

SMARTSHOP: YOUR GROCERY STORE EXPERIENCE

1. Introduction

- Project Title: Smartshop : your grocery store experience
- Team Members:
 - Siddhartha Poral – Full Stack Developer

2. Project Overview

- Purpose:

ResolveNow is designed to allow users to raise and manage complaints related to civic, institutional, or service-related issues. It streamlines the grievance redressal process with a dashboard interface for users and admins.
- Features:
 - User registration and secure login
 - Create, update, and delete complaints
 - Role-based access (User/Admin)
 - Admin dashboard for complaint resolution
 - Status tracking and timestamps
 - Email notifications on complaint updates - Mobile responsive UI

3. Architecture

- Frontend (React.js):
 - Developed with React
 - React Router DOM for navigation
 - Axios for API integration - Context API for state management
- Backend (Node.js + Express.js):
 - RESTful API architecture
 - Authentication middleware using JWT
 - Controllers for route handling
 - Error handling middleware
- Database (MongoDB):
 - MongoDB with Mongoose ODM

- Collections:
- Users: Name, email, hashed password, role
- Complaints: Title, description, status, timestamps, user reference

4. Setup Instructions

- Prerequisites:
 - Node.js (v18+)
 - MongoDB (local or Atlas)
 - Git
 - VS Code / Any IDE
- Installation Steps: `git clone https://github.com/yourusername/resolvenow.git`
`cd resolvenow`

```
# Setup backend
cd server  npm
install
```

```
# Setup frontend
cd ../client  npm
install
```

- Environment Variables (.env in /server):
`MONGO_URI=mongodb://127.0.0.1:27017/details`
`JWT_SECRET=your_jwt_secret`

5. Folder Structure

- Client (React):
 Frontend client/ |—
 public/ |— src/
 | |— components/
 | |— pages/
 | |— context/
 | |— App.js
 | |— index.js
- Server (Node + Express): Backend server/
 |— config/
 |— controllers/
 |— middleware/
 |— models/

```
|— routes/  
|— .env  
|— server.js
```

6. Running the Application • Frontend:

```
cd frontend
```

```
npm start
```

• Backend:

```
cd backend
```

```
npm start
```

7. API Documentation

Endpoint	Method	Description	Body Parameters
/api/auth/register	POST	Register new user	name, email, password
/api/auth/login	POST	Login and get token	email, password
/api/complaints	POST	Add new complaint	title, description
/api/complaints/:id	PUT	Update complaint (admin only)	status
/api/complaints	GET	Get all complaints (admin)	–

Example Response:

```
{  
  
  "success": true,  
  
  "message": "Complaint created",  
  
  "data": { ... }  
}
```

8. Authentication

Method: JWT (JSON Web Tokens)

Flow:

On login, token is issued and stored in localStorage

Protected routes validate token using middleware

Admin role check for sensitive routes

9. User Interface:

- Homepage
- Login/Register
- Complaint Dashboard
- Admin Panel

10. Testing

Manual testing using browser and API tools

Postman used for endpoint validation

(Optional) Jest and React Testing Library for component testing

11. Screenshots / Demo: is in implementation pdf.

12. Known Issues

- Admin role not switchable from frontend (manual DB update needed)
- Responsive layout issues on very small screens
- No complaint image/file upload yet

13. Future Enhancements

- Enable file/image upload for complaints
- Add search/filter/sort functionality
- Push/email notifications for status changes
- Dark mode support
- PWA version or mobile app