FULL STACK DEVELOPMENT WITH MONGO DB

SRI VENKATESWARA COLLEGE OF ENGINEERING

Project submitted by:-

TEAM LEADER: SUDHARSHAN

TEAM MEMBERS:

PONALA PRASAD
MOTHUKURI HARSHA VARDHAN
VAMSHI
ONTERU SANDEEP

Table of Contents:-

- Introduction
- Description
- Scenario
- Source code
- Objectives
- Project Features
- Technology Stack
- Methodology
- Conclusion
- Outputs

Online Complaint Registration And Management System

About Online Complaint Registration And Management System:

INTRODUCTION:-

An online complaint registration and management system is a software application or platform that allows individuals or organizations to submit and track complaints or issues they have encountered. It can help optimize the complaint handling process and empower organizations to develop a safety management system to efficiently resolve customer complaints, while staying in line with industry guidelines and regulatory compliance obligations. It provides a centralized platform for managing complaints, streamlining the complaint resolution process, and improving customer satisfaction.

It consists of some key features which include:

1. User registration: Users can create accounts to submit complaints and track their progress.
2. Complaint submission: Users can enter details of their complaints, including relevant information such name, description of the issue, address etc.
3. Tracking and notifications: Users can track the progress of their complaints, view updates, and receive notifications via email or SMS when there are any changes or resolutions.
4. User can interact with the agent who has assigned the complaint.
5. Assigning and routing complaints: The system assigns complaints to the appropriate department or personnel responsible for handling them. It may use intelligent routing algorithms to ensure efficient allocation of resources.
6. Security and confidentiality: The system ensures the security and confidentiality of user data and complaint information through measures such as user authentication, data encryption, access controls, and compliance with relevant data protection regulations.

DESCRIPTION:-

The Online Complaint Registration and Management System is a user-friendly software solution designed to streamline the process of submitting, tracking, and resolving complaints or issues encountered by individuals or organizations. It provides a centralized platform for efficient complaint management, allowing users to securely register complaints, track their progress in real-time, and interact with assigned agents for issue resolution. With features such as automatic notifications, intelligent complaint routing, and robust security measures, this system ensures timely and effective handling of complaints while prioritizing user Details.

SCENARIO:-

Scenario: John, a customer, recently encountered a problem with a product he purchased online. He notices a defect in the item and decides to file a complaint using the Online Complaint Registration and Management System.

1. **User Registration and Login:**
    - John visits the complaint management system's website and clicks on the "Sign Up" button to create a new account.
    - He fills out the registration form, providing his full name, email address, and a secure password.
    - After submitting the form, John receives a verification email and confirms his account.
    - He then logs into the system using his email and password.
2. **Complaint Submission:**
    - Upon logging in, John is redirected to the dashboard where he sees options to register a new complaint.
    - He clicks on the "Submit Complaint" button and fills out the complaint form.
    - John describes the issue in detail, attaches relevant documents or images showcasing the defect, and provides additional information such as his contact details and the product's purchase date.
    - After reviewing the information, John submits the complaint.
3. **Tracking and Notifications:**

    - After submitting the complaint, John receives a confirmation message indicating that his complaint has been successfully registered.
    - He navigates to the "My Complaints" section of the dashboard, where he can track the status of his complaint in real-time.
    - John receives email notifications whenever there is an update on his complaint, such as it being assigned to an agent or its resolution status.
4. **Interaction with Agent:**
    - A customer service agent, Sarah, is assigned to handle John's complaint.
    - Sarah reviews the details provided by John and contacts him through the system's built-in messaging feature.
    - John receives a notification about Sarah's message and accesses the chat window to communicate with her.
    - They discuss the issue further, and Sarah assures John that the company will investigate and resolve the problem promptly.
5. **Resolution and Feedback:**
    - After investigating the complaint, the company identifies the defect in the product and offers John a replacement or refund.
    - John receives a notification informing him of the resolution, along with instructions on how to proceed.
    - He provides feedback on his experience with the complaint handling process, expressing his satisfaction with the prompt resolution and courteous service provided by Sarah.
6. **Admin Management:**
    - Meanwhile, the system administrator monitors all complaints registered on the platform.
    - The admin assigns complaints to agents based on their workload and expertise.
    - They oversee the overall operation of the complaint management system, ensuring compliance with platform policies and regulations.

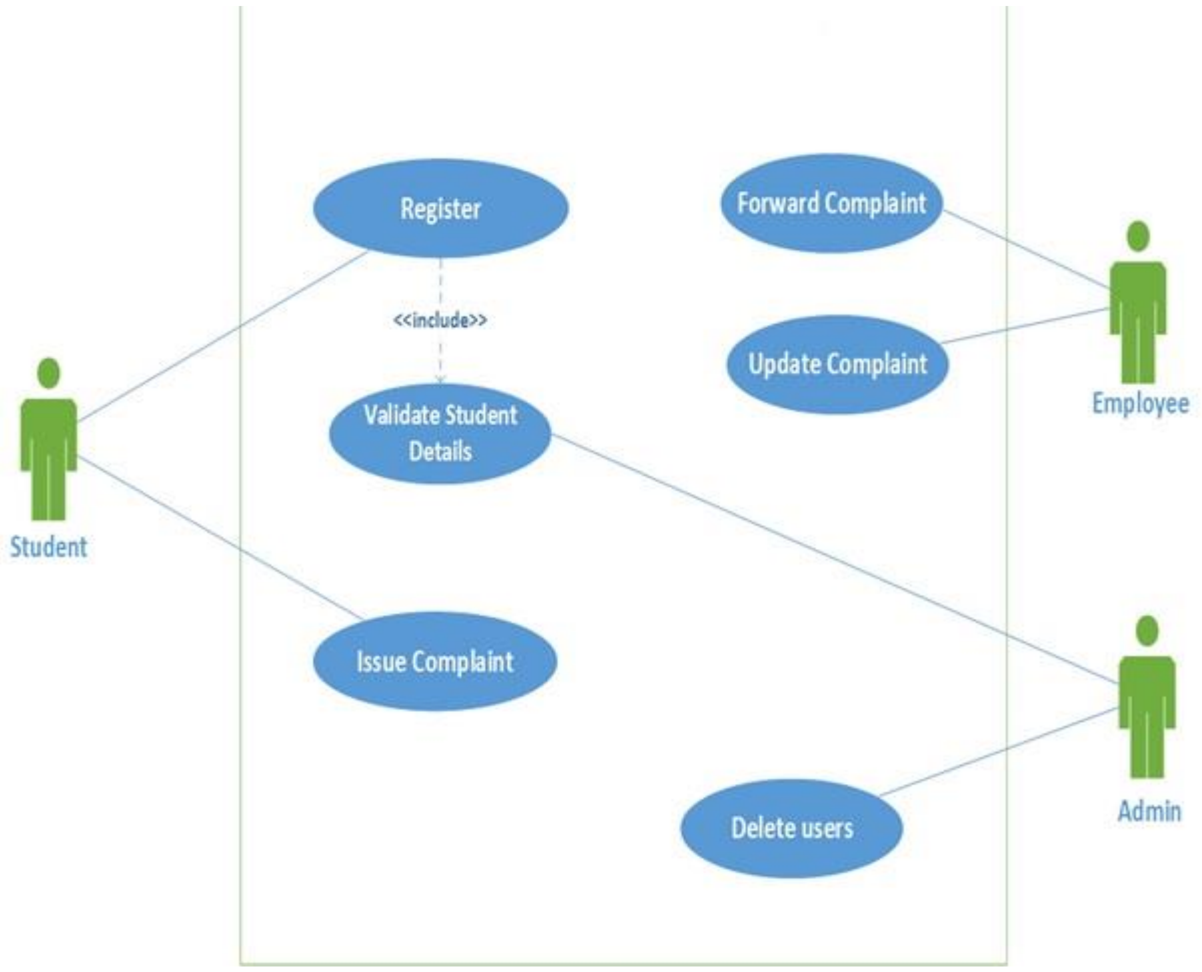
An online complaint registration and management system is a software application designed to facilitate the submission, tracking, and resolution of complaints or grievances raised by users or customers. Here's a breakdown of its key components and functionalities:

User Interface: The system typically has a user-friendly interface accessible through web browsers or mobile apps. Users can register complaints, track their status, and communicate with customer service representatives.

Complaint Submission: Users can submit complaints through an online form. The form may require specific details such as the nature of the complaint, relevant dates, and any supporting documents or evidence.

Complaint Tracking: Once a complaint is submitted, users receive a unique identifier or reference number. They can use this number to track the progress of their complaint, view updates, and check its current status.

Workflow Management: The system manages the workflow of complaints, assigning them to relevant departments or personnel based on predefined rules or criteria. It ensures that complaints are handled promptly and efficiently.



Automated Notifications: Users receive automated notifications via email or SMS at various stages of the complaint resolution process. Notifications may include acknowledgement of receipt, updates on the status, and closure of the complaint.

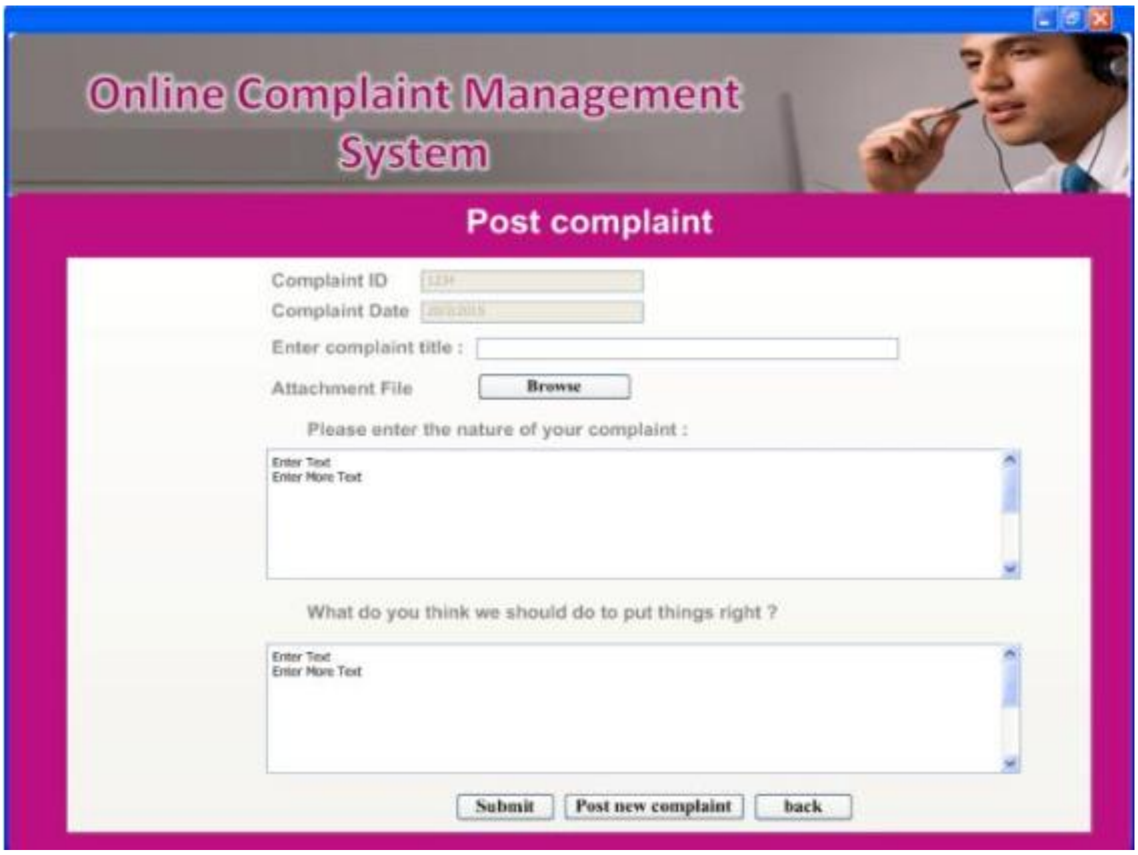
Escalation Mechanism: In cases where a complaint remains unresolved within a specified timeframe or requires further attention, the system can escalate it to higher authorities or senior management for review and intervention.

Analytics and Reporting: The system generates reports and analytics to monitor key performance indicators (KPIs), such as complaint resolution times, types of complaints received, and customer satisfaction levels. These insights help identify trends, areas for improvement, and measure the effectiveness of complaint handling processes.

Integration: The system may integrate with other internal systems such as customer relationship management (CRM) software, helpdesk tools, and databases to access relevant customer information and streamline the resolution process.

Security and Data Privacy: Strong security measures are implemented to protect sensitive customer information and ensure compliance with data privacy regulations. This includes encryption of data, access controls, and regular security audits.

Feedback Mechanism: After a complaint is resolved, users may be prompted to provide feedback on their experience with the complaint handling process. This feedback helps identify areas for improvement and enhances overall customer satisfaction.



Online Complaint Management System

Post complaint

Complaint ID 1234

Complaint Date [illegible]

Enter complaint title :

Attachment File

Please enter the nature of your complaint :

Enter Text
Enter More Text

What do you think we should do to put things right ?

Enter Text
Enter More Text

SOURCE CODE:-

```
const mongoose = require('mongoose')

const dbconnect = require('../db')


//Call the db to connect the mongo db

dbconnect()


// Complaint Schema

const ComplaintMappingSchema = mongoose.Schema({

    complaintID: {

        type: String,

        required: true

    },

    engineerName: {

        type: String,

        required: true

    },

});


const ComplaintMapping = module.exports = mongoose.model('ComplaintMapping', ComplaintMappingSchema);


module.exports.registerMapping = function (newComplaintMapping, callback) {
    newComplaintMapping.save(callback);
}
const mongoose = require('mongoose')
const dbconnect = require('../db')


//Call the db to connect the mongo db
dbconnect()


// Complaint Schema
const ComplaintSchema = mongoose.Schema({
    name: {
        type: String
    },
    email: {
        type: String
    },
    contact: {
        type: String
    },
    desc: {
        type: String
    }
});
```

```
const Complaint = module.exports = mongoose.model('Complaint',
ComplaintSchema);


module.exports.registerComplaint = function (newComplaint, callback) {
    newComplaint.save(callback);
}


module.exports.getAllComplaints = function(callback){
    Complaint.find(callback);
  }
const mongoose = require('mongoose')
const dbconnect = require('../db')
const bcrypt = require('bcryptjs');


//Call the db to connect the mongo db
dbconnect()


// User Schema
const UserSchema = mongoose.Schema({
    name: {
        type: String
    },
    username: {
        type: String,
        unique: true,
        required: true
    },
    email: {
        type: String
    },
    password: {
        type: String,
        required: true
    },
    role: {
        type: String
    }
});

const User = module.exports = mongoose.model('User', UserSchema);

module.exports.registerUser = function (newUser, callback) {
    bcrypt.genSalt(10, (err, salt) => {
        bcrypt.hash(newUser.password, salt, (err, hash) => {
            if (err) {
                console.log(err);
            }
            newUser.password = hash;
            newUser.save(callback);
        });
    });
}

module.exports.getUserByUsername = function(username, callback){
    const query = {username: username}
    User.findOne(query, callback);
}

module.exports.getUserById = function(id, callback){
    User.findById(id, callback);
}

module.exports.comparePassword = function(candidatePassword, hash, callback){
    bcrypt.compare(candidatePassword, hash, (err, isMatch) => {
        if(err) throw err;
        callback(null, isMatch);
    });
}
```

```
module.exports.getEngineer = function(callback){
    const query = {role: "jeng"}
    User.find(query, callback);
}
const express = require('express');
const router = express.Router();
const passport = require('passport');
const LocalStrategy = require('passport-local').Strategy;

let User = require('../models/user');
let Complaint = require('../models/complaint');
let ComplaintMapping = require('../models/complaint-mapping');

// Home Page - Dashboard
router.get('/', ensureAuthenticated, (req, res, next) => {
    res.render('index');
});

// Login Form
router.get('/login', (req, res, next) => {
    res.render('login');
});
```

```
// Register Form

router.get('/register', (req, res, next) => {

    res.render('register');

});



// Logout

router.get('/logout', ensureAuthenticated,(req, res, next) => {

    req.logout();

    req.flash('success_msg', 'You are logged out');

    res.redirect('/login');

});



// Admin

router.get('/admin', ensureAuthenticated, (req,res,next) => {

    Complaint.getAllComplaints((err, complaints) => {

        if (err) throw err;



        User.getEngineer((err, engineer) => {

            if (err) throw err;



            res.render('admin/admin', {

                complaints : complaints,
                engineer : engineer,
            });
        });
    });
});



// Assign the Complaint to Engineer
router.post('/assign', (req,res,next) => {
    const complaintID = req.body.complaintID;
    const engineerName = req.body.engineerName;

    req.checkBody('complaintID', 'Contact field is required').notEmpty();
    req.checkBody('engineerName', 'Description field is required').notEmpty();

    let errors = req.validationErrors();

    if (errors) {
        res.render('admin/admin', {
            errors: errors
        });
    } else {
        const newComplaintMapping = new ComplaintMapping({
            complaintID: complaintID,

            engineerName: engineerName,

        });


        ComplaintMapping.registerMapping(newComplaintMapping, (err, complaint)
=> {

            if (err) throw err;

            req.flash('success_msg', 'You have successfully assigned a complaint to
Engineer');

            res.redirect('/admin');

        });

    }


});


// Junior Eng

router.get('/jeng', ensureAuthenticated, (req,res,next) => {

    res.render('junior/junior');

});


//Complaint

router.get('/complaint', ensureAuthenticated, (req, res, next) => {

    //console.log(req.session.passport.username);
    //console.log(user.name);

    res.render('complaint', {

        username: req.session.user,

    });

});


//Register a Complaint

router.post('/registerComplaint', (req, res, next) => {

    const name = req.body.name;

    const email = req.body.email;

    const contact = req.body.contact;

    const desc = req.body.desc;


    const postBody = req.body;

    console.log(postBody);


    req.checkBody('contact', 'Contact field is required').notEmpty();

    req.checkBody('desc', 'Description field is required').notEmpty();


    let errors = req.validationErrors();


    if (errors) {

        res.render('complaint', {
            errors: errors
        });
    } else {
        const newComplaint = new Complaint({
            name: name,
            email: email,
            contact: contact,
            desc: desc,
        });

        Complaint.registerComplaint(newComplaint, (err, complaint) => {
            if (err) throw err;
            req.flash('success_msg', 'You have successfully launched a complaint');
            res.redirect('/');
        });
    }
});



// Process Register
router.post('/register', (req, res, next) => {
    const name = req.body.name;
    const username = req.body.username;

    const email = req.body.email;

    const password = req.body.password;

    const password2 = req.body.password2;

    const role = req.body.role;



    req.checkBody('name', 'Name field is required').notEmpty();

    req.checkBody('email', 'Email field is required').notEmpty();

    req.checkBody('email', 'Email must be a valid email address').isEmail();

    req.checkBody('username', 'Username field is required').notEmpty();

    req.checkBody('password', 'Password field is required').notEmpty();

    req.checkBody('password2',                'Passwords                do                not
match').equals(req.body.password);

    req.checkBody('role', 'Role option is required').notEmpty();



    let errors = req.validationErrors();



    if (errors) {

        res.render('register', {

            errors: errors

        });

    } else {

        const newUser = new User({
          name: name,

          username: username,

          email: email,

          password: password,

          role: role

        });


        User.registerUser(newUser, (err, user) => {

          if (err) throw err;

          req.flash('success_msg', 'You are Successfully Registered and can Log in');

          res.redirect('/login');

        });

    }

});


// Local Strategy

passport.use(new LocalStrategy((username, password, done) => {

    User.getUserByUsername(username, (err, user) => {

        if (err) throw err;

        if (!user) {

          return done(null, false, {

            message: 'No user found'

          });
        }


        User.comparePassword(password, user.password, (err, isMatch) => {
            if (err) throw err;
            if (isMatch) {
                return done(null, user);
            } else {
                return done(null, false, {
                    message: 'Wrong Password'
                });
            }
        });
    });
}));


passport.serializeUser((user, done) => {
    var sessionUser = {
        _id: user._id,
        name: user.name,
        username: user.username,
        email: user.email,
        role: user.role,
    }
    done(null, sessionUser);
});


passport.deserializeUser((id, done) => {
    User.getUserById(id, (err, sessionUser) => {
        done(err, sessionUser);
    });
});


// Login Processing
router.post('/login', passport.authenticate('local',
    {
        failureRedirect: '/login',
        failureFlash: true


    }), (req, res, next) => {


        req.session.save((err) => {
        if (err) {
            return next(err);
        }
        if(req.user.role==='admin'){
            res.redirect('/admin');
        }
        else if(req.user.role==='jeng'){
            res.redirect('/jeng');
        }
        else{
            res.redirect('/');
        }
    });
});

// Access Control
function ensureAuthenticated(req, res, next) {
    if (req.isAuthenticated()) {
        return next();
    } else {
        req.flash('error_msg', 'You are not Authorized to view this page');
        res.redirect('/login');
    }
}

module.exports = router;
```

Complaint Management System is a web-based application that enables the users to lodge complaints and track their status. The system will be built using Node.js, which is a fast, scalable and efficient server-side JavaScript framework, and MongoDB, which is a popular NoSQL database. The system will be user-friendly and easy to use.

Objectives:

1. To provide a platform for users to lodge complaints and track their status
2. To enable the admin to manage the complaints and assign them to the respective departments
3. To provide real-time updates to the users on the status of their complaints
4. To maintain a centralized database of all the complaints lodged

Project Features:

1. User registration and login
2. User can lodge a complaint and attach relevant documents
3. User can view the status of their complaints
4. Admin login
5. Admin can view all the complaints lodged
6. Admin can assign the complaints to the respective departments
7. Admin can update the status of the complaints
8. Real-time updates to the users on the status of their complaints

Technology Stack:

1. Node.js
2. MongoDB
3. Express.js
4. HTML/CSS
5. JavaScript
6. Bootstrap

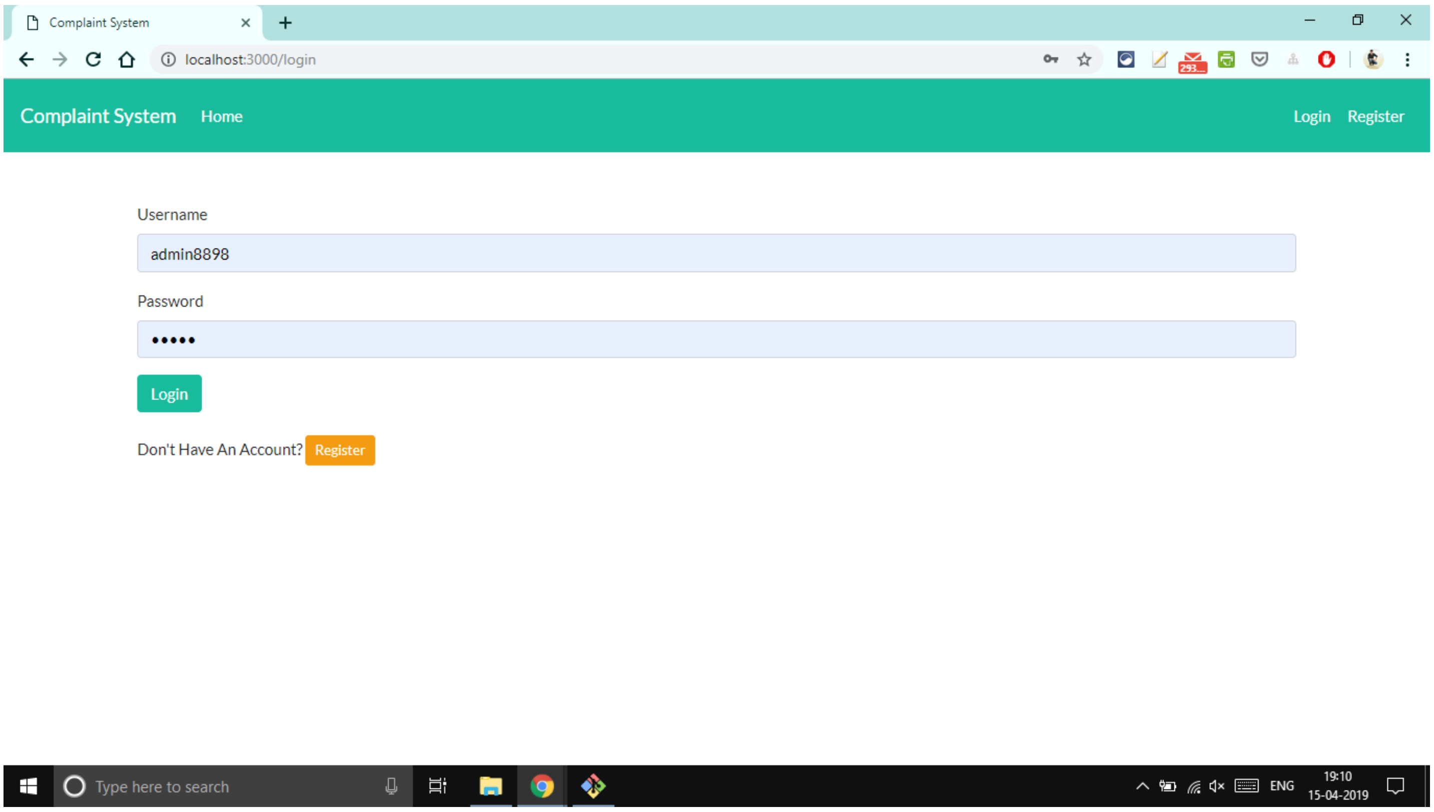
Methodology:

The project will follow the Agile methodology, which involves iterative and incremental development. The project will be divided into sprints, and each sprint will deliver a working software increment. The requirements will be gathered from the stakeholders, and the software will be developed in an iterative manner.

Conclusion:

The Complaint Management System using Node.js and MongoDB will provide an efficient and user-friendly platform for users to lodge complaints and track their status. The system will be built using the latest technologies, and it will follow the Agile methodology for development. The system will be scalable, fast and efficient, and it will help the admin to manage the complaints more effectively.

OUTPUTS:-



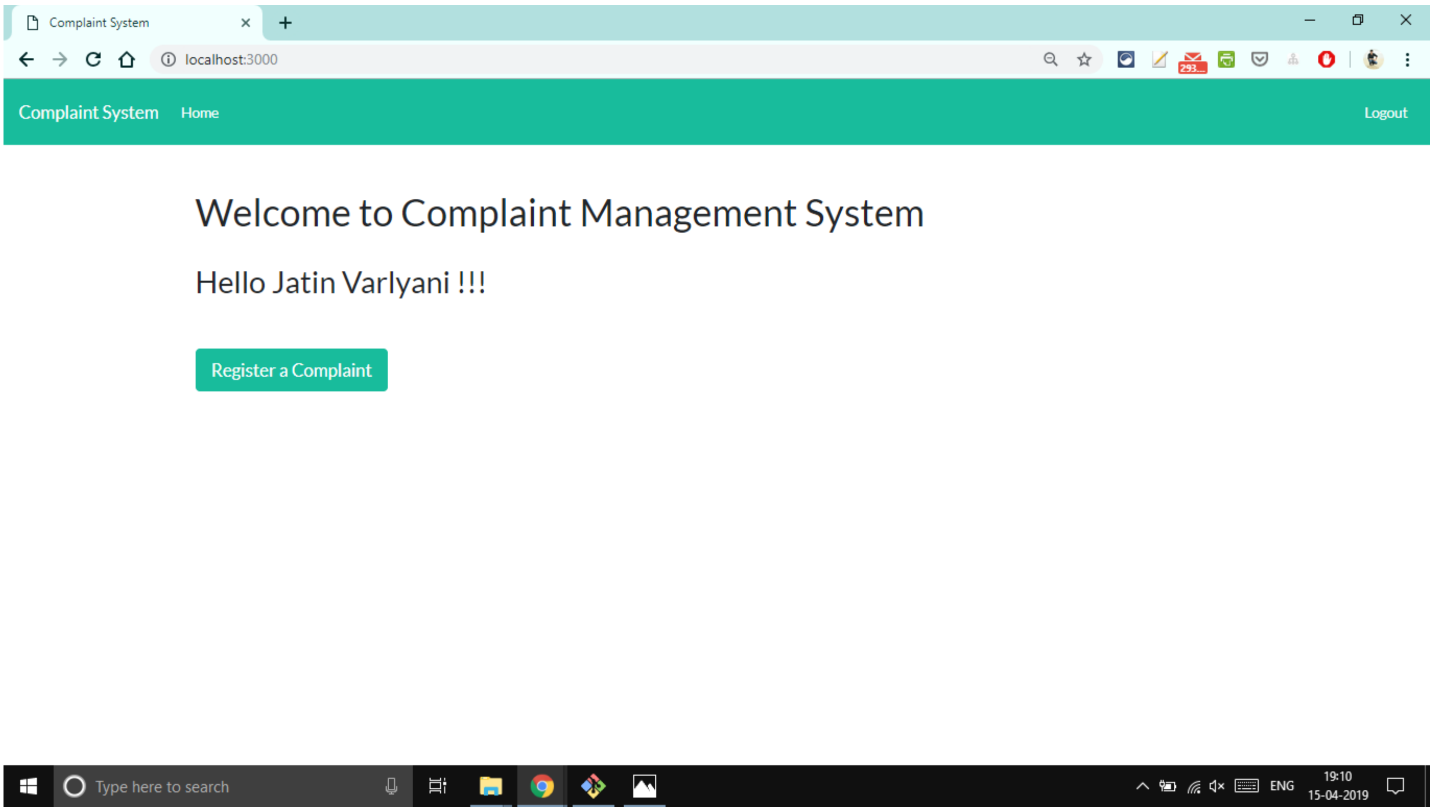
Username

admin8898

Password

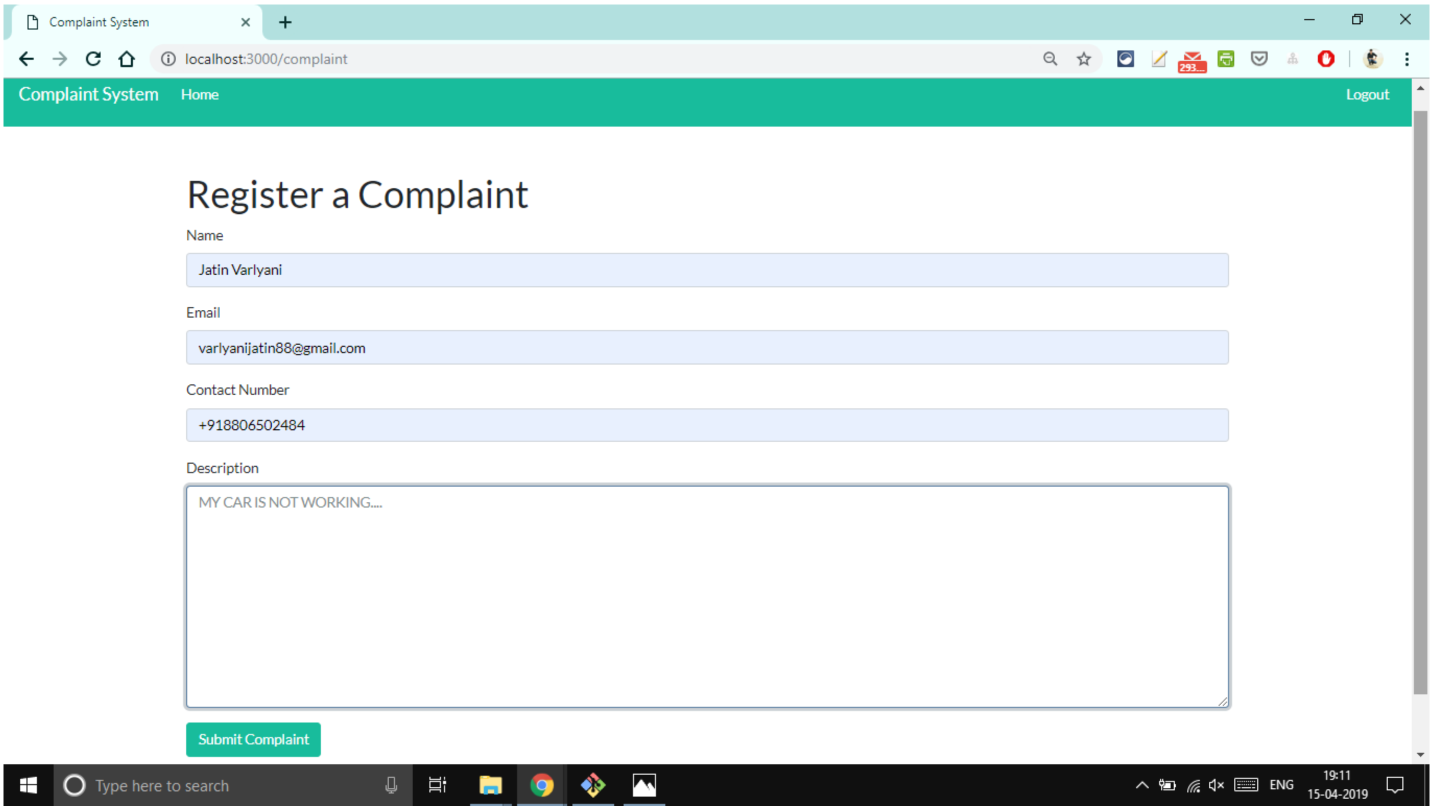
•••••

Don't Have An Account? Register



Welcome to Complaint Management System

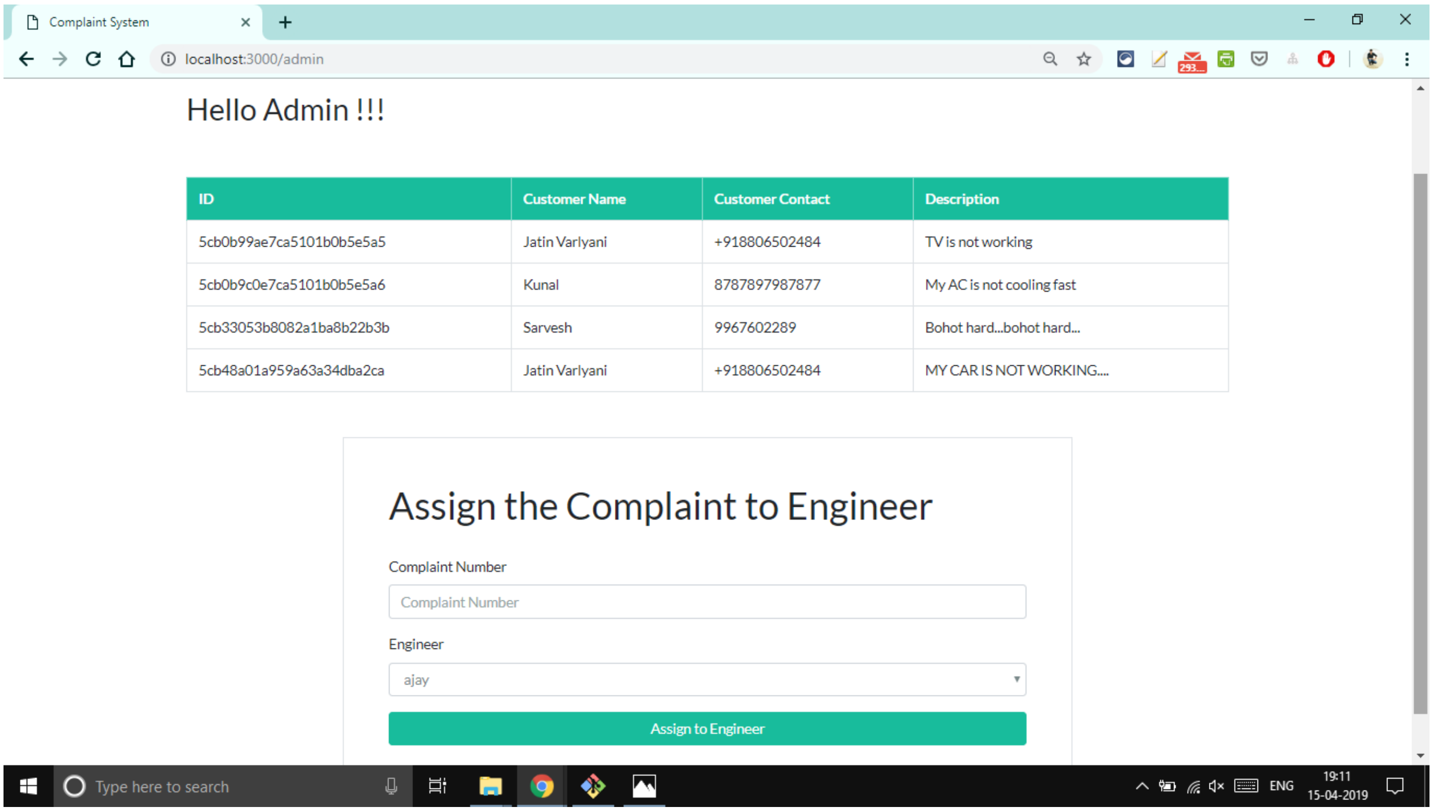
Hello Jatin Varlyani !!!



Register a Complaint

Name	Jatin Varlyani
Email	varlyanijatin88@gmail.com
Contact Number	+918806502484
Description	MY CAR IS NOT WORKING....

Submit Complaint



Hello Admin !!!

ID	Customer Name	Customer Contact	Description
5cb0b99ae7ca5101b0b5e5a5	Jatin Varlyani	+918806502484	TV is not working
5cb0b9c0e7ca5101b0b5e5a6	Kunal	8787897987877	My AC is not cooling fast
5cb33053b8082a1ba8b22b3b	Sarvesh	9967602289	Bohot hard...bohot hard...
5cb48a01a959a63a34dba2ca	Jatin Varlyani	+918806502484	MY CAR IS NOT WORKING....

Assign the Complaint to Engineer

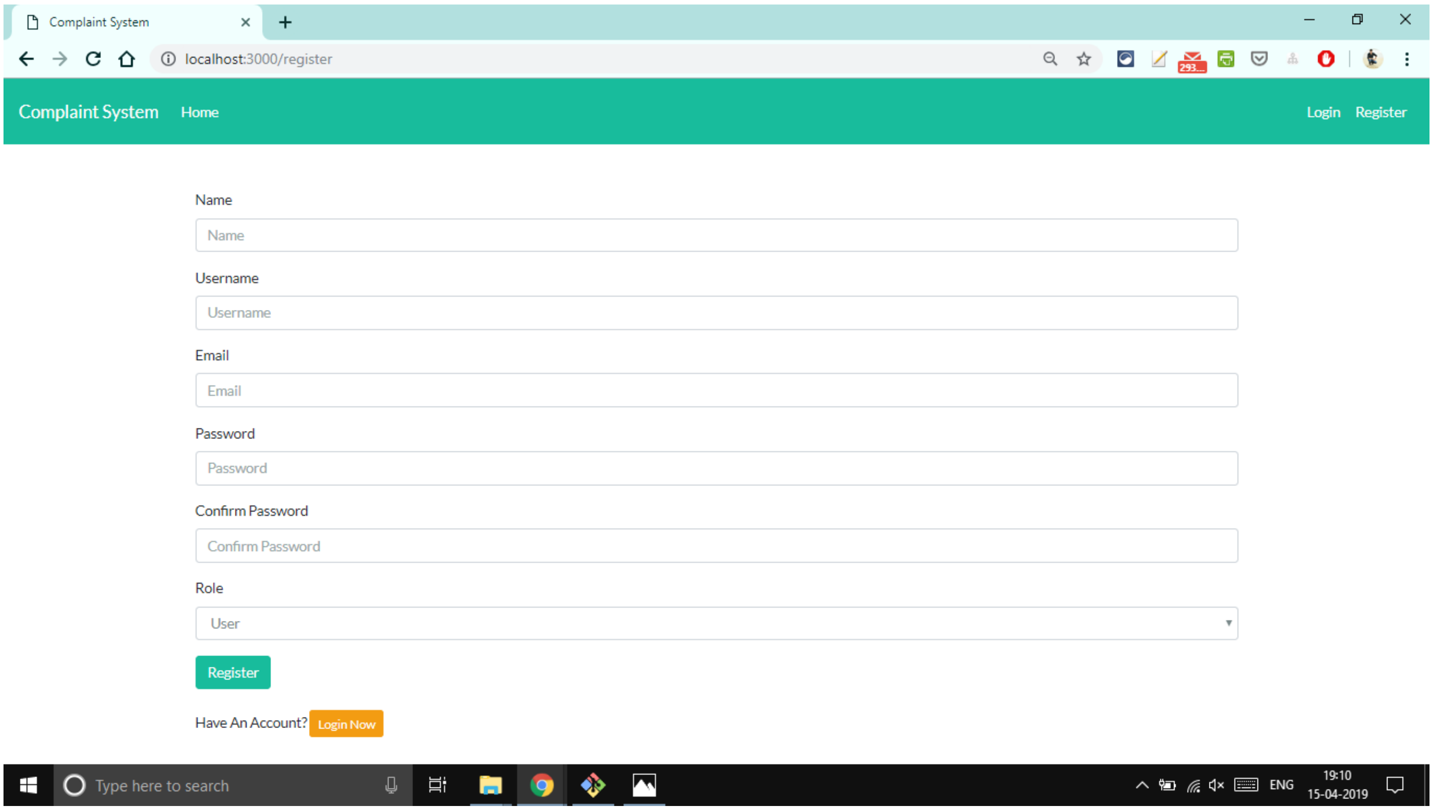
Complaint Number

Complaint Number

Engineer

ajay

Assign to Engineer



Complaint System

localhost:3000/register

Complaint System Home

Login Register

Name

Name

Username

Username

Email

Email

Password

Password

Confirm Password

Confirm Password

Role

User

Register

Have An Account? Login Now

Type here to search

ENG 19:10 15-04-2019