Project Report: Grievance Website Development

# Objective:

The primary goal of this project was to build a fully functional grievance management website where users can submit complaints or issues via a form. The website is backed by a Node.js server with data being stored in a database (MongoDB/MySQL).

# Tools & Technologies Used:

- Frontend: HTML, CSS, JavaScript  
- Backend: Node.js, Express.js, Multer (for handling file uploads)  
- Database: MongoDB (for storing grievance data), MySQL (for an alternative data storage solution)  
- Other Libraries/Tools: Mongoose (for MongoDB interactions), dotenv (for environment variable management)  
- Environment: Windows 10  
- Version Control: Git

# 1. Project Setup

Initially, the project was set up by organizing the directory structure. The key components included the front-end (index.html), back-end (server.js), and necessary configurations such as the .env file for environment variables.

## Key files:

index.html: Contained the grievance submission form.  
server.js: Node.js script that handled the backend operations like form submissions, database connections, and error handling.

## Challenges:

Initial Setup of Node.js: Configuring Node.js with Express, handling static files, and setting up routes for form submissions required setting up multiple dependencies.

# 2. Frontend Development

The grievance submission form was built in index.html using standard HTML and CSS for structure and styling. The form consisted of fields such as name, email, complaint type, and details, along with an optional file upload input.

## Frontend Logic:

JavaScript was used to handle form submission via an async call to the Node.js server using fetch().

## Challenges:

JavaScript Event Handling: Properly setting up submit event listeners to ensure that data is sent correctly to the backend.

# 3. Backend Development

The backend of the project was built using Node.js and Express.js.

## 3.1 Express.js Setup:

Express.js was used to manage routes and serve static files. The primary route was /submit-grievance, which handled form submissions from the front-end.

## Challenges:

Routing Configuration: Ensuring that routes were properly configured to accept both GET and POST requests. Handling asynchronous requests using async/await to manage the flow of operations.

## 3.2 Handling File Uploads:

To handle file uploads, the Multer library was implemented. This allowed users to attach files (such as documents or images) when submitting grievances.

## Challenges:

Multer Setup: Configuring Multer to handle file storage and ensuring that files were correctly processed and saved to the server.

## 3.3 MongoDB Setup:

MongoDB was chosen to store the grievance data. We used the Mongoose library to create a schema for grievance submissions and to manage interactions with the database.

## Challenges:

Database Connection Errors: We initially encountered connection errors due to an undefined uri. This was resolved by properly setting up the .env file and ensuring the MongoDB connection string was passed to Mongoose.

# 4. Database Setup and Configuration

## 4.1 MongoDB:

We used MongoDB to store grievance submissions, including all form fields and file paths.

## Challenges:

Environment Variables: The dotenv library was installed to manage environment variables such as the database connection string, but issues arose when the .env file wasn’t properly configured.  
Connection Error: Initially, the uri parameter in the Mongoose connection was undefined, which was traced back to the .env file not being loaded properly.

## 4.2 MySQL:

We also explored MySQL as a possible database for the project. However, issues arose due to MySQL not being properly installed or configured on the development machine.

## Challenges:

Service Setup Issues: We tried to start MySQL using both net start and sudo service, but encountered issues where the service was not recognized or started properly. This was due to MySQL not being installed as a service on the Windows machine.

# 5. Debugging and Problem Solving

## 5.1 Module Not Found Errors:

We encountered several instances of "module not found" errors (e.g., Multer, dotenv). These errors were resolved by ensuring the missing modules were installed via npm install.

## 5.2 Connection Refused Errors (MySQL):

While attempting to connect to MySQL, we faced ECONNREFUSED errors. These errors stemmed from the MySQL service not running. Ultimately, we decided to stick with MongoDB due to these setup challenges.

# 6. Deployment and Testing

After development, we tested the application by submitting grievances through the frontend. The backend successfully processed the data and stored it in MongoDB. The testing also included file uploads, which were correctly handled by the server.

## Challenges:

Cross-Origin Issues: During testing, we encountered issues with Cross-Origin Resource Sharing (CORS), which were resolved by configuring the cors middleware in Express.  
Error Handling: Several edge cases needed to be accounted for, such as missing form data or database connection issues.

# 7. Key Learnings and Takeaways

1. Effective Use of Node.js and Express.js: This project provided hands-on experience with setting up a Node.js server and managing backend operations effectively.  
2. Database Management: We learned how to integrate MongoDB into a Node.js project using Mongoose, and the importance of properly configuring environment variables to manage sensitive information like connection strings.  
3. Error Debugging: Throughout the project, multiple errors (e.g., module not found, connection refused) were encountered and resolved, enhancing our debugging skills.  
4. File Uploads with Multer: We gained experience working with file uploads and learned how to securely handle and store files on the server.  
5. Service Management in Windows: Attempting to configure MySQL as a service on a Windows machine provided insight into managing and troubleshooting Windows services.

# Conclusion:

The grievance website project allowed us to explore full-stack web development, from frontend design to backend implementation and database management. While we faced several challenges, including dependency errors, connection issues, and database service problems, each provided a valuable learning experience.  
Moving forward, the project could be enhanced by adding user authentication, creating a more robust admin panel for managing grievances, and deploying the website to a production environment.