEDULE FEASIBILITY

A project will fail if it takes too long to be completed before it is useful. Typically this means estimating how long the system will take to develop, and if it can be completed in a given time period using some methods like payback period. Schedule feasibility is a measure of how reasonable the project timetable is.

Projects are initiated with specific deadline. We need to evaluate whether the deadlines are mandatory or desirable. Time is the one of the critical factor in the development of any system but this kind of feasibility is hardly perfect in any system.

We have been asked to complete the project within the working days of the organization having period of 4 months approximately. So we have managed to complete the project before given deadline. In the Project Planning section we elaborate our ideas to develop the system within the given period.

Hence, it is feasible to develop a system in predetermined time interval.

2.3 OPERATIONAL FEASIBILITY

The following areas have been probed to declare the proposed system as Operational feasible:

- There is sufficient support for the proposed system from the management and the users as well.
- The results produced by the proposed system will be more effective and efficient in terms of speed and accuracy.
- The system is easier to use and save the time of the developer in generating the common database related task.
- It enhances the efficiency of the department.

The system will work when it is developed and installed and there are no major barriers to its implementation within the whole organization. This accounts for the operational feasibility of the proposed system.

2.4 IMPLEMENTATION FEASIBILITY

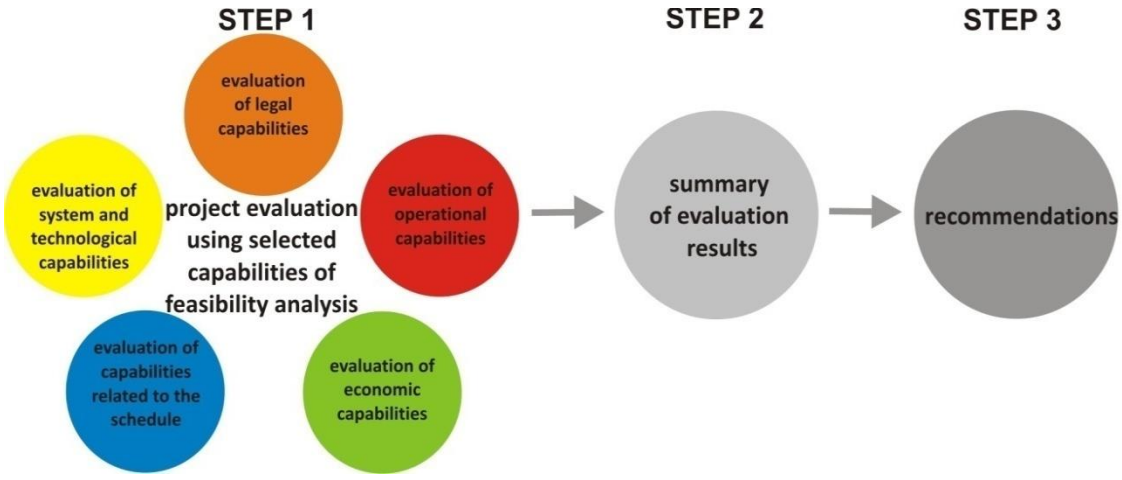
The Framework is open Source it is feasible to develop project in framework. As the TCS core team has knowledge about framework and system flow the system is the quite feasible.

2.5 ECONOMIC FEASIBILITY

Economic analysis could also be referred to as cost/benefit analysis.

- The cost of business employee time
- Estimated cost of hardware
- Estimated cost of software/software development
- The cost of doing full system study.

The economical feasibility will review the expected costs to see if they are in-line with the projected budget.
The exact costs are not required to determine economic feasibility.



CHAPTER 4

PROJECT PLAN

4.1 PROJECT SCHEDULE

Work Tasks	Planned Start Date	Planned Finish Date	Remarks
Analysis of the project.	1 -Jan – 2014	20 - Jan- 2014	On Schedule
Created Database Design	21 -Jan – 2014	1- Feb- 2014	On Schedule
Implementation of contractor module	2 - Feb- 2014	28 - Feb – 2014	On Schedule
Designing usecase, class and activity diagrams	1- Mar – 2014	4 - Mar – 2014	On Schedule
Created citizen application form, generate acknowledgement receipt and download filled form	5 - Mar – 2014	10 - Mar – 2014	On Schedule
Created BO files, DAO files, DAOImpl files and controller for citizen module	11 - Mar – 2014	20 - Mar- 2014	On Schedule
Implemented backoffice clerk module	21- Mar- 2014	29- Mar- 2014	On Schedule
Implemented Inspector module	30- Mar - 2014	14- April - 2014	On Schedule

System testing phase	15 - April - 2014	16 -April – 2014	On Schedule
Report	17-April - 2014	25-April- 2014	On Schedule

4.2 ESSENTIAL ELEMENTS FOR PROJECT PLANNING

- Management Structure
    - Share project progress and manage the work.
- Milestones
    - Break the projects into parts. We can easily know which section is completed.
- Tolerance
    - Will help manage the project without continually seeking guidance from the top executives as to whether you should carry on.
- Dependencies
    - What need to happen before something else.
- Risks
    - What are the things that had gone wrong.
- Scheduling
    - Changes are always there in schedule. We must expect the changes according to requirement.

"Plan is a living document"

CHAPTER 3

PROCESS MODEL

3.1 Prototype Model

Online Pothole Repairing Application follows "The Prototyping Model" for Development.

Often, a customer defines a set of general objectives for software but does not identify detailed input, processing, or output requirements. In other cases, the developer may be unsure of the efficiency of an algorithm, the adaptability of an operating system, or the form that human/machine interaction should take. In these, and many other situations, a *prototyping paradigm* may offer the best approach.

The prototyping paradigm begins with requirements gathering. Developer and customer meet and define the overall objectives for the software, identify whatever requirements are known, and outline areas where further definition is mandatory. A "quick design" then occurs. The quick design focuses on a representation of those aspects of the software that will be visible to the customer/user (e.g., input approaches and output formats). The quick design leads to the construction of a prototype. The prototype is evaluated by the customer/user and used to refine requirements for the software to be developed. Iteration occurs as the prototype is tuned to satisfy the needs of the customer, while at the same time enabling the developer to better understand what needs to be done.

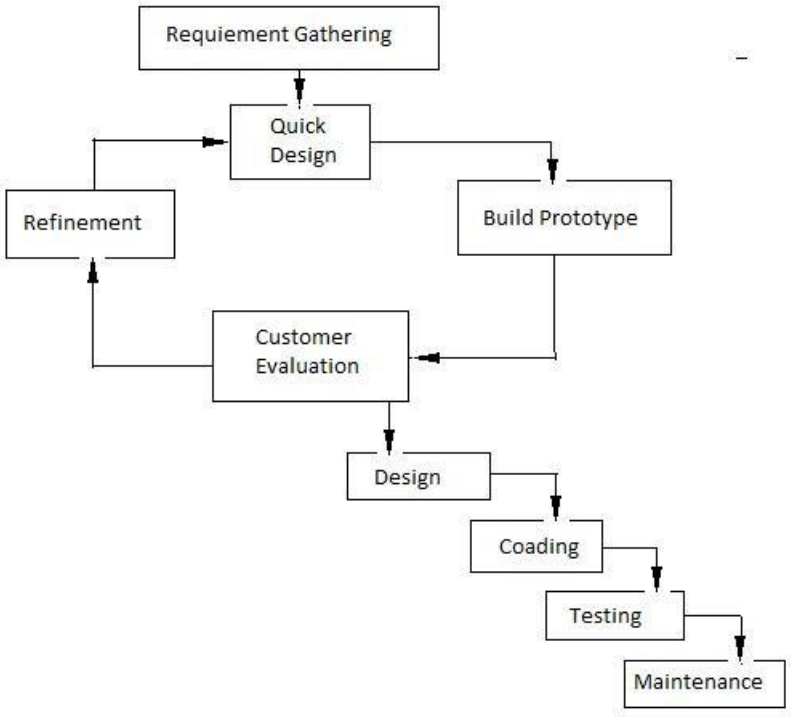


Fig 3.1 Prototype Model

Ideally, the prototype serves as a mechanism for identifying software requirements. If a working prototype is built, the developer attempts to use existing program fragments or applies tools (e.g., report generators, window managers) that enable working programs to be generated quickly.

But what do we do with the prototype when it has served the purpose just described? Answer: In most projects, the first system built is barely usable. It may be too slow, too big, and awkward in use or all three. There is no alternative but to start again, smarting but smarter, and build a redesigned version in which these problems are solved . . . When a new system concept or new technology is used, one has to build a system to throw away, for even the best planning is not so omniscient as to get it right the first time. The management question, therefore, is not whether to build a pilot system and throw it away. You will do that. The only question is whether to plan in advance to build a throwaway, or to promise to deliver the throwaway to customers

The prototype can serve as "the first system." The one that Brooks recommends we throw away. But this may be an idealized view. It is true that both customers and developers like the prototyping paradigm. Users get a feel for the actual system and developers get to build something immediately.

Reason for Using "The Prototyping Model" in Online Pothole Repairing Application

- As the parent project of Online Pothole Repairing Application IWDMS DigiGov 2.0 is live at Onsite and this application should full fill all requirements of the live project with the development phase with large requirements.
- The Functional need of this System changes according to Secretary, Chief Secretary.
- To set the behavioral functioning and attach project to the parent project.
- The accurate reason for the System to change rapidly at customer evaluation and have stability accordingly

CHAPTER 5

REQUIREMENT ANALYSIS

5.1 LIST OF INPUTS AND OUTPUTS

User: Citizen

R1.1: Register a Pothole complaint

I/p: Fill the address details and description for Pothole.

O/p: Acknowledgement Receipt is generated.

Processing: On click of submit button, details are stored in the frontend database and send to the backend side via web service and stored at backend database also.

R1.2: View Track list of Pothole complaint

I/p: Click on track list link.

O/p: Track list of Pothole complaints is displayed.

Processing: Details of complaint are fetched from database and displayed as a list on screen.

User: Clerk

R2.1: View Work list of Pothole complaints

I/p: Click on work list link.

O/p: Work list of Pothole complaints is displayed.

Processing: Details of complaint are fetched from database and displayed as a list on screen.

R2.2: Forward file to Inspector.

I/p: Click on send to within hierarchy link.

O/p: File has been sent to inspector.

Processing: Details of complaint send to inspector for further processes.

User: Inspector

R3.1: View Work list of Pothole complaints

I/p: Click on work list link.

O/p: Work list of Pothole complaints is displayed.

Processing: Details of complaint are fetched from database and displayed as a list on screen.

R3.2: Update Inspection details of pothole
I/p: Update the inspection details of pothole.

O/p: Success message and two buttons generate order, send to contractor is displayed.

Processing: Details are updated in database and page is refreshed with new values.

R3.3: Generate Pothole Repair Order
I/p: After update inspection, click on generate order button.

O/p: Order is displayed.

Processing: Details of order are fetched from the database and displayed on window.

R3.4: Send Pothole Repair Order to Contractor
I/p: Click on send to contractor button.

O/p: Success message is displayed.

Processing: Web service is called and order will go to contractor.

R3.5: Approve Pothole Invoice Details

R3.5.1: View Contractor Invoice Details
I/p: Click on view contractor invoice link.

O/p: Details sent by contractor is displayed.

Processing: Details of invoice are fetched from the database and displayed on screen.

R3.5.2: Approve Contractor Invoice Details
I/p: After checking and add approved invoice details click on approve button.

O/p: Success message is displayed.

Processing: Details of invoice are saved in the database and complaint is approved.

User: Contractor

R4.1: View Work list of Pothole complaints

I/p: Click on work list link.

O/p: Work list of Pothole complaints is displayed.

Processing: Details of complaint are fetched from database and displayed as a list on screen.

R4.2: Submit repair details for repaired Pothole

I/p: Insert the repair details and remarks for repaired pothole.

O/p: Contractor's Work list is displayed.

Processing: Repair Details are stored in frontend database and then send to inspector through web service and stored at backend also.

5.2 USECASE DIAGRAM

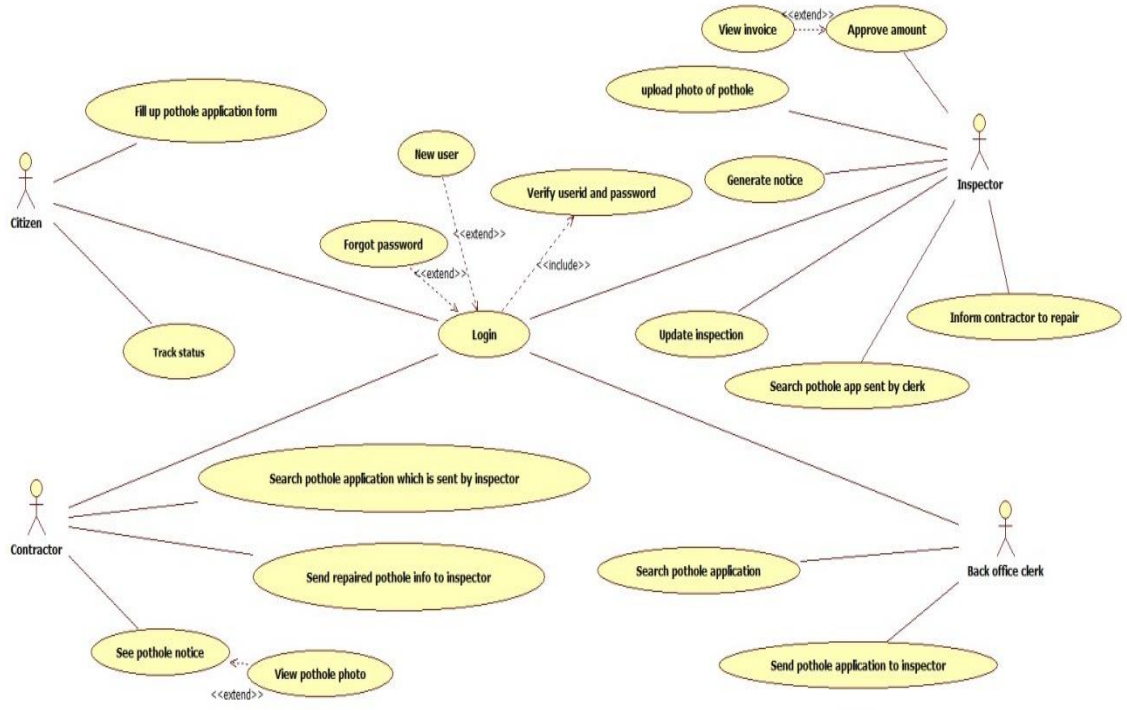


Fig 5.2 Usecase diagram

5.3 HARDWARE AND SOFTWARE REQUIREMENTS

5.3.1 Hardware Requirement

Processor : Intel Core Duo 2.0 GHz or more

RAM : 1 GB RAM (Recommended)
Hard disk : 40 GB or above Hard Disk Drive

5.3.2 Software Requirement

Platform support: All Leading OS
Application Development tool used:

Portal : Hibernate Framework, Apache Tomcat Server, Eclipse Helios.

Back Office : Eclipse Helios, JBoss Server, DigiGov framework.

Standard web Browser (Chrome or Firefox required for best result)

5.3.3 Functional Requirement

The various functional requirements of the system can be summarized as:

- Citizen, clerk, inspector, contractor login facility.
- System gives service to citizen that citizen can directly register a Pothole complaint online.
- Citizen can track the status about his/her complaint so he/she know That process.
- Citizen can download filled form.
- File can only be approved by Inspector.
- It's totally online system so it's very fast.

5.3.4 Non Functional Requirement

Availability:
The system is available at all times on condition that both side server should be running properly.

Security:
The system is password protected. User with correct user-Id and password can login to the system no one else.

Maintainability:
The system designs in a maintainable manner. It is easy to incorporate new requirements in the individual modules.

Reliability:
The system is highly reliable.

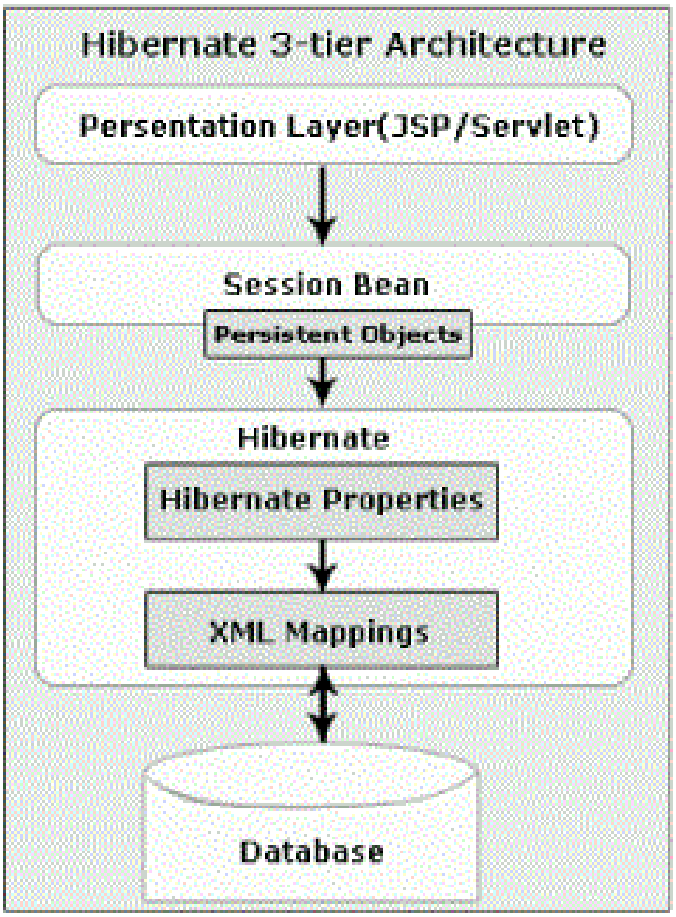
CHAPTER 6

SYSTEM DESIGN

6.1 DESIGN STRATEGY

- **3-Tier Architecture**

  Where the client request is sent to the server and the server in turn sends the request to the database. The database sends back the information/data required to the server which in turn sends it to the client.



6.2 DFD (DATA FLOW DIAGRAM)

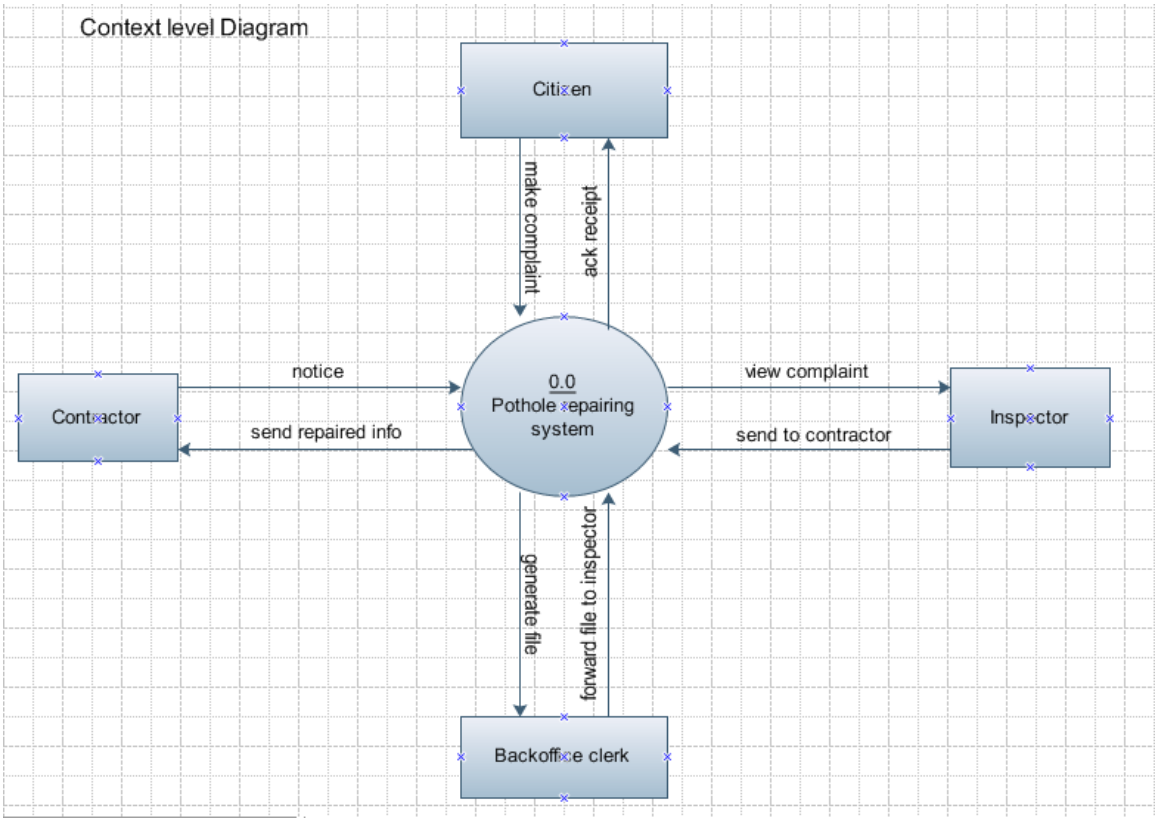


Fig 6.2.1 Context level diagram

Level 1

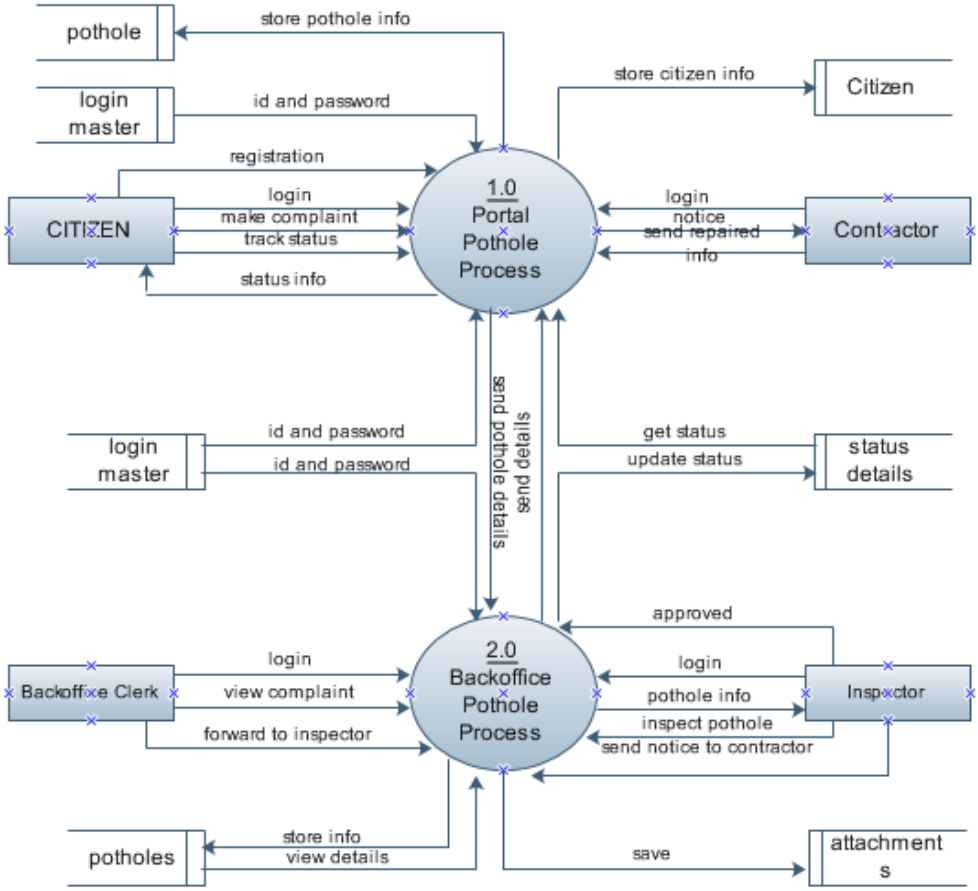


Fig 6.2.2 level-1

Level 2

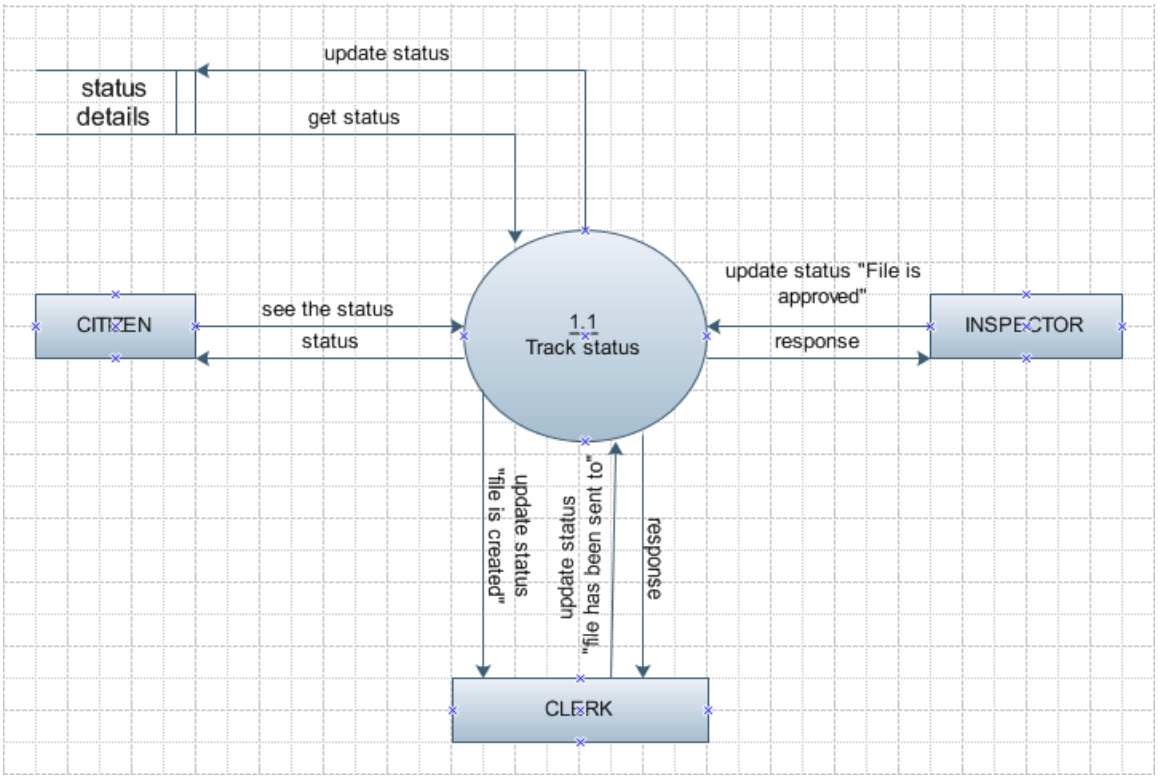


Fig 6.2.3 level-2

6.2 CLASS DIAGRAM

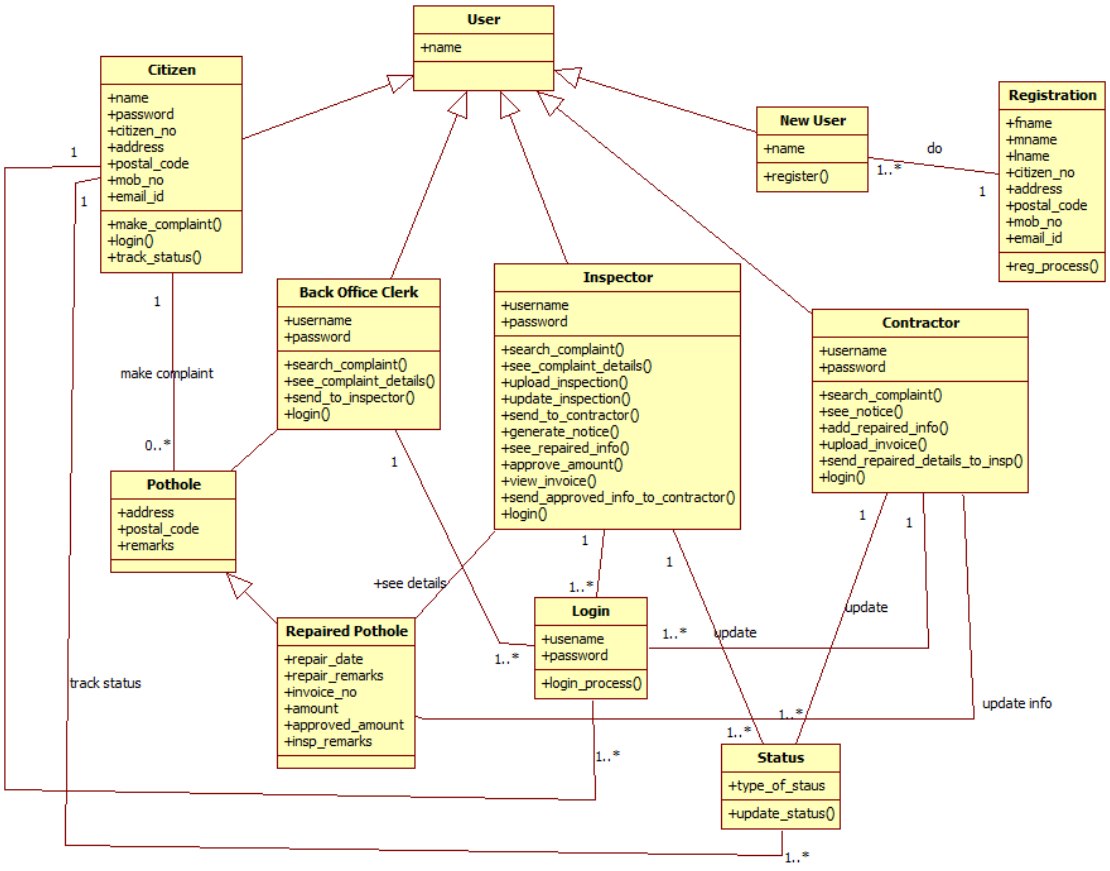


Fig 6.3 Class diagram

6.4 SEQUENCE DIAGRAM

6.4.1 Sequence Diagram for Register Pothole

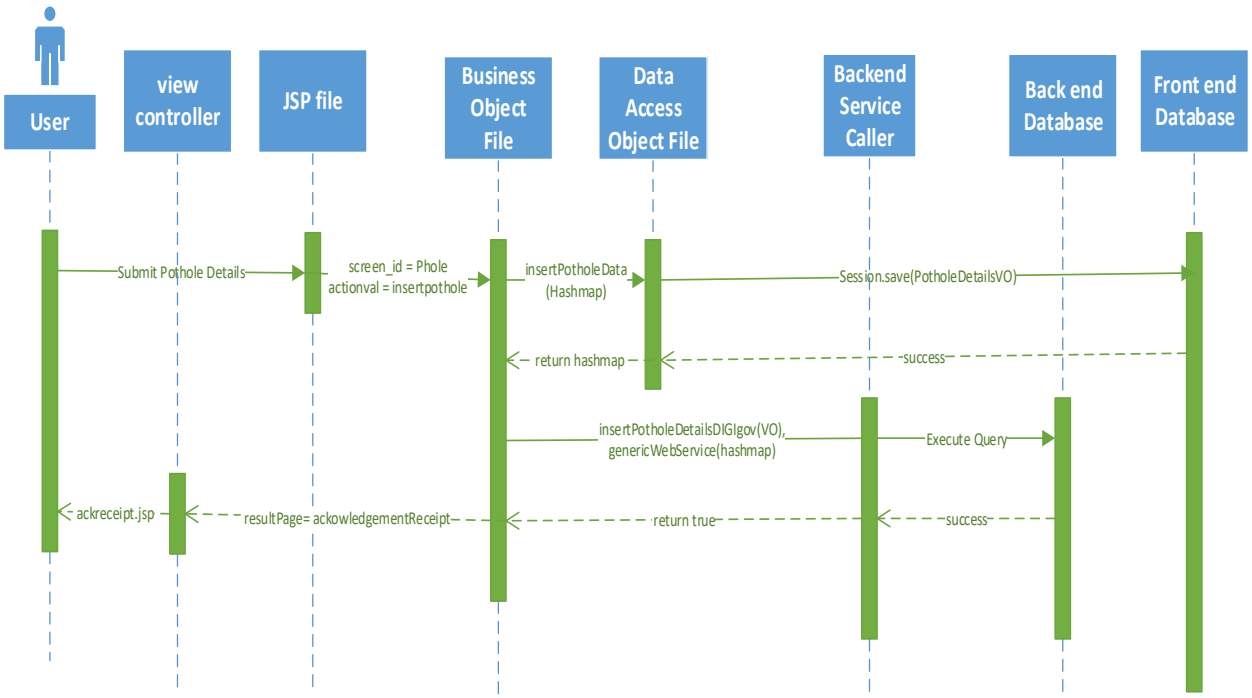


Fig 6.4.1 Sequence Diagram for Register Pothole

6.4.2 Sequence Diagram for viewing Work list

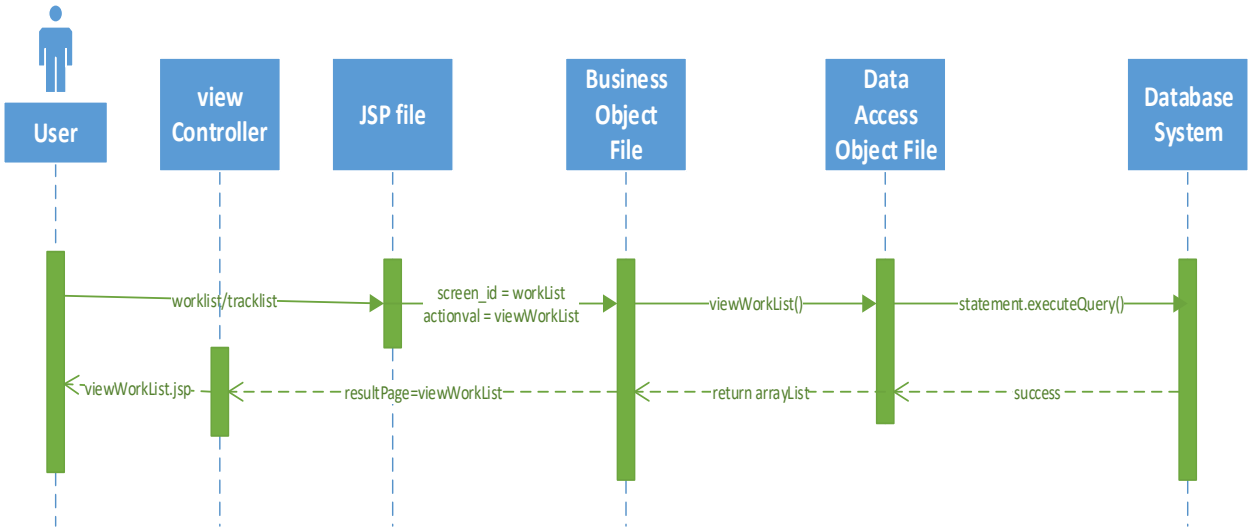


Fig 6.4.2 Sequence Diagram for viewing Work list

6.4.3 Sequence Diagram for Update Inspection

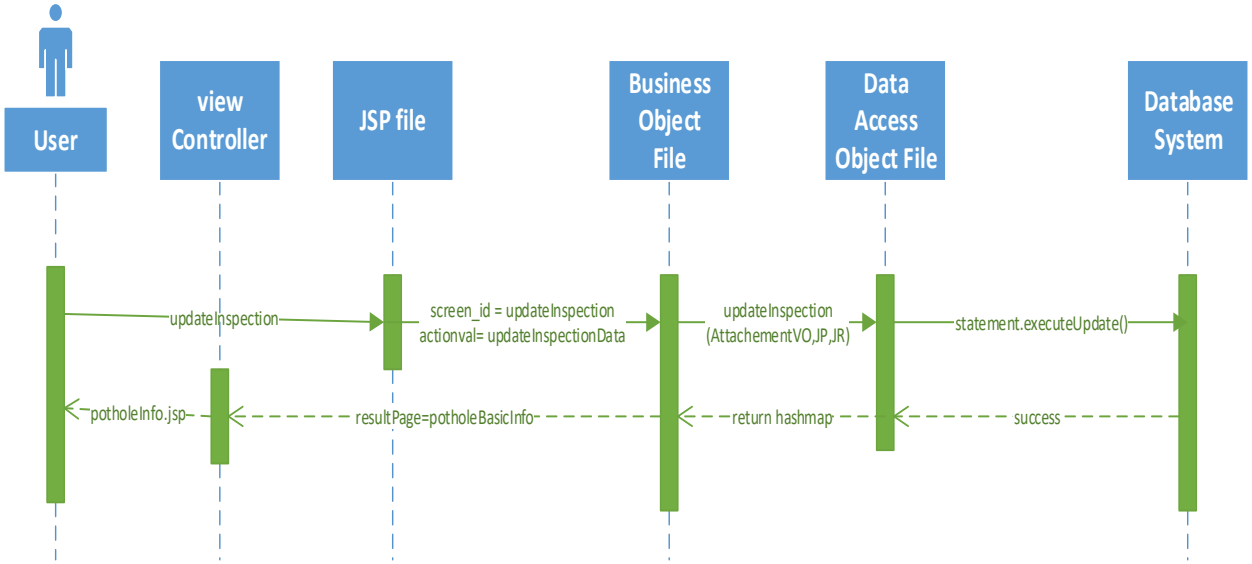


Fig 6.4.3 Sequence Diagram for Update Inspection

6.4.4 Sequence Diagram for Generating Order

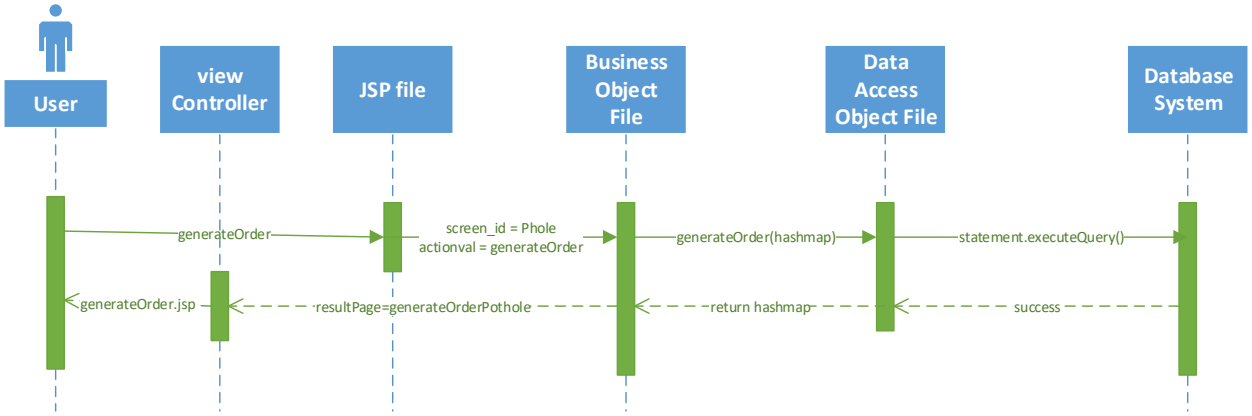


Fig 6.4.4 Sequence Diagram for Generating Order

6.4.5 Sequence Diagram for Sending Pothole complaint to Contractor

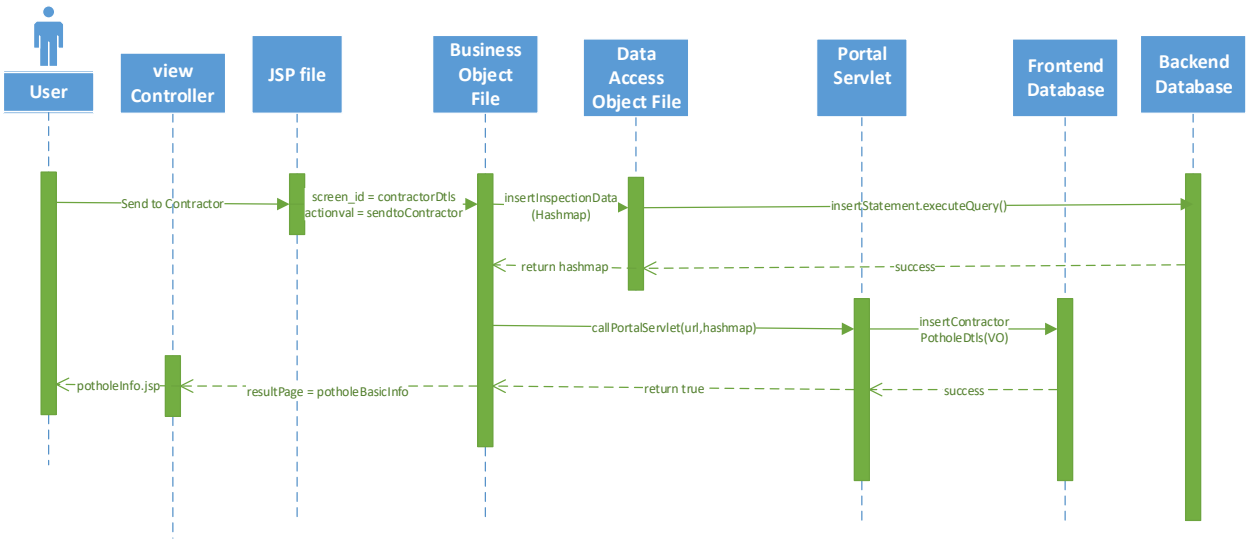


Fig 6.4.5 Sequence Diagram for Sending Pothole complaint to Contractor

6.4.6 Sequence Diagram for Sending Invoice Details to Inspector

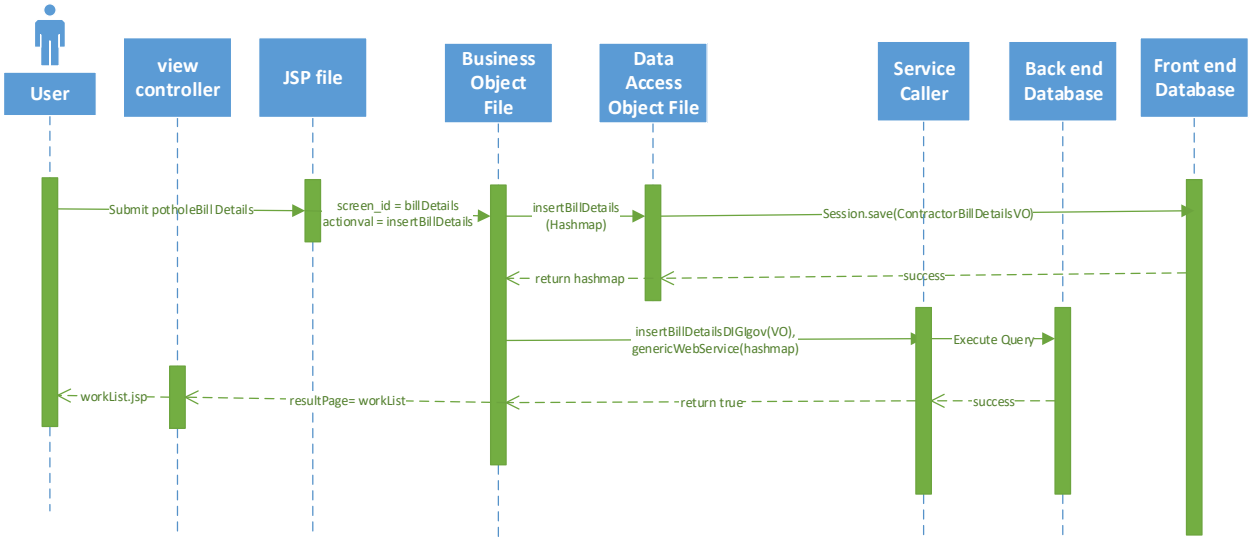


Fig 6.4.6 Sequence Diagram for Sending Invoice Details to Inspector

6.5 ACTIVITY DIAGRAM

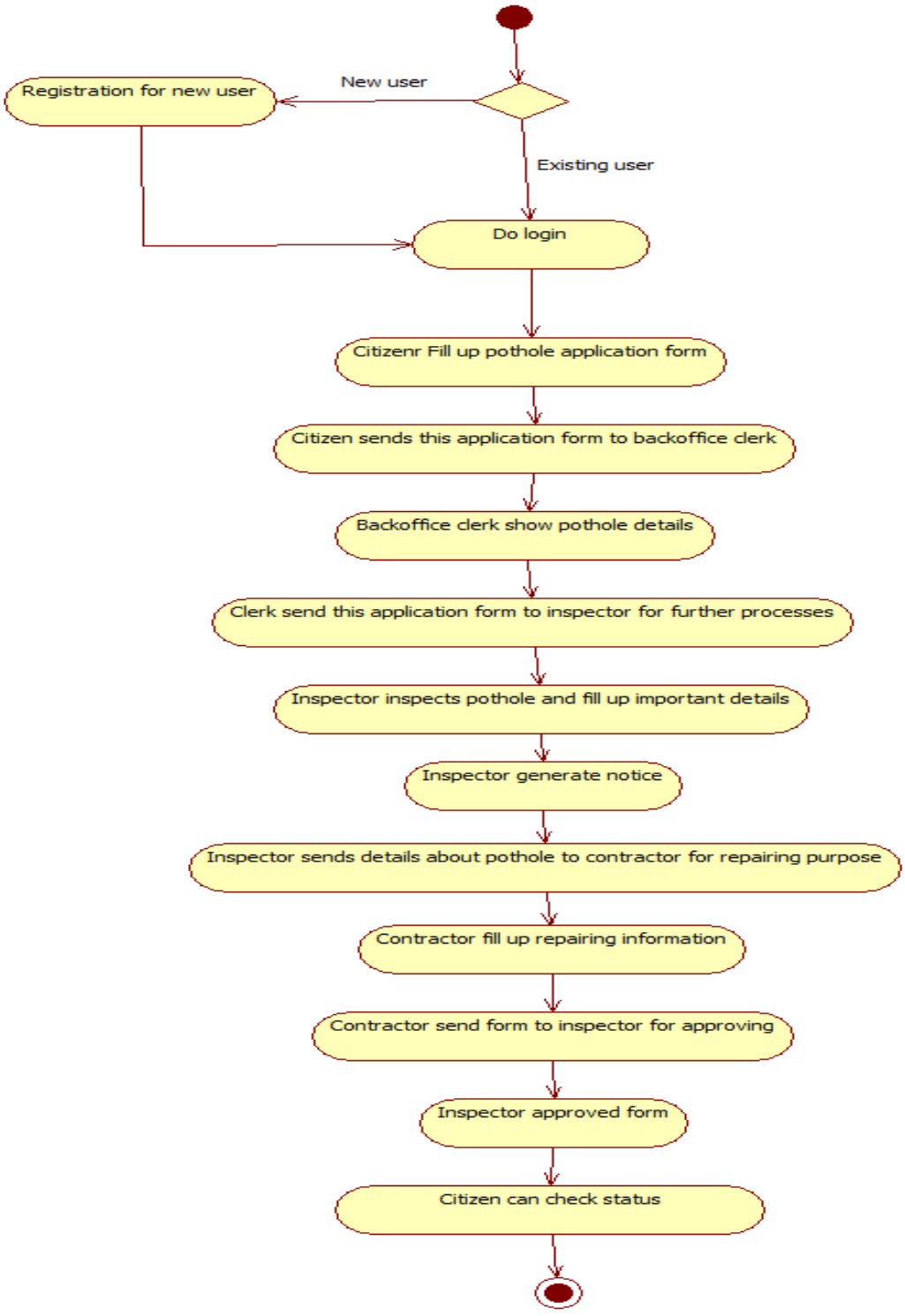


Fig 6.5 Activity Diagram

6.6 DATA DICTIONARY

The data in the system has to be stored and retrieved from database. Designing the database is part of system design. Data elements and data structures to be stored have been identified at analysis stage.They are structured and put together to design the data storage and retrieval system.

A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make database access easy, quick, inexpensive and flexible for the user. Relationships are established between the data items and unnecessary data items are removed.

Normalization is done to get an internal consistency of data and to have minimum redundancy and maximum stability. This ensures minimizing data storage required, minimizing chances of data inconsistencies and optimizing for updates.

Front-end (Portal) Database:-

Portal_Pothole_Dtls			
Column Name	Type	Key	Referenced Table
Sr_No	BIGINT(10)	Primary Key	
Token_ID	BIGINT(10)		
Person_ID	BIGINT(10)		
App_ID	BIGINT(10)		
Address_ID	BIGINT(10)	Foreign Key	Address_Dtls
Remarks	LONGTEXT		
Crt_Date	DATETIME		
Crt_User	BIGINT(10)		
Lst_Upd_Date	DATETIME		
Lst_Upd_User	BIGINT(10)		
Loc_ID	VARCHAR(10)		
Lang_ID	VARCHAR(10)		
Sync_Flag	VARCHAR(5)		
Sent_Flag	VARCHAR(5)		

Portal_Pothole_attachment			
Column Name	Type	Key	Referenced Table
Attach_ID	BIGINT(10)	Primary Key	
Token_ID	BIGINT(10)	Foreign Key	Portal_Contractor_Bill
Attach_Name	VARCHAR(50)		
Attach_Ext	VARCHAR(10)		
Attach_Databytes	LONGBLOB		
Crt_Date	DATE		
Crt_User	VARCHAR(20)		

Portal_Contractor_Bill			
Column Name	Type	Key	Referenced Table
Token_ID	BIGINT(10)	Primary Key	
Node_ID	BIGINT(10)		
Bill_No	BIGINT(10)		
Repair_Date	DATETIME		
Repair_Remark	VARCHAR(250)		
Amount	BIGINT(20)		

City_Dtls			
Column Name	Type	Key	Referenced Table
City_Id	BIGINT(10)	Primary Key	
City_Name	VARCHAR(30)		

Area_Dtls			
Column Name	Type	Key	Referenced Table
Area_Id	BIGINT(10)	Primary Key	
City_Id	BIGINT(10)	Foreign Key	City_Dtls
Area_Name	VARCHAR(30)		

Street_Dtls			
Column Name	Type	Key	Referenced Table
Street_Id	BIGINT(10)	Primary Key	
Area_Id	BIGINT(10)	Foreign Key	Area_Dtls

Street_Name	VARCHAR(30)		
Post _Code	VARCHAR(10)		

Address_Dtls			
Column Name	Type	Key	Referenced Table
Address_Id	BIGINT(10)	Primary Key	
Post _Code	VARCHAR(10)		
City_Id	BIGINT(10)	Foreign Key	City_Dtls
Area_Id	BIGINT(10)	Foreign Key	Area_Dtls
Street_Id	BIGINT(10)	Foreign Key	Street_Dtls

Back-end Database:-

Pothole_Dtls			
Column Name	Type	Key	Referenced Table
Sr No.	BIGINT(10)	Primary Key	
Token_Id	BIGINT(10)		
Person_Id	BIGINT(10)		
App_Id	BIGINT(10)		
Address_Id	BIGINT(10)	Foreign Key	Address_Dtls
Crt_Date	DATETIME		
Crt_User	BIGINT(10)		
Lst_Upd_Date	DATETIME		

Lst_Upd_User	BIGINT(10)		
Loc_Id	VARCHAR(10)		
Lang_Id	VARCHAR(10)		
Sent_Flag	VARCHAR(5)		
Job_Priority	VARCHAR(5)		
Inspection_Desc	VARCHAR(250)		
Job_Req	VARCHAR(5)		
File_Id	BIGINT(10)		
Repair_Flag	VARCHAR(5)		
Approve_Flag	VARCHAR(5)		

Contractor_Uploads			
Column Name	Type	Key	Referenced Table
Attach_Id	BIGINT(10)	Primary Key	
Token_Id	BIGINT(10)	Foreign Key	Contractor_Bill
Attach_Name	VARCHAR(50)		
Attach_Ext	VARCHAR(10)		
Attach_Databytes	LONGBLOB		
Crt_Date	DATE		
Crt_User	VARCHAR(20)		

Contractor_Bill			
Column Name	Type	Key	Referenced Table
Token_Id	BIGINT(10)	Primary Key	
Node_Id	BIGINT(10)		
Bill_No	BIGINT(10)		
Repair_Date	DATETIME		
Repair_Remark	VARCHAR(250)		
Contractor_Amount	BIGINT(20)		
Approved_Amount	BIGINT(20)		
Inspector_Remarks	VARCHAR(200)		

City_Dtls			
Column Name	Type	Key	Referenced Table
City_Id	BIGINT(10)	Primary Key	
City_Name	VARCHAR(30)		

Area_Dtls			
Column Name	Type	Key	Referenced Table
Area_Id	BIGINT(10)	Primary Key	
City_Id	BIGINT(10)	Foreign Key	City_Dtls

Area_Name	VARCHAR(30)		

Street_Dtls			
Column Name	Type	Key	Referenced Table
Street_Id	BIGINT(10)	Primary Key	
Area_Id	BIGINT(10)	Foreign Key	Area_Dtls
Street_Name	VARCHAR(30)		
Post _Code	VARCHAR(10)		

Address_Dtls			
Column Name	Type	Key	Referenced Table
Address_Id	BIGINT(10)	Primary Key	
Post _Code	VARCHAR(10)		
City_Id	BIGINT(10)	Foreign Key	City_Dtls
Area_Id	BIGINT(10)	Foreign Key	Area_Dtls
Street_Id	BIGINT(10)	Foreign Key	Street_Dtls

CHAPTER 7

IMPLEMENTATION DETAILS

In 'Online Pothole Repairing Application', at portal side Model, View and Controller is used and hibernate is also used. And at backend side Model, View, Controller and JDBC is used. For transferring the data between portal and backend, web service is used. One Servlet at portal side and service caller at backend side is used. Servlet and Service caller both contains URL of each other. Using the URL, the method of each other's is called and remotely executed. Like on register of pothole complaint, the entry is done at portal database and all the details are sent to backend using web service and executed one method at backend side and do entry at backend database also. Web services is used when pothole complaint is send to contractor by inspector and send invoice details to inspector by contractor.

7.1 DETAIL EXPLANATION OF STEPS CARRIED OUT TO IMPLEMENT THE PROJECT

7.1.1 Register a Pothole complaint

In this part, user enters postcode, city, street, area of POTHOLE and detailed description about the POTHOLE. You can enter address details of POTHOLE by area or by postcode. If you select by postcode and enter the postcode then city, street and area details are dynamically auto-populated with the city, area and street related with that postcode. And if you select by area details then you can select area, city, and street from the list.

All that data comes from database by applying queries and mapping of all postcode to areas are done in the database tables. AJAX is used for dynamically loading address details into combo boxes. It contains submit and reset button. After fill up the details about the pothole, you can submit it or reset all the fields. On submit entry is done in portal database and also enter the record at backend database using web service.

7.1.2 View Track list of complaints

Using this functionality, user can view status of his/her complaints. When user clicks on link of Track list then it retrieves the list of complaints for that user from database and displays on screen. User can track his/her complaint from it and can check the current status of the complaint

7.1.3 Acknowledgement Receipt

Acknowledgement Receipt is generated after register a Pothole complaint. User can download filled form using Download Filled Form button and also can print the receipt using Print button.

7.1.4 Forward File to Inspector

Citizen registers pothole complaint that details will go to back Office clerk and then back office clerk forward it to inspector.

7.1.5 Update Inspection of Pothole by Inspector

In this part, Inspector attaches the attachment related to Pothole and update job required or not and set the priority of job. Inspector can update the inspection as many times he wants to by using ‘update inspection button.

7.1.6 Generate Order for Pothole Complaint

In this part, Inspector generates order for contractor to repair the pothole. After update the inspection, Inspector can click on ‘generate order’ button to generate the order for pothole. By clicking that button it will go to JSP file to BO file and then to DAOImpl file and at last from database, fetch the values and will display on screen.

7.1.7 Send to Contractor

In this part, Inspector sends the generated order to contractor using ‘send to contractor’ button. By clicking button, one sent flag is set in backend database so that it proves that particular complaint is sent to contractor and details will send to contractor using web service. In web service, one Servlet is used at portal side and service caller at backend side. Details are sent in hash map and URL of portal is also sent with the hash map and at portal side all the details are gotten from map and one entry is done in portal database also.

7.1.8 Submit the Repair Details by Contractor

In this part, after repairing the pothole by contractor, he will fill repair details of pothole and submit it. On submit, one entry is done in portal database and repair details are sent to inspector via web service call.

7.1.9 View Work list

This is for Inspector and Contractor. Both can see their pothole complaints by clicking work list link. It will fetch details of complaints from the database and displays as a list (Inbox) of pothole complaints.

7.1.10 Prototypes of some of the important methods:

- insertPotholeAddressDtls(AddressVO)
- insertPotholeDtls(PotholeDtlsVO)
- insertPotholeAttachmentDtls(tokenId,arrayList)
- getPotholeAddressDtls(Address ID)
- getAtttachment(AttachId)

- getCountyDtls(countyId)
- getAreaDtls(areaId)
- getStreetDtls(streetId)
- insertContractorBillDtls(ContractorBillVO)
- setApproveFlag(tokenId)
- setSentFlag(tokenId)
- getContractorRepairDtls(tokenId)
- insertContractorUploadDtls(tokenId,arrayList)
- insertDIGIgovPotholeDtls(PotholeDtlsVO,lbytes,filePath)
- insertApproveDtls(tokenId,arrayList)
- insertPotholeInfo(PotholeDtlsVO

7.1.11 Implementation Methodology

7.1.11.1 *Object Oriented Methodology*

We live in a world of objects. These objects exist in nature, in man-made entities, in business, and in the products that we use. They can be categorized, described, organized, combined, manipulated and created. Therefore, an object-oriented view has come into picture for creation of computer software. An object-oriented approach to the development of software was proposed in late 1960s.

Object-Oriented development requires that object-oriented techniques be used during the analysis, and implementation of the system. This methodology asks the analyst to determine what the objects of the system are, how they behave over time or in response to events, and what responsibilities and relationships an object has to other objects. Object-oriented analysis has the analyst look at all the objects in a system, their commonalties, difference, and how the system needs to manipulate the objects.

The Object Oriented Methodology of Building Systems takes the objects as the basis. For this, first the system to be developed is observed and analyzed and the requirements are defined as in any other method of system development. Once this is done, the objects in the required system are identified.

In simple terms, Object Modeling is based on identifying the objects in a system and their interrelationships. Once this is done, the coding of the system is done. Object Modeling is somewhat similar to the traditional approach of system designing, in that it also follows a sequential process of system designing but with a different approach.

7.1.11.2 Advantages of Object Oriented Methodology

Object Oriented Methodology closely represents the problem domain. Because of this, it is easier to produce and understand designs. The objects in the system are immune to

requirement changes. Therefore, allows changes more easily.

Object Oriented Methodology designs encourage more re-use. New applications can use the existing modules, thereby reduces the development cost and cycle time. Object Oriented Methodology approach is more natural. It provides nice structures for thinking and abstracting and leads to modular design.

7.1.12 Coding Standards

To make the system coding easy, easy to remember and reducing the chances of errors some techniques are used at the time of coding of the application which is called coding system. The coding system which we adopted during the coding is explained as follows:

- File names follow Pascal Casing where First characters of all words are in upper case and other characters are in lower case.
- Every file name has a suffix like DAO, DAOImpl, BO, VO
  Example: PotholeDAO.java
  PotholeBO.java
- Method names are written like,
  Example: insertPotholeDtls()
- Variable names follow **Camel Casing** where First characters of all words, except the first word are in upper case and other characters are in lower case.
  Example: isUpdateSuccess

CHAPTER 8

TESTING

8.1 TESTING

Testing is just like quality assurance to review of software products and related documents for correctness, completeness, reliability, and maintainability. And it includes assurance that the system meets the specifications and requirements for its intended use and performance. The common view of the testing is to prove that there are no errors. System Testing is too much expensive as well as it is not possible for analyst to prove that software is free and clear of errors.

8.1.1 Code Testing

The code testing examines the logic of the program. To follow this testing we have developed test cases that result in every instruction in the program or module; that every path is specific combination of conditions that is handled by the program. However, code testing does not define the failure of the system. As well as this testing dose not determine that whether the system meets its specifications or does it determines whether all aspects are implemented

8.1.2 Unit testing

Unit testing focuses first on the modules, independently of one another to locate the errors in coding and logic that contained within that module alone. For example, Test cases needed to check user, whether user is authorized or not, as well as if uses is authorized then which permissions are granted and then checks whether user allows to do function with respect to the permissions assigned to him/her. We have performed unit testing from Bottom up, tested each components of the modules independently, than integrate all the components and tested each modules independently. After that we have merged all the modules and tested the whole system.

8.1.3 Specification Testing

Perform specification testing, we have to examine the specification-defines that the system should do and how it should do under various conditions. Then test cases are developed for each conditions or combinations of conditions. So we can determine whether the system performs according to its specified requirements.

8.1.4 System Testing

System testing does not test the software per second but rather the integration of each module in the system. The primarily concern is the compatibility of each individual module. Analysts try to find areas where module has been assigned different specifications for data length type and data element name. For example, one module may expect that data for No to be a number while other module expect it to be a character data item. The system may itself not report to this is an error.

8.1.5 Performance Test

When we are developing a design, their concerns are more on reports, inputs, and processing sequence than on performance time. Performance time testing is conducted prior to implementation to determine how long it takes to receive a response to an inquiry. We have used only one connection for the whole system, which also helpful to increase performance time.

8.1.6 Output consideration

All the report were produced & documented as per the general requirements of business organizations. The reports are in the required as per the usual requirements. The formats, depending on the need of organization, can vary as compared to designed formats.

8.2 TEST CASES AND TEST RESULTS

8.2.1 Register a Pothole Complaint

Sr. No	Test Case	Excepted Result	Test Result
1	Push Submit button	Acknowledgement Receipt of Pothole Complaint should be generated	Successful
2	Select by Area Details Radio Button	Combo Boxes should be auto-populated by the details(area, street and city)	Successful
3	Select by Postcode Details Radio Button	All combo boxes become uneditable textboxes.	Successful
4	Push check button after entering valid Postcode	All Combo Boxes of city, area and street should be auto populated.	Successful
5	Push check button after entering invalid Postcode	Proper Alert is shown.	Successful
6	Push Reset button	All fields are reset.	Successful
7	Push back button	Redirect to previous page	Successful

Table 8.2.1 Register a Pothole Complaint

8.2.2 Update Inspection

Sr. No	Test Case	Excepted Result	Test Result

1	Push update inspection button	Proper alert is shown and details should be updated	Successful

Table 8.2.2 Update Inspection

8.2.3 Generate Order

Sr. No	Test Case	Excepted Result	Test Result
1	Push Generate Order button	Order should be displayed	Successful

Table 8.2.3 Generate Order

8.2.4 Send to Contractor

Sr. No	Test Case	Excepted Result	Test Result
1	Push Send to Contractor button without any error	Proper alert is shown and details should send to contractor.	Successful
2	Push Send to Contractor button with error	Proper alert is shown.	Successful

Table 8.2.4 Send to Contractor

8.2.5 Submit repair details

Sr. No	Test Case	Excepted Result	Test Result

1	Push Submit button.	Work list of contractor is displayed and invoice details should be stored.	Successful
2	Push back button	Redirect to previous page	Successful

Table 8.2.5 Submit invoice details

8.2.6 Save and Approve Repair Details

Sr. No	Test Case	Excepted Result	Test Result
1	Push Save button	Proper alert is shown and details should be saved.	Successful
1	Push Approve button	Proper alert is shown and complaint should be approved.	Successful

Table 8.2.6 Save and Approve Repair Details

CHAPTER 9

USER MANUAL

1. Portal Login

Any citizen can login to the System using Login Page at Portal.

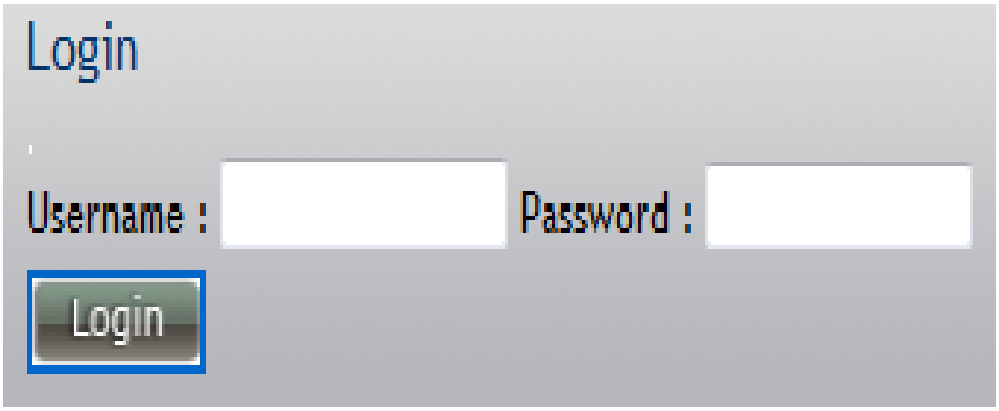


Fig 9.1 Portal Login

2. Portal Home Screen

After successful login to the system, home screen of portal is displayed.

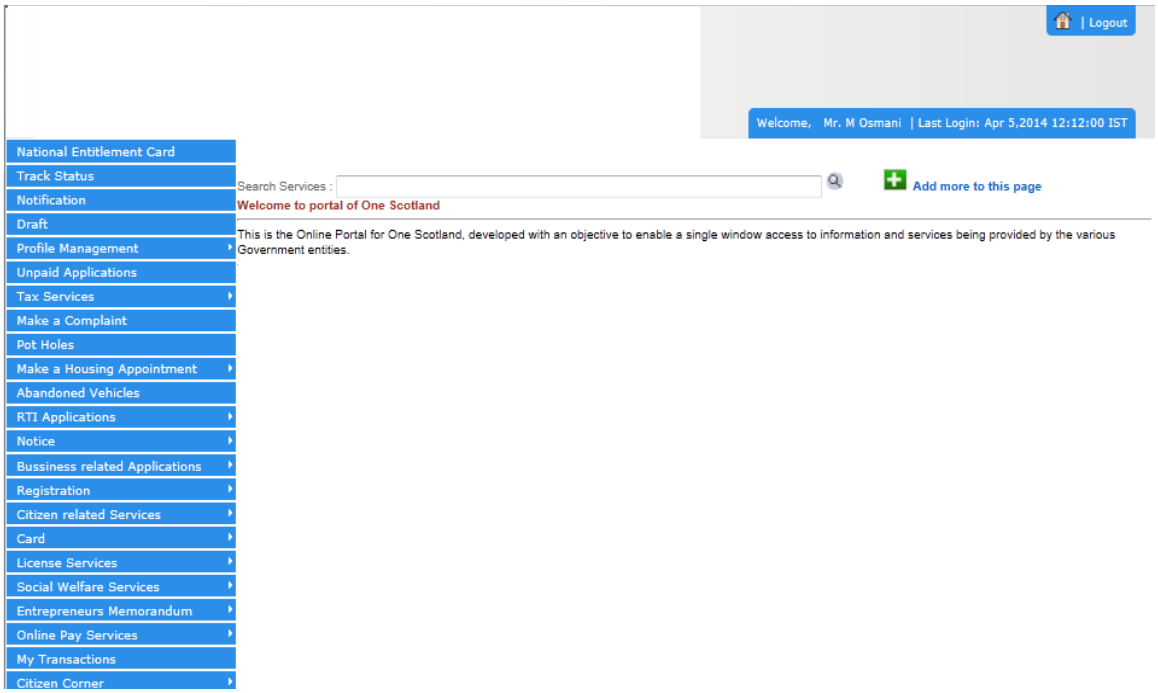


Fig 9.2 Portal Home Screen

3. Pothole address details by Area

After clicking on Pothole link which is displayed on the left panel of the screen, Pothole screen is displayed. Citizen can give Pothole address details by selecting ‘by area’ radio button and after that choose address from the list of City, area and street.

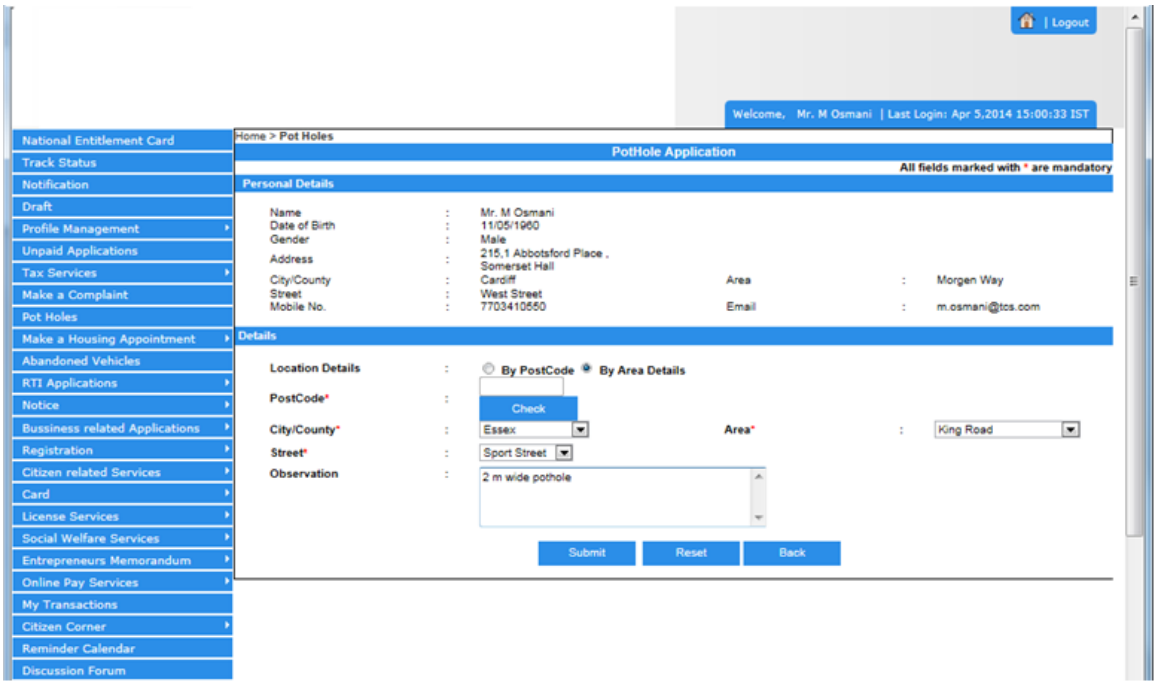


Fig 9.3 Pothole address details by Area

4. Pothole address details by Postcode

Citizen can give Pothole address details by selecting ‘by postcode’ radio button and then

give postcode and push ‘check’ button then city, street and area textboxes are auto populated with the values related with that postcode.

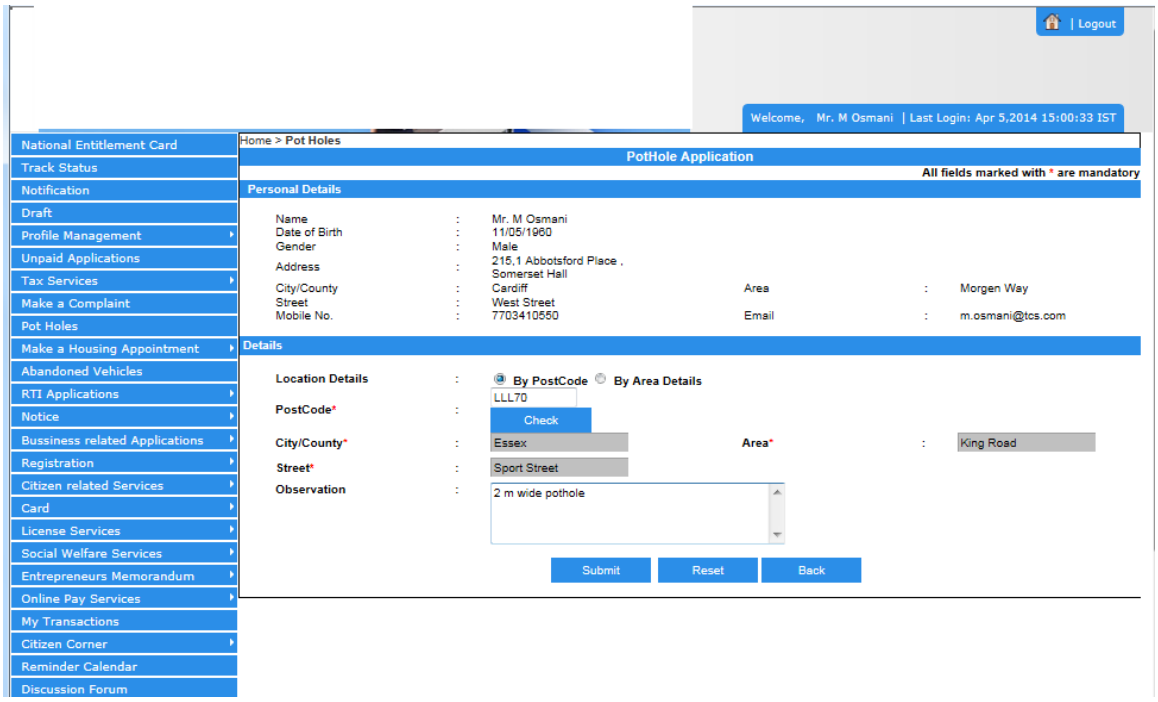


Fig 9.4 Pothole address details by Postcode

5. Acknowledgement receipt of Pothole complaint

After filling all details, Citizen can register a complaint by clicking on submit button and then acknowledgement receipt of that complaint is generated.

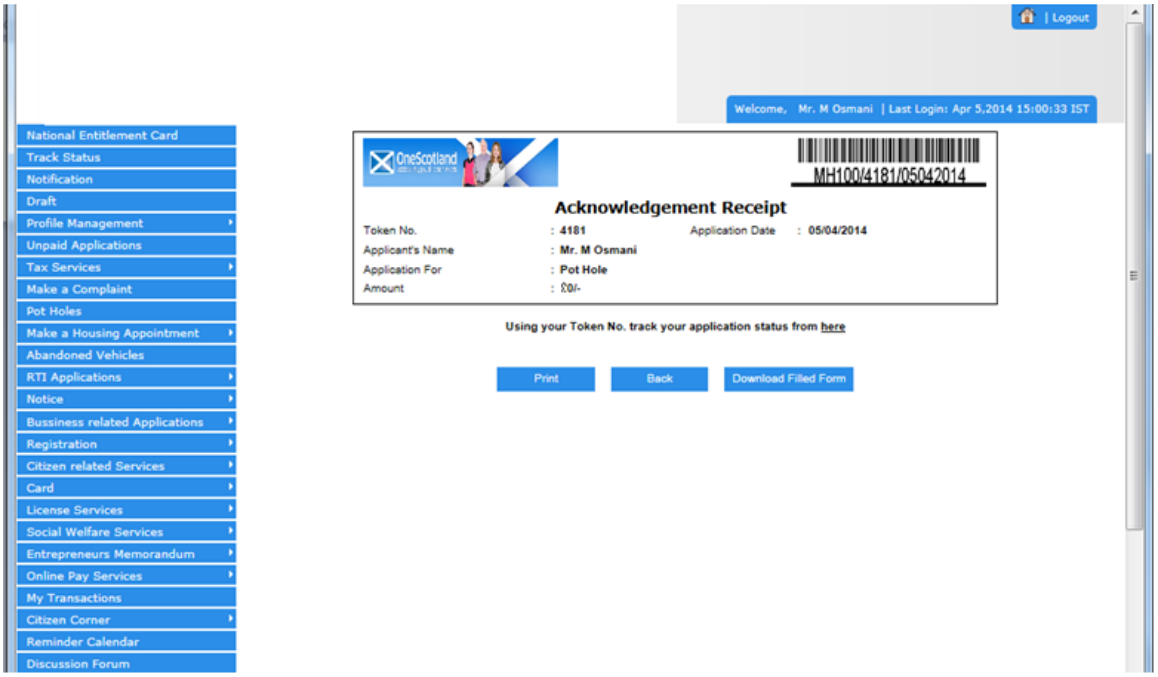


Fig 9.5 Acknowledgement receipt of Pothole complaint

6. Download Filled Form of Pothole Complaint

Citizen can see his/her filled form by clicking 'Download Filled Form' button.

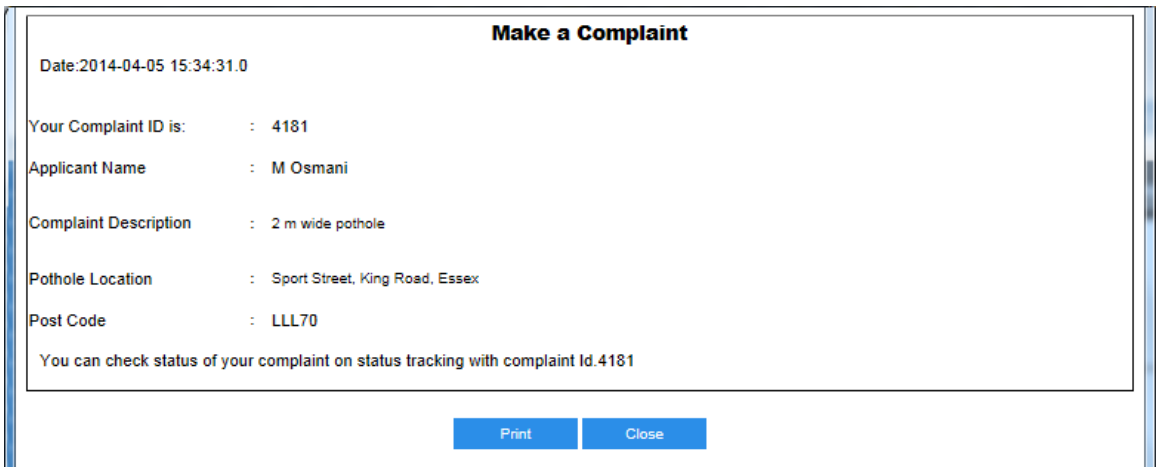


Fig 9.6 Download Filled Form of Pothole Complaint

7. List of Complaints with pending status

Citizen can track the status of his/her complaint by clicking on 'Track Status' link on the left panel of the screen. Then list of complaints is displayed.

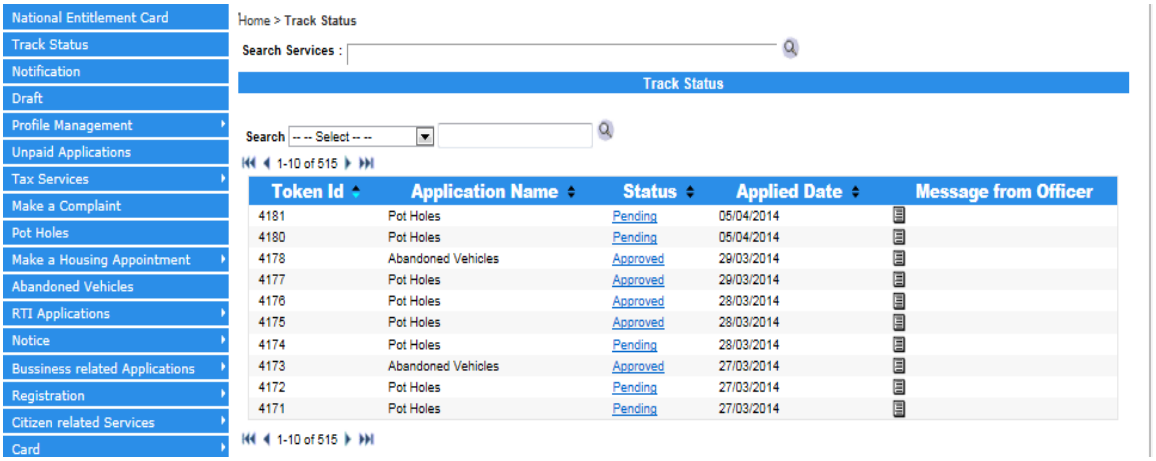


Fig 9.7 List of Complaints with pending status

8. Status

Citizen sees his/her status.

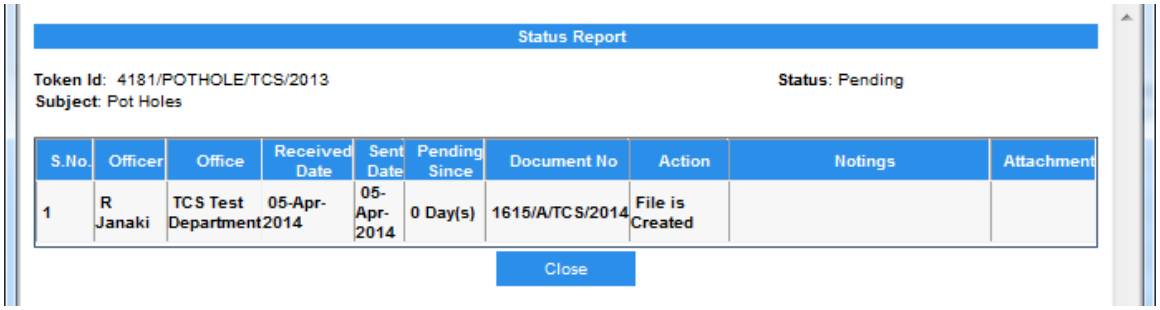


Fig 9.8 Status

9. Backend Login

At backend side, Clerk and Inspector can login to the backend system through the login page of backend.

Fig 9.9 Backend Login

10. Clerk Home Page

After successful login to the system, home screen of clerk is displayed.

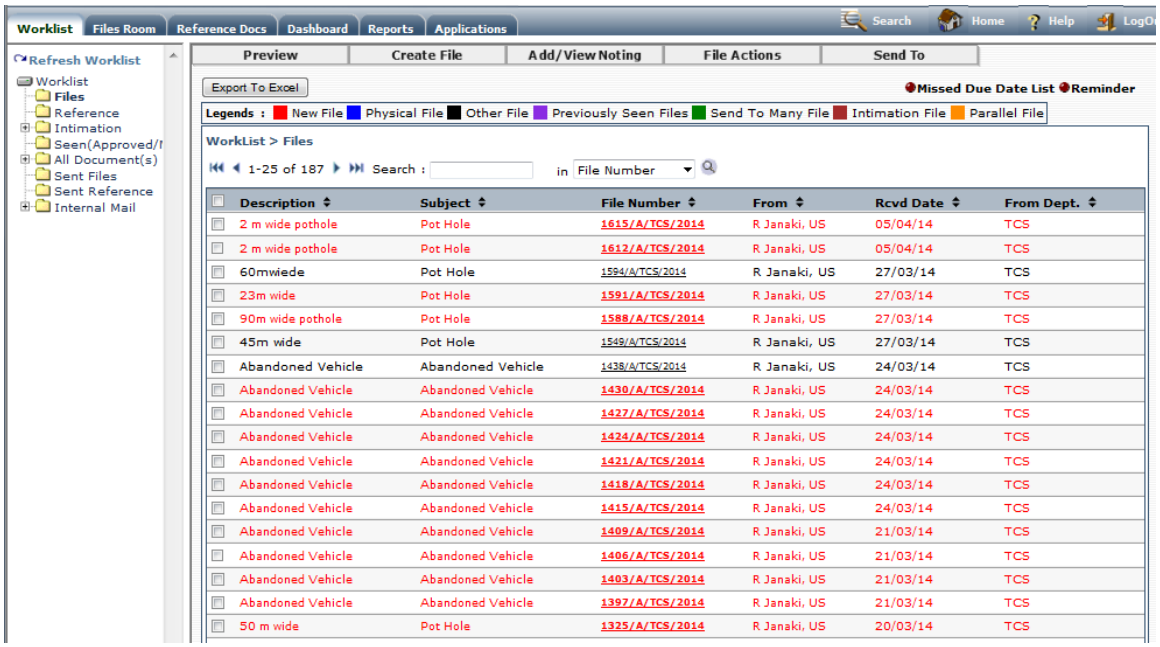


Fig 9.10 Clerk Home Page

11. File forward to Inspector

Clerk forwards file to Inspector.

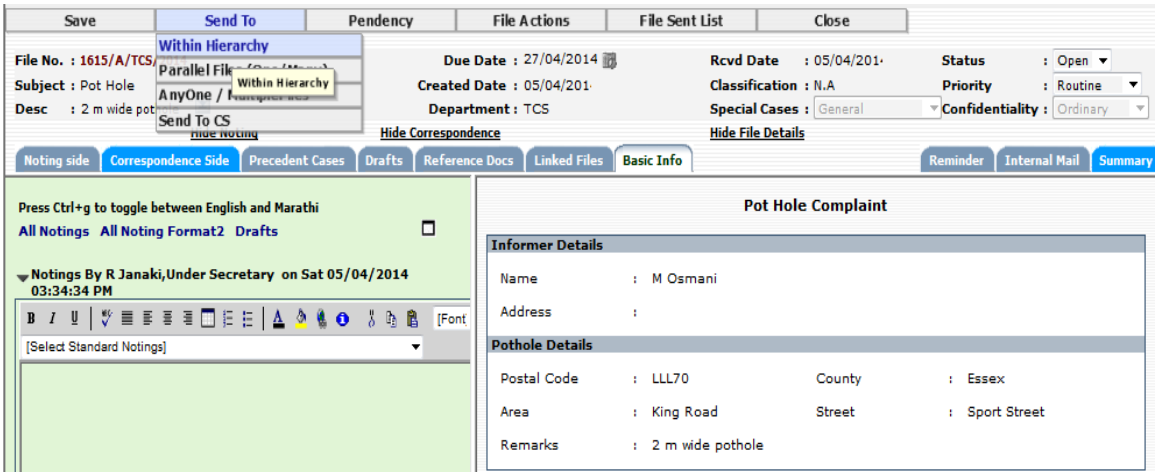


Fig 9.11 File forward to Inspector

12. File has been sent to Status

Citizen tracks status of file that has been sent to inspector Sanat Jethva.

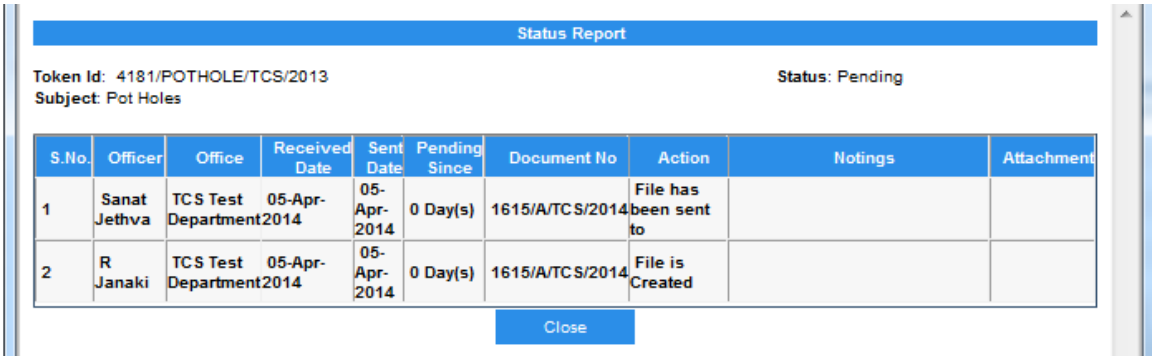


Fig 9.12 File has been sent to Status

13. Inspector Home Page

After successful login to the system, home screen of inspector is displayed.

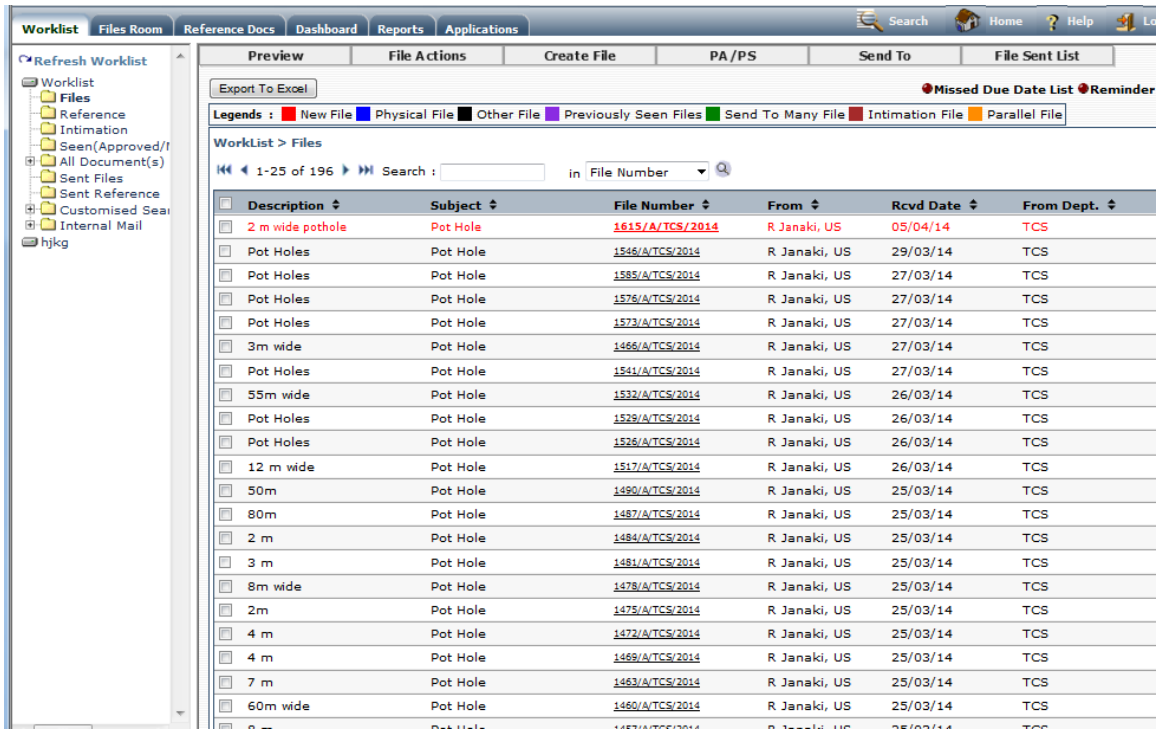


Fig 9.13 Inspector Home Page

14. Pothole Complaint Form

Inspector clicks on Pothole file and Pothole complaint from citizen is displayed on the screen with all pothole details entered by citizen at portal then inspector inspects the pothole and then fill the Inspection Details to that Pothole file and then click on update

Inspection button to save the Inspection details.

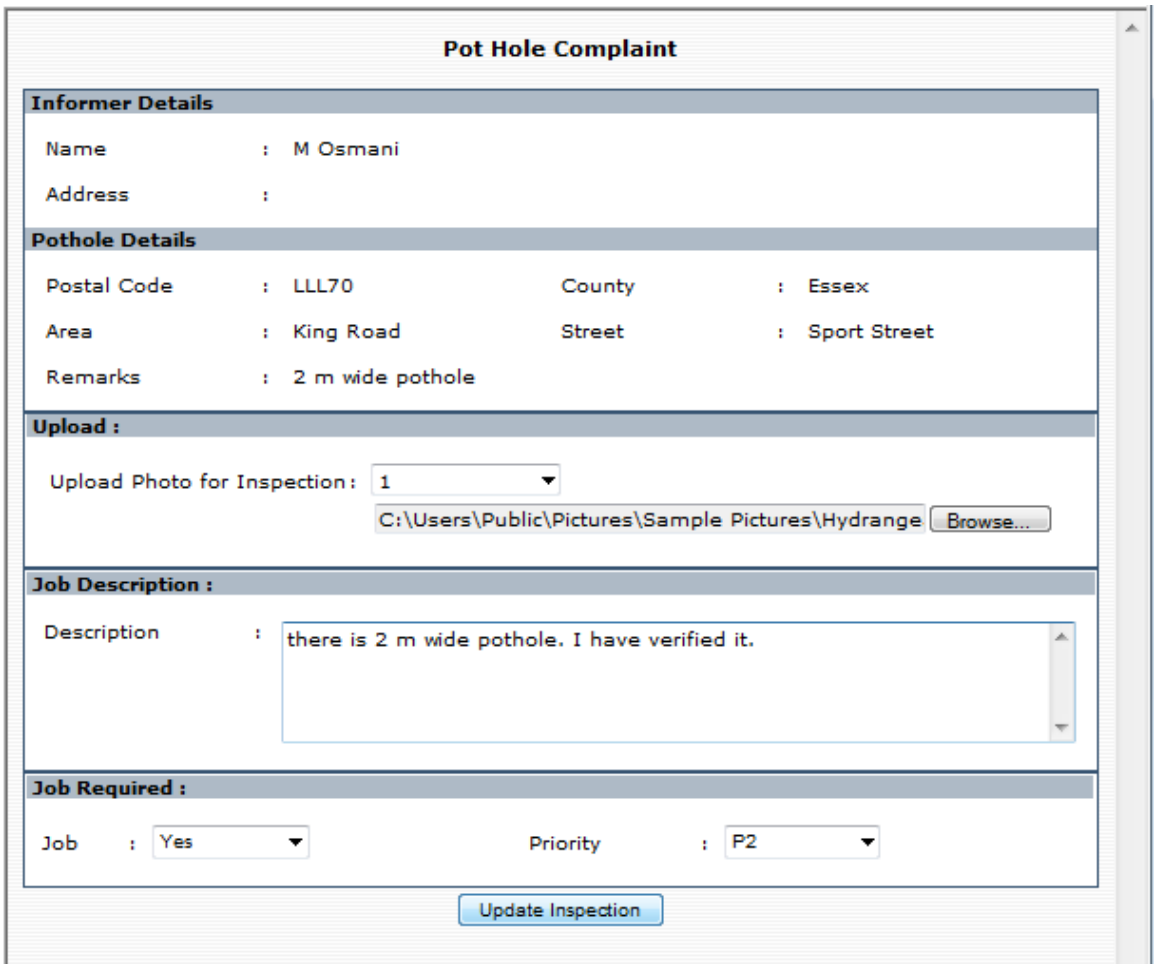


Fig 9.14 Pothole Complaint Form

15. Data saved successfully

Data saved successfully message box displayed.

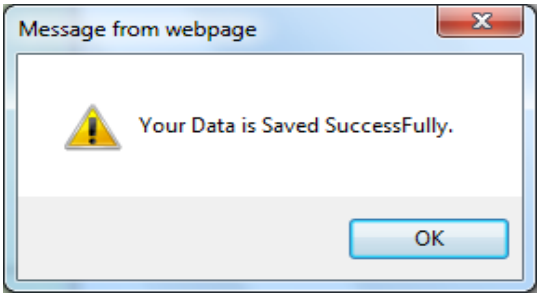


Fig 9.15 Data saved successfully

16. Pothole complaint form with generate order and send to contractor buttons

After clicking update Inspection, two more button is displayed generate order and send to contractor.

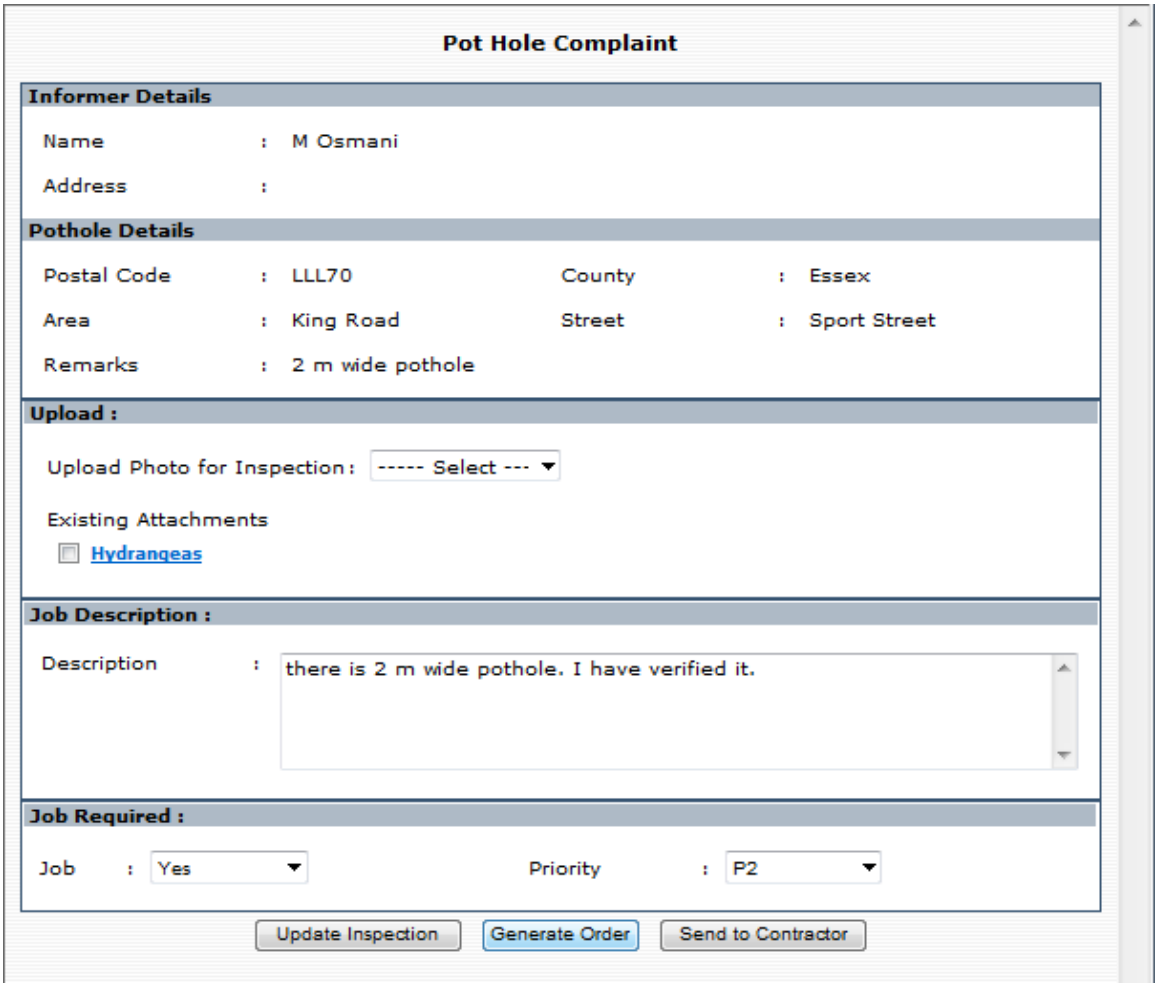


Fig 9.16 Pothole complaint form with generate order and send to contractor buttons

17. Generate Order for Contractor

When clicking on ‘Generate Order’ the order for contractor to repair the pothole is generated. And after that using ‘Send to Contractor’ button, Inspector will send the

Pothole Repairing Order to Contractor.

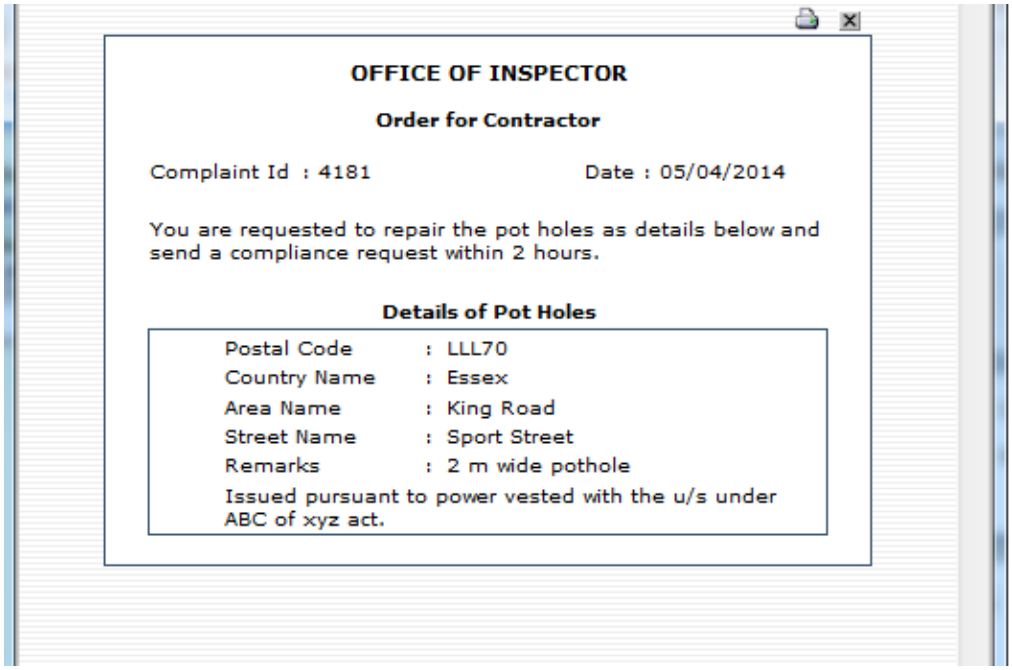


Fig 9.17 Generate Order for Contractor

18. Sent to contractor successfully

Request has been sent to contractor successfully message box displayed.

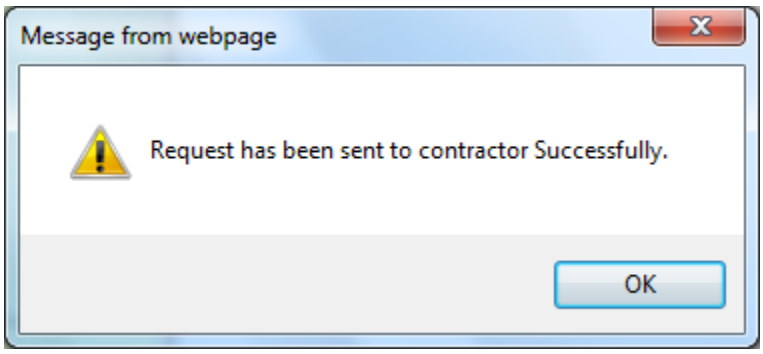


Fig 9.18 Sent to contractor successfully

19. Contractor Inbox

Contractor is at Portal side and can login to the system from the same login page as citizen. After successful login, work list (Inbox) of that Contractor is displayed. It contains the list of orders from the Inspector.

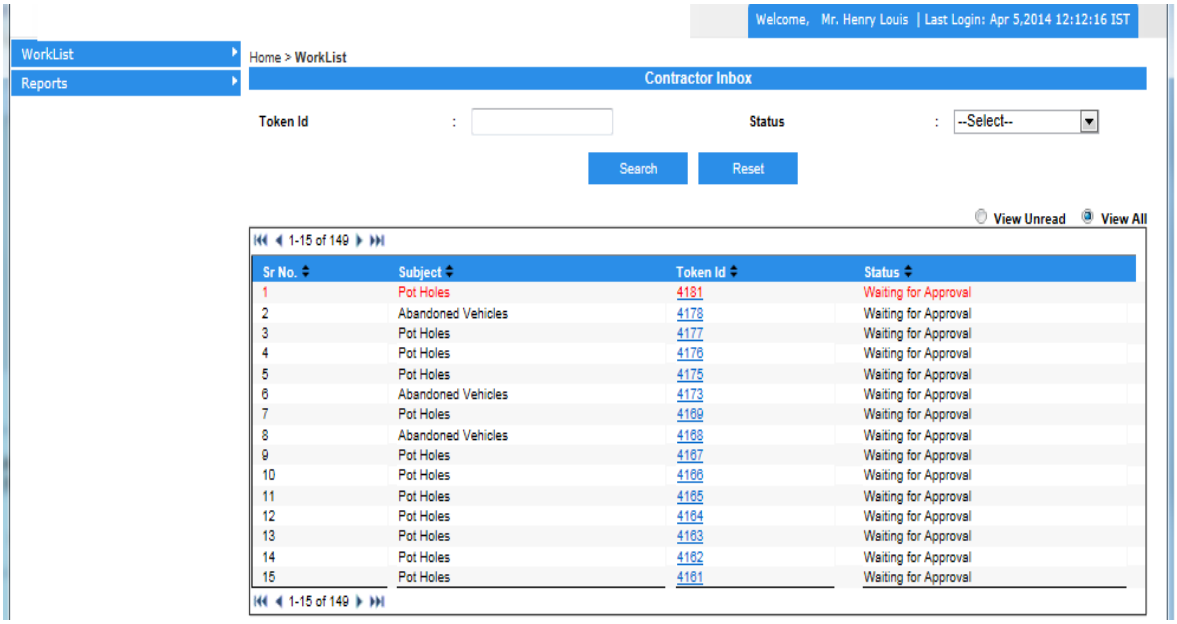


Fig 9.19 Contractor Inbox

20. Contractor submits Repairing Details

By clicking the File, it will going to display the order from the Inspector and details those contractor needs to fill after repairing pothole. After filling all details, Contractor submits that details and then that will being sent to Inspector.

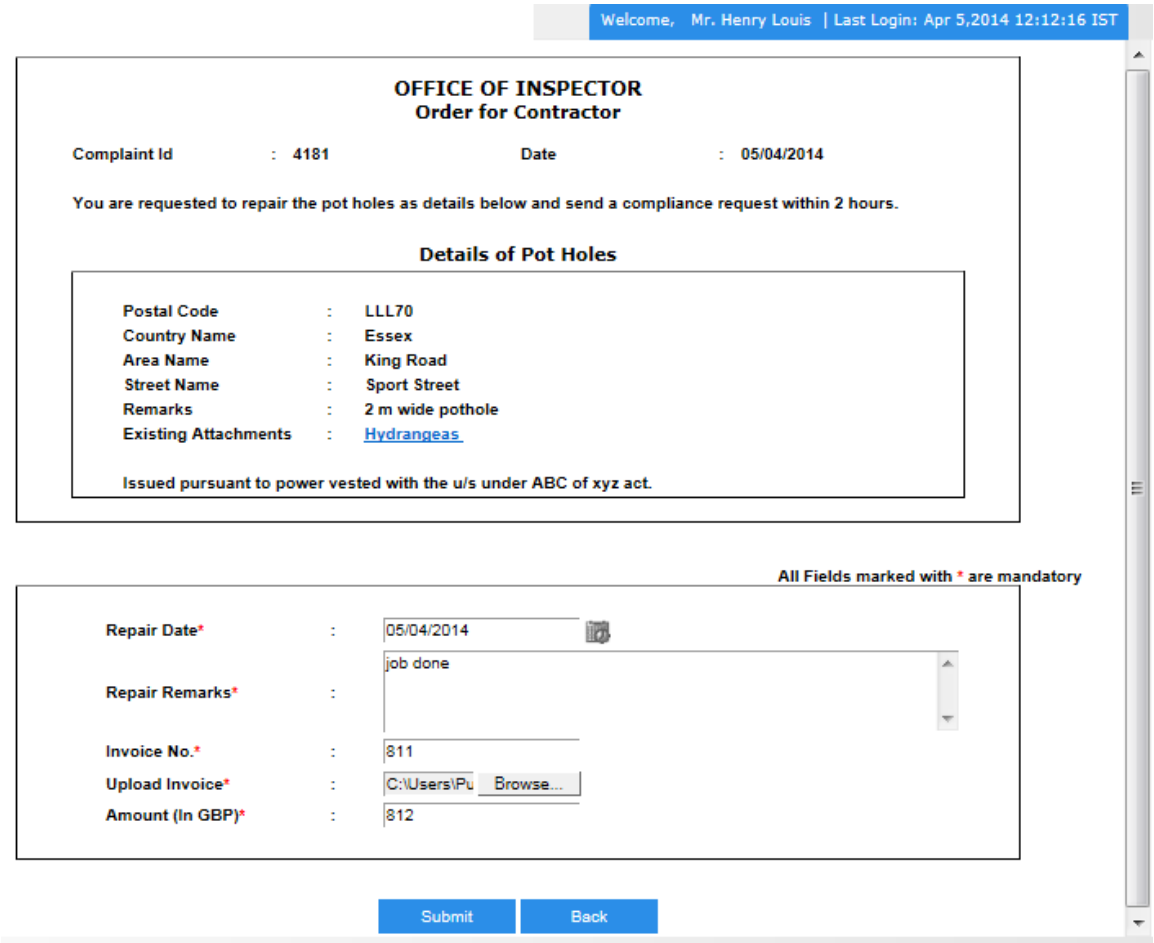


Fig 9.20 Contractor submits Repairing Details

21. View Invoice Link

After submitting Repairing details by Contractor, Inspector clicks on that file from his Inbox and one new link ‘View Invoice Contractor’ is displayed on the top-right part of screen. This link is displayed if and only if contractor has submitted the repairing details for that pothole complaint.

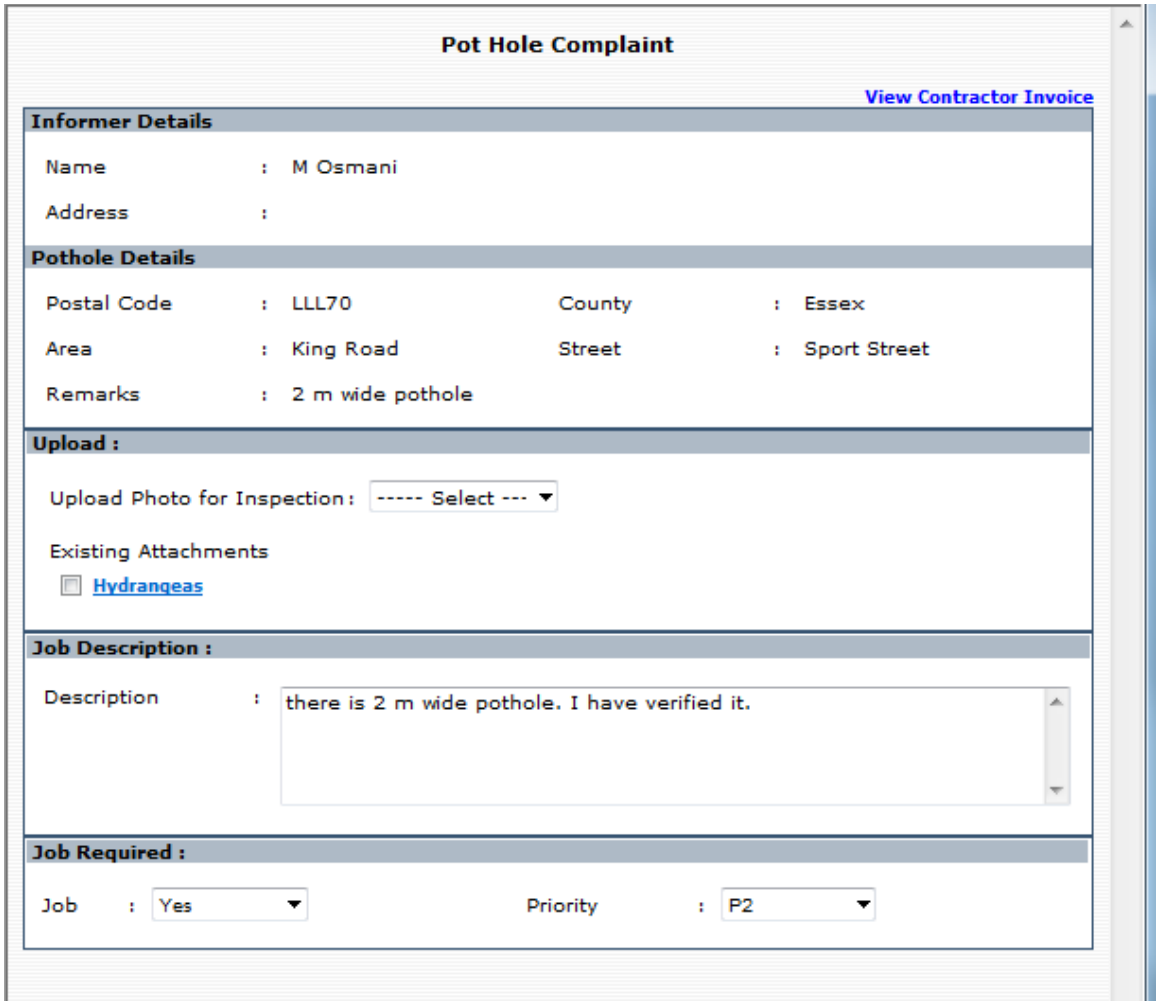


Fig 9.21 View Invoice Link

22. Repair Details Save by Inspector

Inspector clicks on that link and fill up the approved amount and remarks and then click on save to save the details. After that Inspector can approve the Pothole complaint also. After approve by Inspector, Citizen can check the status of complaint, it is being 'approved' instead of 'pending'.

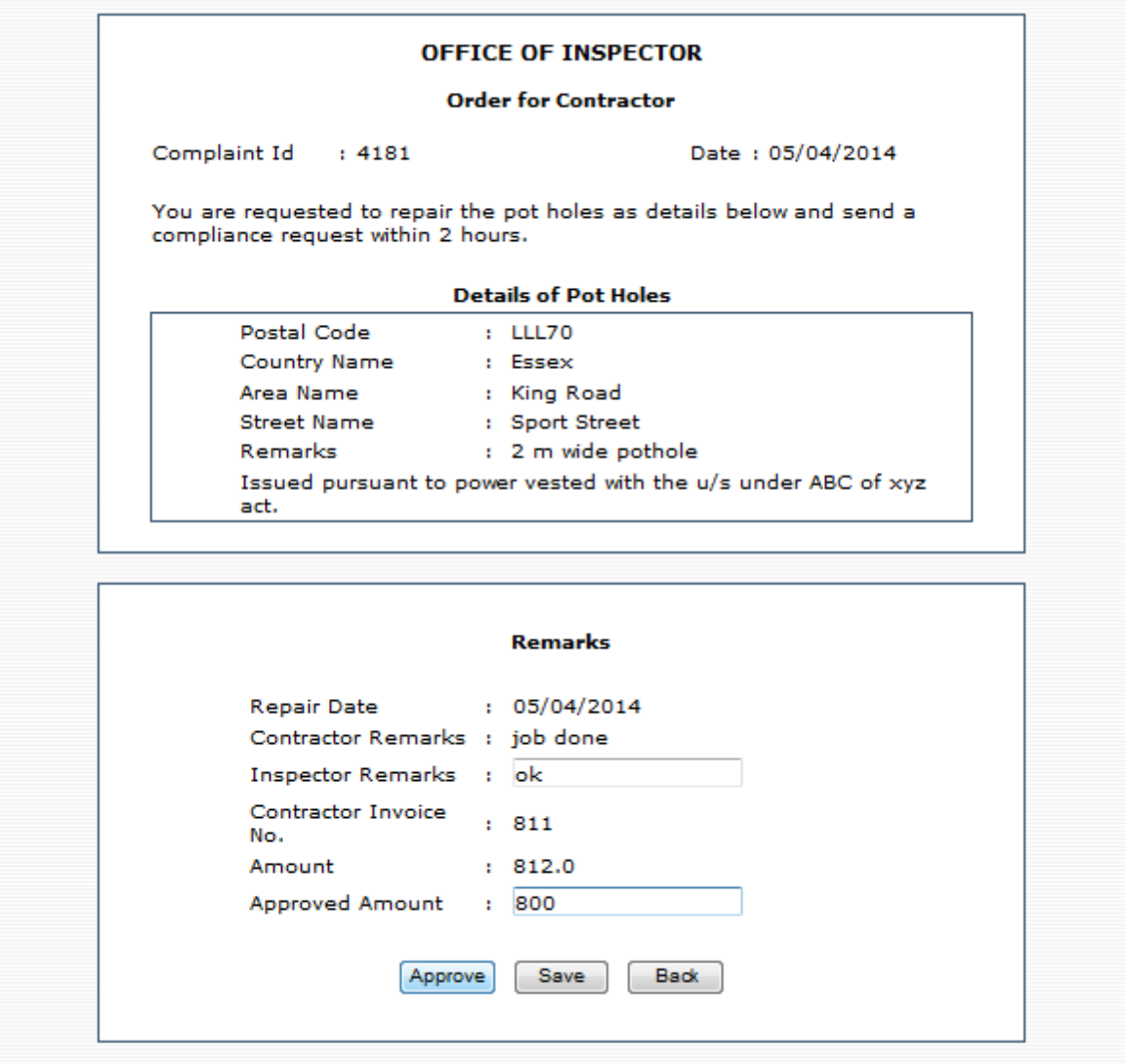


Fig 9.22 Repair Details Save by Inspector

23. Do you want to approve

Do you want to approve message box displayed

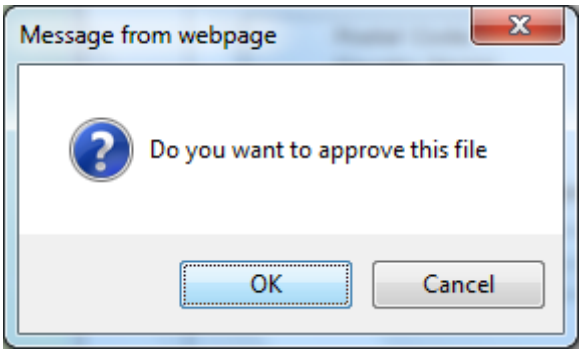


Fig 9.23 Do you want to approve

24. Approve

Approve message box displayed

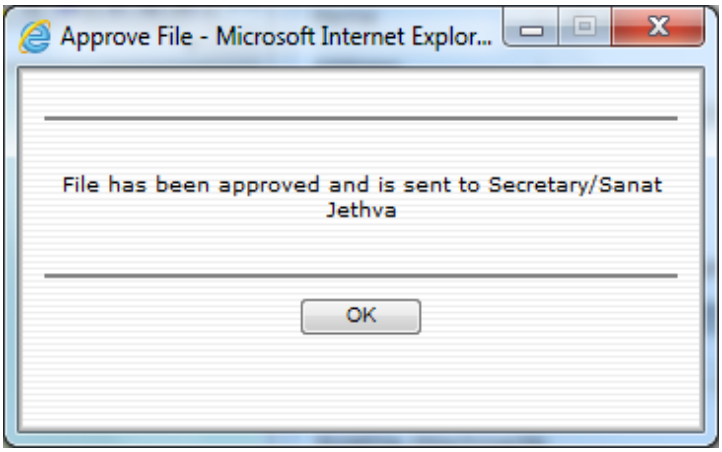


Fig 9.24 Approve

25. List of Complaints with approved status

Citizen can track the status of his/her complaint by clicking on ‘Track Status’ link on the left panel of the screen. Then list of complaints is displayed.

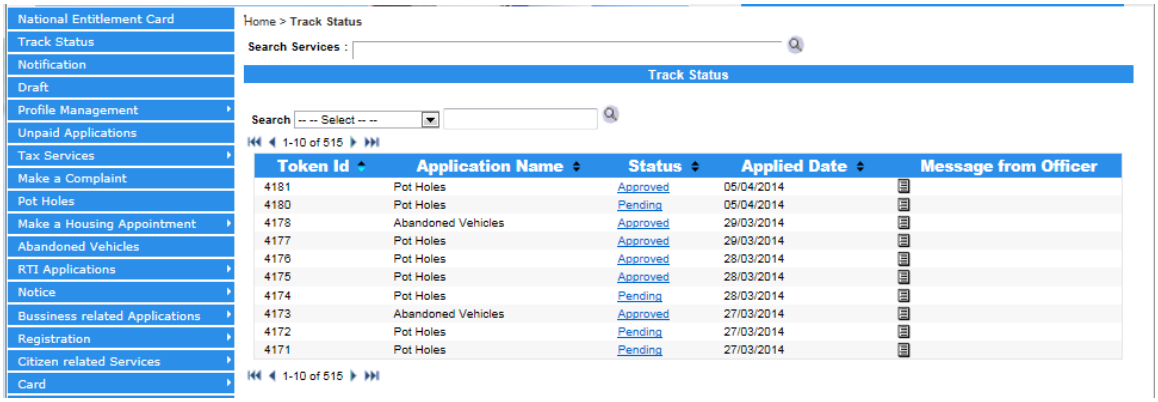


Fig 9.25 List of Complaints with approved status

26. Approved Status

Citizen tracks status of file that has been approved.

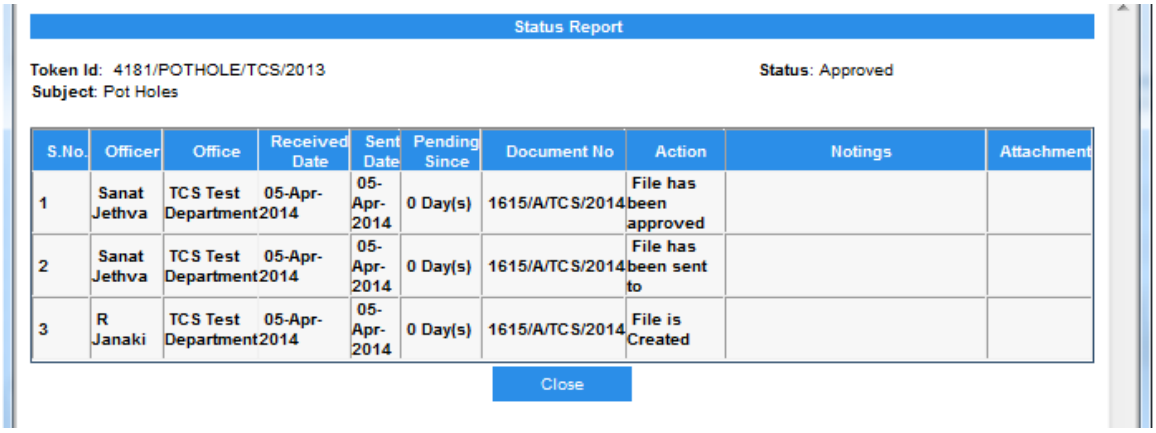


Fig 9.26 Approved Status

CHAPTER 10

CONCLUSION AND FUTURE WORK

10.1 CONCLUSION

Online Pothole Repairing System provides facility to citizen that he/she does not need to go to pothole repairing office and make a complaint. If citizen finds any pothole then can directly make a complaint online and complaint is being solved and citizen can track the status of complaint also. It saves the time of all people involving in pothole repairing system.

10.2 FUTURE WORK

In this system, citizen can register a complaint and complaint is resolved but after some time if pothole is there at that same place then citizen can reopen that same complaint and he/she does not need to fill again all the details, just need to send one notification to the inspector that pothole is not repaired or again pothole is made.

CHAPTER 11

ANNEXURE

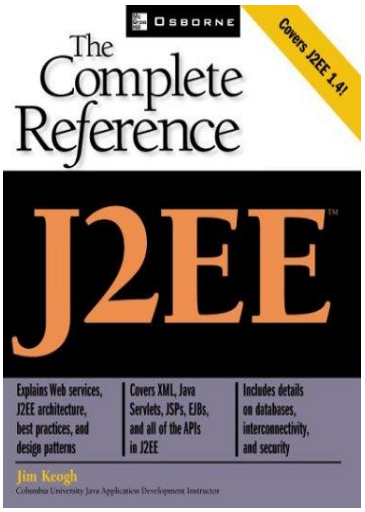
11.1 GLOSSARY OF TERMS AND ABBREVIATIONS:

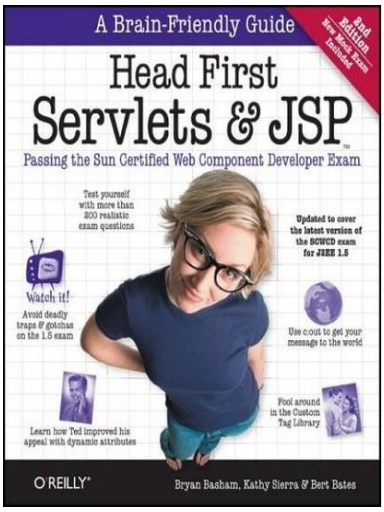
- Pothole
    - A deep natural underground cave formed by the erosion of rock, especially by the action of water.
    - A depression or hollow in a road surface caused by wear or subsidence.
- BO : Business Object
- DAO : Data Access Object
- DAOImpl : Data Access Object Implementation
- VO : Value Object

11.2 REFERENCES

1. TCS Training Guidelines
2. Online tutorials for hibernate, JDBC and web services.
3. Study the framework used in backend and portal of the system.
4. Online tutorials for converting attachment to byte array and store attachment to the database and how to send attachment via web service.
5. Getting online information on different Exceptions and syntaxes.

11.2.1 Books





The Complete Reference J2EE
Jim Keogh
Tata McGraw-Hill

Head First Servlets & JSP,
Kathy Sierra, Bert Bates, Bryan Basham
O'Reilly Publication

11.2.2 Referred websites:

- www.w3schools.com
- www.tutorialspoint.com
- www.stackoverflow.com
- www.javatpoint.com

11.3 TOOLS AND TECHNOLOGY

11.3.1 Hibernate (Front-End Side):

Hibernate ORM (Hibernate in short) is an object-relational mapping library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions.

Hibernate is a free software that is distributed under the GNU Lesser General Public License.

Hibernate's primary feature is mapping from Java classes to database tables (and from Java data types to SQL data types). Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from manual result set handling and object conversion. Applications using Hibernate are portable to supported SQL databases with little performance overhead.

Mapping in Hibernate:

Mapping Java classes to database tables is accomplished through the configuration of an XML file or by using Java Annotations. When using an XML file, Hibernate can generate skeleton source code for the persistence classes. This is unnecessary when annotations are used. Hibernate can use the XML file or the annotations to maintain the database schema.

Facilities to arrange one-to-many and many-to-many relationships between classes are provided. In addition to managing associations between objects, Hibernate can also manage reflexive associations where an object has a one-to-many relationship with other instances of its own type.

Hibernate supports the mapping of custom value types. This makes the following scenarios possible:

1) Overriding the default SQL type that Hibernate chooses when mapping a column to a property.
2) Mapping Java Enum to columns as if they were regular properties.
3) Mapping a single property to multiple columns.

Hibernate Query Language (HQL):

Hibernate provides an SQL inspired language called Hibernate Query Language (HQL) which allows SQL-like queries to be written against Hibernate's data objects. Criteria Queries are provided as an object-oriented alternative to HQL. Criteria Query is used to modify the objects and provide the restriction for the objects.

11.3.2 JDBC (Back-End Side):

JDBC is a Java-based data access technology (Java Standard Edition platform) from Oracle Corporation. This technology is an API for the Java programming language that defines how a client may access a database. It provides methods for querying and updating data in a database. JDBC is oriented towards relational databases. A JDBC-to-ODBC bridge enables connections to any ODBC-accessible data source in the JVM host environment. The JDBC classes are contained in the Java package java.sql and javax.sql.

JDBC allows multiple implementations to exist and be used by the same application. The API provides a mechanism for dynamically loading the correct Java packages and registering them with the JDBC Driver Manager. The Driver Manager is used as a connection factory for creating JDBC connections.

JDBC connections support creating and executing statements. These may be update statements such as SQL's CREATE, INSERT, UPDATE and DELETE, or they may be query statements such as SELECT. Additionally, stored procedures may be invoked through a JDBC connection. JDBC represents statements using one of the following classes:

- Statement – the statement is sent to the database server each and every time.
- Prepared Statement – the statement is cached and then the execution path is pre-determined on the database server allowing it to be executed multiple times in an efficient manner.
- Callable Statement – used for executing stored procedures on the database.

Update statements such as INSERT, UPDATE and DELETE return an update count that indicates how many rows were affected in the database. These statements do not return any other information.

Query statements return a JDBC row result set. The row result set is used to walk over the result set. Individual columns in a row are retrieved either by name or by column number. There may be any number of rows in the result set. The row result set has metadata that describes the names of the columns and their types. There is an extension to the basic JDBC API in the javax.sql.
JDBC connections are often managed via a connection pool rather than obtained directly from the driver.

11.3.3 Servers:

11.3.3.1 *Apache Tomcat (Front-End Side):*

Apache Tomcat (or simply Tomcat, formerly also Jakarta Tomcat) is an open source web server and servlet container developed by the Apache Software Foundation (ASF). Tomcat implements the Java Servlet and the Java Server Pages (JSP) specifications from Sun Microsystems, and provides a "pure Java" HTTP web server environment for Java code to run in.

Apache Tomcat includes tools for configuration and management, but can also be configured by editing XML configuration files. Jasper is Tomcat's JSP Engine. Jasper parses JSP files to compile them into Java code as servlets (that can be handled by Catalina). At runtime, Jasper detects changes to JSP files and recompiles them.

11.3.3.2 *JBoss (Back-End Side):*

JBoss, is an application server authored by JBoss, now developed by Red Hat. JBoss known as 'WildFly' and is written in Java and is executable on top of the Java Platform, Enterprise Edition (Java EE), which is available cross-platform.

11.4 ABOUT THE ORGANISATION

Tata Consultancy Services (TCS) is one of the leading information technology companies in the world. With a workforce of over 74,000 professionals spread across more than 50 global delivery centers, it helps organizations stay ahead with new technology. Its clients include seven of the top ten corporations in the Fortune 500 list of the largest corporations in the United States challenges. With technical expertise and employing a flexible approach to client relationships, TCS offers its clients: consulting, IT services, business process outsourcing, infrastructure outsourcing, and engineering and industrial services.

Since its inception, the company has invested in new technologies, processes, and people in order to help its customers succeed. With inputs from its innovation labs and university alliances, and drawing on the expertise of key partners, TCS keeps clients up-to-date with new technology. This has helped the company meet various benchmarks of excellence in software development - it is the world's first organization to achieve an enterprise-wide Maturity Level 5 on quality improvement models.

Mission
To help customers achieve their business objectives, by providing Innovative, Best-in-class Consulting, IT solutions and services. We shall make it a joy for all stakeholders to work with us.

Values
Integrity, Leading change, Excellence, Respect for individual, and Learning & Sharing.

Locations
TCS is headquartered in Mumbai, and operates in more than 50 countries and has more than **170** offices across the world.

11.5 ABOUT THE COLLEGE

U. V. Patel College of Engineering is a hi-tech education institute in the campus **GANPAT VIDYANAGAR** near village Kherva, 45 km away from Gandhinagar and 65 km away from Ahmadabad. The dream of establishing a campus for professional courses came true by the generous support of Shree Ganpatbhai Ishwarlal Patel, an eminent settled in U.S.A., after whom the campus has been named as Ganpat Vidyanagar.

The campus is being developed by Mehsana District Education Foundation, a highly progressive trust, dedicated to the cause of impairing quality technical and professional education in North Gujarat region. A visionary educationalist and eminent industrialist Shree Anilbhai T. Patel with his diligent group of trustees is leading this movement.

The campus sprawling over 300 acres having natural surrounding with full if greenery, several gardens ,a pond, playground and plantation of over 40,000 trees provide refreshing and enliven academic environment on the campus. To conserve the energy and ensure light over the campus during the night with higher reliability, the entire campus is illuminated by solar street lights a unique feature.

Mission: - It shall be the constant endeavor of the Mehsana District Education Foundation to meet the educational needs of the youth in the area of professional studies and provide state-of-the-art learning opportunities along with inculcation of values of professional commitment and uprightness