Sri Sivasubramaniya Nadar College of Engineering, Kalavakkam – 603110

(An Autonomous Institution, Affiliated to Anna University, Chennai)

Department of Computer Science & Engineering

UCS2611--- Internet Programming Lab

Consult Ease: Automated Consultancy Management System

Saisandeep Sangeetham 3122 22 5001 119

Srivathsan S 3122 22 5001 141

UCS2611-—Internet Programming Lab

Mini Project

AIM:

The aim of this project is to develop a user-friendly and efficient web application titled **"ConsultEase: Automated Consultancy Management System"** that facilitates the secure acquisition, management, and retrieval of consultancy project data. The system is designed to streamline the data entry process, provide advanced filtering and download options, and ensure timely communication through periodic email notifications — all while utilizing Excel sheets for data storage, without relying on traditional databases.

Requirements

These are the tools and technologies needed to develop and run the project:

Frontend

1. **React:** For building user interfaces.
2. **Vite:** A fast build tool and development server for modern web projects.
3. **Material-UI (MUI):** A React component library for building responsive and accessible UIs.
4. **Axios:** A promise-based HTTP client for making API requests.
5. **React Router:** For handling routing in the application.
6. **XLSX:** For exporting data to Excel files.
7. **File-Saver:** For saving files on the client-side.

Backend

1. **Express.js: F**or building the backend API.
2. **MongoDB:** A NoSQL database for storing user auth information.
3. **Mongoose:** An ODM library for MongoDB.
4. **JWT (jsonwebtoken):** For user authentication and authorization.
5. **Bcrypt.js:** For hashing passwords securely.
6. **Nodemailer:** For sending emails (e.g., OTPs) and notification alerts.
7. **Google APIs:** For integrating Google Drive and Google Sheets.
8. **Multer:** For handling file uploads.
9. **dotenv:** For managing environment variables.
10. **Morgan:** For logging HTTP requests.
11. **Nodemon:** For automatically restarting the backend server during development.

Functionalities in ConsultEase

1. User Authentication

Login:

Allows users to log in using their email and password.

Verifies credentials using JWT for secure authentication.

Register:

Enables new users to create an account by providing their details.

Passwords are securely hashed using bcrypt.js before storing them in the database.

2. Password Management

Forgot Password:

Users can reset their password by verifying their email with an OTP.

After OTP verification, users are redirected to a page where they can set a new password.

OTP Generation and Verification:

An OTP is sent to the user's email using Nodemailer.

The OTP is verified before allowing access to the password reset page.

3. Project Submission

Submit Consultancy Project:

Users can submit details of consultancy projects, including:

Industry Name

Project Title

Principal Investigator (PI)

Co-Principal Investigator (Co-PI)

Academic Year

Financial details (e.g., Amount Sanctioned, Amount Received)

Supporting documents (e.g., Bill Proof, Signed Agreement)

Files are uploaded to Google Drive using the Google Drive API.

Project details are stored in Google Sheets using the Google Sheets API.

4. Project Filtering and Viewing

View Projects:

Displays a list of all consultancy projects in a tabular format.

Filter Projects:

Users can filter projects based on:

Academic Year

Industry Name

Principal Investigator (PI)

Co-Principal Investigator (Co-PI)

Amount Range (Sanctioned Amount)

Export to Excel:

Users can export the filtered project data to an Excel file using the XLSX library.

5. Notifications

Email Notifications:

Sends OTPs to users for email verification during password reset.
Sends confirmation emails after successful project submission.

6. Admin Dashboard

View All Submissions:

Admins can view all project submissions in a centralized dashboard.

Filter and Search:

Admins can filter submissions based on various criteria (e.g., PI, Co-PI, Industry Name).

Download Data:

Admins can download project data as an Excel file for offline analysis.

7. Security

JWT Authentication:

Protects routes and ensures only authenticated users can access certain functionalities.

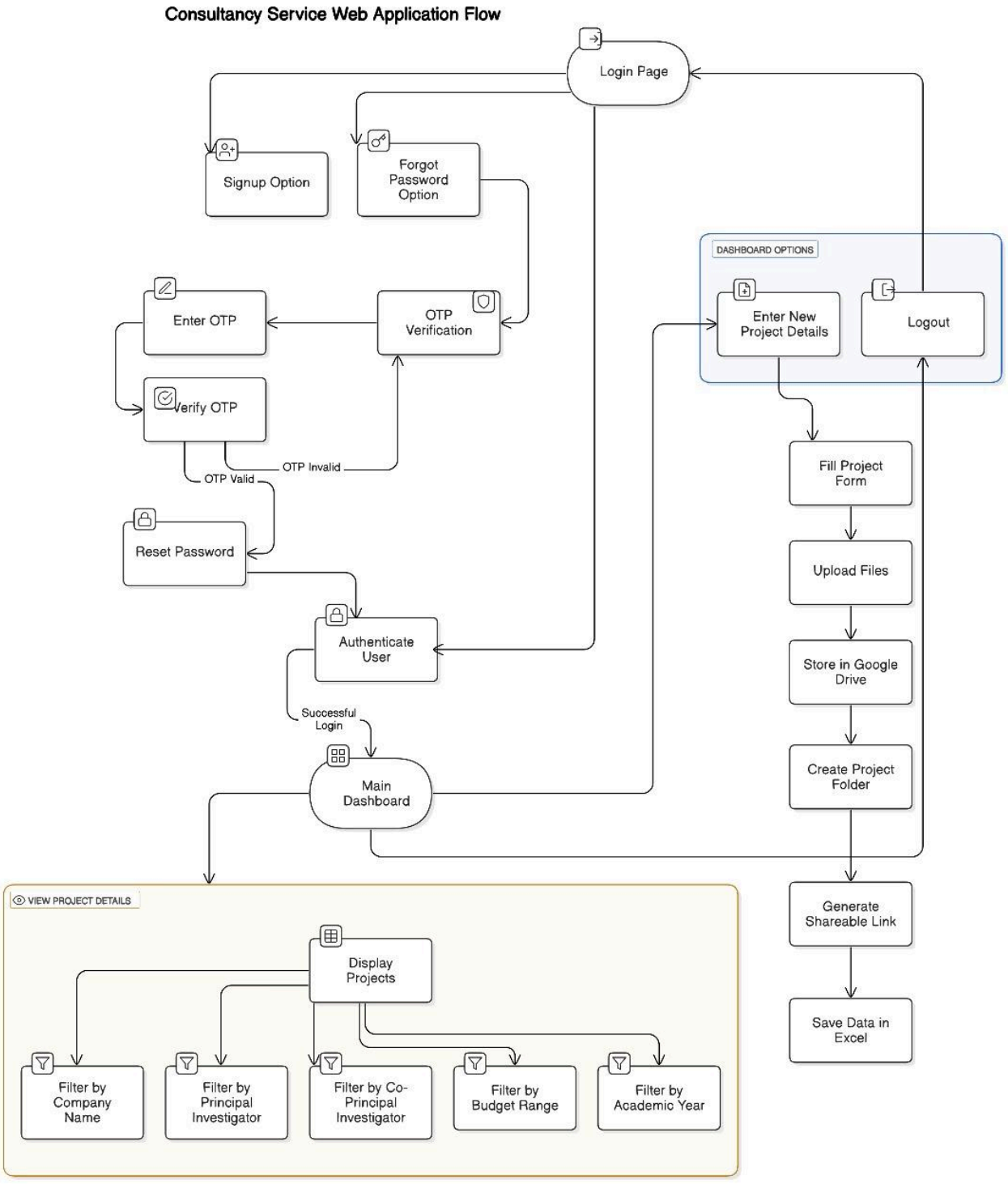
Password Hashing:

Uses bcrypt.js to hash passwords before storing them in the database.

Environment Variables:

Sensitive information (e.g., API keys, database credentials) is stored securely using doten

Flow Diagram :



CODE :

Frontend :

React Components :

Login.jsx:

```
import React, { useState, useEffect } from "react";
import "../CSS/login.css";
import { useNavigate } from "react-router-dom";
import { loginUser } from "../Helpers/api_communicator";

const Login = () => {
  const navigate = useNavigate();
  const [credentials, setCredentials] = useState({ email: "", psd: "" });
  const [error, setError] = useState("");
  const [loading, setLoading] = useState(false);

  // Check if user is already logged in
  useEffect(() => {
    const token = localStorage.getItem("token");
    if (token) {
      navigate("/form");
    }
  }, [navigate]);

  const gotoSignup = () => {
    navigate("/signup");
  };

  const gotoForgotPsd = ()=>{
    navigate("/verifyotp");
  };

  const handleLogin = async (event) => {
    event.preventDefault();
    setLoading(true);
    setError("");

    try {
      const result = await loginUser(credentials);

      if (result.success) {
```

```
      // Successfully logged in
      alert("Login Successfull");
      navigate("/form", { replace: true });
    } else {
      setError(result.error || "Login failed");
    }
  } catch (err) {
    setError("An unexpected error occurred");
    console.error(err);
  } finally {
    setLoading(false);
  }
};

return (
  <div className="login_container">
    <div className="brand"></div>
    <div className="login">
      <form onSubmit={handleLogin}>
        <h2>LOGIN</h2>
        <input
          type="email"
          placeholder="Enter your email"
          value={credentials.email}
          onChange={(e) =>
            setCredentials({ ...credentials, email: e.target.value })
          }
          required
        />
        <input
          type="password"
          placeholder="Enter your password"
          value={credentials.psd}
          onChange={(e) =>
            setCredentials({ ...credentials, psd: e.target.value })
          }
          required
        />
        {error && (
          <div
            style={{
              color: "rgb(255, 237, 39)",
              fontSize: "15px",
              height: "10px",
              textAlign: "left",
              marginBottom: "8px",
            }}>
```

```
              {error}
            </div>
          )}
          <input
            type="submit"
            value={loading ? "Logging in..." : "Login"}
            disabled={loading}
          />
          <a onClick={gotoSignup} style={{ cursor: "pointer" }}>
            Don't have an account?
          </a>

          <a onClick={gotoForgotPsd} style={{ cursor: "pointer" }}>
            Forgot Password
          </a>
        </form>
      </div>
    </div>
  );
};

export default Login;
```

Signup.jsx:

```
import React, { useState } from "react";
import "../CSS/login.css";
import { useNavigate } from "react-router-dom";
import { registerUser } from "../Helpers/api_communicator";

const Signup = () => {
  const navigate = useNavigate();
  const [credentials, setCredentials] = useState({
    name: "",
    email: "",
    psd: "",
    con_psd: "",
  });
  const [error, setError] = useState("");
  const [loading, setLoading] = useState(false);

  React.useEffect(() => {
    const token = localStorage.getItem("token");
```

```
  if (token) {
    navigate("/form");
  }
}, [navigate]);

const gotoLogin = () => {
  navigate("/");
};

const handleSignup = async (event) => {
  event.preventDefault();
  setLoading(true);
  setError("");

  if (!credentials.name.trim()) {
    setError("Name is required");
    setLoading(false);
    return;
  }

  if (credentials.psd !== credentials.con_psd) {
    setError("Passwords do not match");
    setLoading(false);
    return;
  }

  try {
    const result = await registerUser(credentials);

    if (result.success) {
      navigate("/", {
        state: { message: "Registration successful! Please login." },
      });
    } else {
      setError(result.error || "Registration failed");
    }
  } catch (error) {
    setError("An unexpected error occurred");
    console.error(error);
  } finally {
    setLoading(false);
  }
};

return (
  <div className="login_container">
    <div className="brand"></div>
```

```
    <div className="login">
      <form onSubmit={handleSignup}>
        <h2>Sign Up</h2>
        <input
          type="text"
          placeholder="Enter your name"
          value={credentials.name}
          onChange={(e) =>
            setCredentials({
              ...credentials,
              name: e.target.value,
            })
          }
          required
        />
        <input
          type="email"
          placeholder="Enter your email"
          value={credentials.email}
          onChange={(e) =>
            setCredentials({
              ...credentials,
              email: e.target.value,
            })
          }
          required
        />
        <input
          type="password"
          placeholder="Enter your password"
          value={credentials.psd}
          onChange={(e) =>
            setCredentials({
              ...credentials,
              psd: e.target.value,
            })
          }
          required
        />
        <input
          type="password"
          placeholder="Confirm your password"
          value={credentials.con_psd}
          onChange={(e) =>
            setCredentials({
              ...credentials,
              con_psd: e.target.value,
```

```
            })
          }
          required
        />
        {error && (
          <div
            style={{
              color: "rgb(255, 237, 39)",
              fontSize: "15px",
              height: "10px",
              textAlign: "left",
              marginBottom: "8px",
            }}>
            {error}
          </div>
        )}
        <input
          type="submit"
          value={loading ? "Signing up..." : "Sign Up"}
          disabled={loading}
        />
        <a onClick={gotoLogin} style={{ cursor: "pointer" }}>
          Already have an account?
        </a>
      </form>
    </div>
  </div>
 );
};

export default Signup;
```

ForgotPassword.jsx :

```
import React, { useState } from "react";
import "../CSS/login.css";
import { useNavigate } from "react-router-dom";
import { registerUser } from "../Helpers/api_communicator";

const Signup = () => {
  const navigate = useNavigate();
  const [credentials, setCredentials] = useState({
    name: "",
    email: "",
```

```
  psd: "",
  con_psd: "",
});
const [error, setError] = useState("");
const [loading, setLoading] = useState(false);

React.useEffect(() => {
  const token = localStorage.getItem("token");
  if (token) {
    navigate("/form");
  }
}, [navigate]);

const gotoLogin = () => {
  navigate("/");
};

const handleSignup = async (event) => {
  event.preventDefault();
  setLoading(true);
  setError("");

  if (!credentials.name.trim()) {
    setError("Name is required");
    setLoading(false);
    return;
  }

  if (credentials.psd !== credentials.con_psd) {
    setError("Passwords do not match");
    setLoading(false);
    return;
  }

  try {
    const result = await registerUser(credentials);

    if (result.success) {
      navigate("/", {
        state: { message: "Registration successful! Please login." },
      });
    } else {
      setError(result.error || "Registration failed");
    }
  } catch (error) {
    setError("An unexpected error occurred");
    console.error(error);
```

```
    } finally {
      setLoading(false);
    }
  };

  return (
    <div className="login_container">
      <div className="brand"></div>
      <div className="login">
        <form onSubmit={handleSignup}>
          <h2>Sign Up</h2>
          <input
            type="text"
            placeholder="Enter your name"
            value={credentials.name}
            onChange={(e) =>
              setCredentials({
                ...credentials,
                name: e.target.value,
              })
            }
            required
          />
          <input
            type="email"
            placeholder="Enter your email"
            value={credentials.email}
            onChange={(e) =>
              setCredentials({
                ...credentials,
                email: e.target.value,
              })
            }
            required
          />
          <input
            type="password"
            placeholder="Enter your password"
            value={credentials.psd}
            onChange={(e) =>
              setCredentials({
                ...credentials,
                psd: e.target.value,
              })
            }
            required
          />
```

```
          <input
            type="password"
            placeholder="Confirm your password"
            value={credentials.con_psd}
            onChange={(e) =>
              setCredentials({
                ...credentials,
                con_psd: e.target.value,
              })
            }
            required
          />
          {error && (
            <div
              style={{
                color: "rgb(255, 237, 39)",
                fontSize: "15px",
                height: "10px",
                textAlign: "left",
                marginBottom: "8px",
              }}>
              {error}
            </div>
          )}
          <input
            type="submit"
            value={loading ? "Signing up..." : "Sign Up"}
            disabled={loading}
          />
          <a onClick={gotoLogin} style={{ cursor: "pointer" }}>
            Already have an account?
          </a>
        </form>
      </div>
    </div>
  );
};

export default Signup;
```

Otp.jsx :

```jsx
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import {
  Box,
  Container,
  Typography,
  TextField,
  Button,
  Paper,
  Stack,
  InputAdornment,
  IconButton,
  Snackbar,
  Alert,
  CircularProgress,
  Divider,
} from "@mui/material";

import { generateOTP, verifyOtp } from "../Helpers/api_communicator";
import LockIcon from "@mui/icons-material/Lock";
import EmailIcon from "@mui/icons-material/Email";
import ArrowBackIcon from "@mui/icons-material/ArrowBack";
import SendIcon from "@mui/icons-material/Send";

const OTP = () => {
  const [email, setEmail] = useState("");
  const [otp, setOtp] = useState("");
  const [loading, setLoading] = useState(false);
  const [otpSent, setOtpSent] = useState(false);
  const [alert, setAlert] = useState({
    open: false,
    message: "",
    severity: "info",
  });

  const navigate = useNavigate();

  const handleSendOTP = async (e) => {
    e.preventDefault();

    if (!email || !/\S+@\S+\.\S+/.test(email)) {
      setAlert({
        open: true,
        message: "Please enter a valid email address",
        severity: "error",
      });
      return;
```

```
    }

    setLoading(true);
    try {
      const result = await generateOTP(email);

      if (result.success) {
        setAlert({
          open: true,
          message: "OTP sent successfully! Please check your email",
          severity: "success",
        });
        setOtpSent(true);
      } else {
        setAlert({
          open: true,
          message: result.error || "Failed to send OTP",
          severity: "error",
        });
      }
    } catch (error) {
      setAlert({
        open: true,
        message: "An error occurred while sending OTP",
        severity: "error",
      });
      console.log(error);
    } finally {
      setLoading(false);
    }
  };

  const handleVerifyOTP = async (e) => {
    e.preventDefault();

    if (!otp || otp.length < 6) {
      setAlert({
        open: true,
        message: "Please enter a valid 6-digit OTP",
        severity: "error",
      });
      return;
    }

    setLoading(true);
    try {
      const result = await verifyOtp({ email, otp });
```

```
    if (result.success) {
      setAlert({
        open: true,
        message: "OTP verified successfully!",
        severity: "success",
      });

      sessionStorage.setItem("verified_email", email);

      setTimeout(() => {
        navigate("/forgotPsd");
      }, 1500);
    } else {
      setAlert({
        open: true,
        message: result.error || "OTP verification failed",
        severity: "error",
      });
    }
  } catch (error) {
    setAlert({
      open: true,
      message: "OTP verification failed. Please try again.",
      severity: "error",
    });
    console.log(error);
  } finally {
    setLoading(false);
  }
};

const handleResendOTP = async () => {
  if (!email) {
    setAlert({
      open: true,
      message: "Email address is required",
      severity: "error",
    });
    return;
  }

  setLoading(true);
  try {
    const result = await generateOTP(email);

    if (result.success) {
```

```
      setAlert({
        open: true,
        message: "OTP resent successfully! Please check your email",
        severity: "success",
      });
    } else {
      setAlert({
        open: true,
        message: result.error || "Failed to resend OTP",
        severity: "error",
      });
    }
  } catch (error) {
    setAlert({
      open: true,
      message: "An error occurred while resending OTP",
      severity: "error",
    });
    console.log(error);
  } finally {
    setLoading(false);
  }
};

const handleCloseAlert = () => {
  setAlert({ ...alert, open: false });
};

const goBack = () => {
  navigate("/");
};

return (
  <Container maxWidth="sm" sx={{ mt: 8 }}>
    <IconButton
      sx={{ mb: 2 }}
      onClick={goBack}
      color="primary"
    >
      <ArrowBackIcon /> Back
    </IconButton>

    <Paper elevation={3} sx={{ p: 4, borderRadius: 2 }}>
      <Stack spacing={3} alignItems="center">
        <Box sx={{ textAlign: "center" }}>
          <Box
            sx={{
```

```
                display: "inline-flex",
                bgcolor: "primary.light",
                p: 2,
                borderRadius: "50%",
                mb: 2
              }}
            >
              <LockIcon fontSize="large" color="primary" />
            </Box>
            <Typography variant="h5" component="h1" gutterBottom
fontWeight="medium">
              Forgot Password
            </Typography>
            <Typography variant="body1" color="text.secondary" sx={{ mb: 2 }}>
              {otpSent
                ? `Enter the verification code sent to ${email}`
                : "Enter your email to receive a verification code"}
            </Typography>
          </Box>

          {!otpSent ? (
            <Box component="form" noValidate sx={{ width: "100%" }}
onSubmit={handleSendOTP}>
              <TextField
                fullWidth
                label="Email Address"
                type="email"
                value={email}
                onChange={(e) => setEmail(e.target.value)}
                variant="outlined"
                required
                placeholder="Enter your email address"
                sx={{ mb: 3 }}
                InputProps={{
                  startAdornment: (
                    <InputAdornment position="start">
                      <EmailIcon color="action" />
                    </InputAdornment>
                  ),
                }}
              />

              <Button
                type="submit"
                variant="contained"
                fullWidth
                size="large"
```

```
                disabled={loading}
                sx={{ py: 1.5 }}
                startIcon={!loading && <SendIcon />}
              >
                {loading ? (
                  <CircularProgress size={24} color="inherit" />
                ) : (
                  "Send OTP"
                )}
              </Button>
            </Box>
          ) : (
            <Box component="form" noValidate sx={{ width: "100%" }}
onSubmit={handleVerifyOTP}>
              <TextField
                fullWidth
                label="Enter OTP"
                value={otp}
                onChange={(e) => setOtp(e.target.value)}
                variant="outlined"
                required
                inputProps={{
                  maxLength: 6,
                  inputMode: 'numeric',
                  pattern: '[0-9]*'
                }}
                placeholder="Enter the 6-digit code"
                sx={{ mb: 3 }}
                InputProps={{
                  startAdornment: (
                    <InputAdornment position="start">
                      <LockIcon color="action" />
                    </InputAdornment>
                  ),
                }}
              />

              <Button
                type="submit"
                variant="contained"
                fullWidth
                size="large"
                disabled={loading}
                sx={{ py: 1.5 }}
              >
                {loading ? (
                  <CircularProgress size={24} color="inherit" />
```

```
                ) : (
                  "Verify OTP"
                )}
              </Button>

              <Typography variant="body2" color="text.secondary"
textAlign="center" sx={{ mt: 2 }}>
                Didn't receive the code?
                <Button
                  color="primary"
                  sx={{ ml: 1 }}
                  onClick={handleResendOTP}
                  disabled={loading}
                >
                  Resend OTP
                </Button>
              </Typography>
            </Box>
          )}
        </Stack>
      </Paper>

      <Snackbar
        open={alert.open}
        autoHideDuration={6000}
        onClose={handleCloseAlert}
        anchorOrigin={{ vertical: "bottom", horizontal: "center" }}
      >
        <Alert
          onClose={handleCloseAlert}
          severity={alert.severity}
          variant="filled"
          sx={{ width: "100%" }}
        >
          {alert.message}
        </Alert>
      </Snackbar>
    </Container>
  );
};

export default OTP;
```

Form.jsx :

```
import React, { useState } from "react";
import NavBar from "../components/NavBar";
import { formsubmission } from "../Helpers/api_communicator";
import {
  Container,
  Typography,
  TextField,
  Button,
  FormControl,
  InputLabel,
  Select,
  MenuItem,
  Box,
  Paper,
  Stack,
  FormHelperText,
  InputAdornment,
  Snackbar,
  Alert,
  CircularProgress,
} from "@mui/material";

const Form = () => {
  const [formData, setFormData] = useState({
    industryName: "",
    projectDuration: "",
    projectTitle: "",
    pi: "",
    coPI: "",
    academicYear: "",
    amountSanctioned: "",
    amountReceived: "",
    billDetails: "",
    billProof: null,
    agreement: null,
    studentDetails: "",
    projectSummary: "",
  });

  const [loading, setLoading] = useState(false);
  const [alert, setAlert] = useState({
    open: false,
```

```
    message: "",
    severity: "success",
  });

  const handleChange = (e) => {
    const { name, value, files } = e.target;
    if (files) {
      setFormData((prev) => ({
        ...prev,
        [name]: files[0],
      }));
    } else {
      setFormData((prev) => ({
        ...prev,
        [name]: value,
      }));
    }
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    const { billProof, agreement } = formData;

    if (!billProof || !agreement) {
      setAlert({
        open: true,
        message: "Please upload all the required documents",
        severity: "error",
      });
      return;
    }

    const data = new FormData();
    for (let key in formData) {
      data.append(key, formData[key]);
    }

    setLoading(true);
    try {
      const response = await formsubmission(data);
      if (response.status === 200) {
        setAlert({
          open: true,
          message: "Form submitted successfully",
          severity: "success",
        });
        setFormData({
```

```
        industryName: "",
        projectDuration: "",
        projectTitle: "",
        pi: "",
        coPI: "",
        academicYear: "",
        amountSanctioned: "",
        amountReceived: "",
        billDetails: "",
        billProof: null,
        agreement: null,
        studentDetails: "",
        projectSummary: "",
      });
    }
  } catch (error) {
    setAlert({
      open: true,
      message:
        error.response?.data?.message ||
        "There was an error submitting the form.",
      severity: "error",
    });
  } finally {
    setLoading(false);
  }
};

const handleCloseAlert = () => {
  setAlert({ ...alert, open: false });
};

return (
  <div>
    <NavBar />
    <Container maxWidth="md" sx={{ mt: 4, mb: 8 }}>
      <Paper sx={{ p: 3, borderRadius: 2 }}>
        <Box sx={{ mb: 4, textAlign: "start" }}>
          <Typography
            variant="h5"
            component="h2"
            color="black"
            fontWeight="medium">
            Consultancy Project Details
          </Typography>
        </Box>
```

```
        <Box component="form" noValidate>
          <Stack spacing={3}>
            <TextField
              fullWidth
              label="Industry Name"
              name="industryName"
              value={formData.industryName}
              onChange={handleChange}
              required
              variant="outlined"
            />

            <TextField
              fullWidth
              label="Project Duration (in months)"
              name="projectDuration"
              type="number"
              value={formData.projectDuration}
              onChange={handleChange}
              required
              variant="outlined"
              InputProps={{
                endAdornment: (
                  <InputAdornment position="end">months</InputAdornment>
                ),
              }}
            />

            <TextField
              fullWidth
              label="Project Title"
              name="projectTitle"
              value={formData.projectTitle}
              onChange={handleChange}
              required
              variant="outlined"
            />

            <TextField
              fullWidth
              label="Principal Investigator"
              name="pi"
              value={formData.pi}
              onChange={handleChange}
              required
              variant="outlined"
            />
```

```
          <TextField
            fullWidth
            label="Co-Principal Investigator"
            name="coPI"
            value={formData.coPI}
            onChange={handleChange}
            variant="outlined"
          />

          <FormControl fullWidth required>
            <InputLabel id="academic-year-label">Academic Year</InputLabel>
            <Select
              labelId="academic-year-label"
              id="academicYear"
              name="academicYear"
              value={formData.academicYear}
              onChange={handleChange}
              label="Academic Year">
              <MenuItem value="">
                <em>--Select Academic Year--</em>
              </MenuItem>
              <MenuItem value="1st Year">1st Year</MenuItem>
              <MenuItem value="2nd Year">2nd Year</MenuItem>
              <MenuItem value="3rd Year">3rd Year</MenuItem>
              <MenuItem value="4th Year">4th Year</MenuItem>
            </Select>
          </FormControl>

          <TextField
            fullWidth
            label="Amount Sanctioned (₹)"
            name="amountSanctioned"
            type="number"
            value={formData.amountSanctioned}
            onChange={handleChange}
            required
            variant="outlined"
            InputProps={{
              startAdornment: (
                <InputAdornment position="start">₹</InputAdornment>
              ),
            }}
          />

          <TextField
            fullWidth
```

```
    label="Amount Received (₹)"
    name="amountReceived"
    type="number"
    value={formData.amountReceived}
    onChange={handleChange}
    required
    variant="outlined"
    InputProps={{
      startAdornment: (
        <InputAdornment position="start">₹</InputAdornment>
      ),
    }}
  />

  <TextField
    fullWidth
    label="Bill Settlement Details"
    name="billDetails"
    multiline
    rows={3}
    value={formData.billDetails}
    onChange={handleChange}
    required
    variant="outlined"
  />

  <Box>
    <Typography variant="subtitle1" gutterBottom>
      Bill Proof Upload:
    </Typography>
    <input
      type="file"
      name="billProof"
      onChange={handleChange}
      accept=".pdf,.jpg,.jpeg,.png"
      required
      style={{ width: "100%", padding: "10px 0" }}
    />
    {formData.billProof && (
      <FormHelperText>
        Selected file: {formData.billProof.name}
      </FormHelperText>
    )}
  </Box>

  <Box>
    <Typography variant="subtitle1" gutterBottom>
```

```
            Signed Agreement Upload:
          </Typography>
          <input
            type="file"
            name="agreement"
            onChange={handleChange}
            accept=".pdf"
            required
            style={{ width: "100%", padding: "10px 0" }}
          />
          {formData.agreement && (
            <FormHelperText>
              Selected file: {formData.agreement.name}
            </FormHelperText>
          )}
        </Box>

        <TextField
          fullWidth
          label="Student Details"
          name="studentDetails"
          multiline
          rows={3}
          placeholder="Enter student names and roles"
          value={formData.studentDetails}
          onChange={handleChange}
          required
          variant="outlined"
        />

        <Box>
          <TextField
            fullWidth
            label="Project Summary (100 words)"
            name="projectSummary"
            multiline
            rows={4}
            value={formData.projectSummary}
            onChange={handleChange}
            inputProps={{ maxLength: 500 }}
            required
            variant="outlined"
          />
          <Typography
            variant="caption"
            sx={{ display: "block", mt: 1, textAlign: "right" }}>
            {formData.projectSummary.length}/500 characters
```

```
              </Typography>
            </Box>

            <Box sx={{ mt: 3, textAlign: "center" }}>
              <Button
                type="submit"
                variant="contained"
                color="primary"
                size="large"
                onClick={handleSubmit}
                disabled={loading}
                sx={{ px: 4, py: 1, fontSize: "1rem" }}>
                {loading ? (
                  <>
                    <CircularProgress
                      size={24}
                      color="inherit"
                      sx={{ mr: 1 }}
                    />{" "}
                    Submitting...
                  </>
                ) : (
                  "Submit"
                )}
              </Button>
            </Box>
          </Stack>
        </Box>
      </Paper>
    </Container>

    <Snackbar
      open={alert.open}
      autoHideDuration={6000}
      onClose={handleCloseAlert}
      anchorOrigin={{ vertical: "bottom", horizontal: "center" }}>
      <Alert
        onClose={handleCloseAlert}
        severity={alert.severity}
        sx={{ width: "100%" }}>
        {alert.message}
      </Alert>
    </Snackbar>
  </div>
 );
};
```

```
export default Form;
```

ViewData.jsx :

```
import React, { useState, useEffect } from "react";
import {
  Container,
  TextField,
  Select,
  MenuItem,
  InputLabel,
  FormControl,
  Button,
  Table,
  TableBody,
  TableCell,
  TableContainer,
  TableHead,
  TableRow,
  Paper,
  Box,
  Grid,
  Typography,
  CircularProgress,
  Divider,
  InputAdornment,
} from "@mui/material";
import * as XLSX from "xlsx";
import { saveAs } from "file-saver";
import NavBar from "../components/NavBar";
import axios from "axios";
import SearchIcon from "@mui/icons-material/Search";
import ClearAllIcon from "@mui/icons-material/ClearAll";

const ViewData = () => {
  const [data, setData] = useState([]);
  const [loading, setLoading] = useState(false);
  const [error, setError] = useState("");
  const [filters, setFilters] = useState({
    academicYear: "",
    minAmount: "",
    maxAmount: "",
    pi: "",
```

```
    copi: "",
    industryName: "",
  });

  useEffect(() => {
    fetchData();
  }, []);

  const fetchData = async () => {
    setLoading(true);
    setError("");
    try {
      const response = await axios.get("http://localhost:8000/api/getData", {
        headers: {
          Authorization: `Bearer ${localStorage.getItem("token")}`,
        },
      });
      setData(response.data.data || []);
    } catch (error) {
      setError("Failed to fetch data. Please try again.");
      console.error("Error fetching data:", error);
    } finally {
      setLoading(false);
    }
  };

  const handleChange = (e) => {
    setFilters({ ...filters, [e.target.name]: e.target.value });
  };

  const applyFilters = async () => {
    setLoading(true);
    setError("");
    try {
      let endpoint = "http://localhost:8000/api/getData";

      if (filters.pi) {
        endpoint = `http://localhost:8000/api/getDataByPI/${filters.pi}`;
      }
      else if (filters.copi) {
        endpoint = `http://localhost:8000/api/getDataBycoPI/${filters.copi}`;
      }
      else if (filters.minAmount && filters.maxAmount) {
        endpoint =
`http://localhost:8000/api/getDataBySanctionAmt/${filters.minAmount}/${filters.m
axAmount}`;
      }
```

```
      else if (filters.industryName){
        endpoint =
`http://localhost:8000/api/getDataByIndustryName/${filters.industryName}`;
      }

      const response = await axios.get(endpoint, {
        headers: {
          Authorization: `Bearer ${localStorage.getItem("token")}`,
        },
      });

      let filteredData = response.data.data || [];

      if (filters.academicYear && filteredData.length > 0) {
        filteredData = filteredData.filter(item =>
          item.academicYear === filters.academicYear
        );
      }

      setData(filteredData);
    } catch (error) {
      setError("Failed to apply filters. Please try again.");
      console.error("Error applying filters:", error);
    } finally {
      setLoading(false);
    }
  };

  const resetFilters = () => {
    setFilters({
      academicYear: "",
      minAmount: "",
      maxAmount: "",
      pi: "",
      copi: "",
      industryName: "",
    });
    fetchData();
  };

  const downloadExcel = () => {
    const worksheetData = data.map(item => ({
      "Industry Name": item.industryName,
      "Project Title": item.projectTitle,
      "Duration (months)": item.projectDuration,
      "Principal Investigator": item.pi,
      "Co-Principal Investigator": item.coPI,
```

```
      "Academic Year": item.academicYear,
      "Amount Sanctioned": item.amountSanctioned,
      "Amount Received": item.amountReceived,
      "Bill Details": item.billDetails,
      "Student Details": item.studentDetails,
      "Project Summary": item.projectSummary,
    }));

    const worksheet = XLSX.utils.json_to_sheet(worksheetData);
    const workbook = XLSX.utils.book_new();
    XLSX.utils.book_append_sheet(workbook, worksheet, "ConsultancyProjects");

    const excelBuffer = XLSX.write(workbook, { bookType: "xlsx", type: "array"
});
    const file = new Blob([excelBuffer], { type: "application/octet-stream" });
    saveAs(file, "ConsultancyProjects.xlsx");
  };

  return (
    <div>
      <NavBar />
      <Container maxWidth="lg" sx={{ mt: 4, mb: 4 }}>
        <Typography variant="h4" gutterBottom>
          View Consultancy Projects
        </Typography>

        <Paper sx={{ p: 3, mb: 4, bgcolor: '#f9f9f9', borderRadius: 2 }}>
          <Box sx={{ mb: 2, display: 'flex', alignItems: 'center',
justifyContent: 'space-between' }}>
            <Typography variant="h6" fontWeight="medium" color="primary">
              Filter Projects
            </Typography>
            <Box>
              <Button
                variant="contained"
                color="primary"
                onClick={applyFilters}
                disabled={loading}
                startIcon={<SearchIcon />}
                sx={{ mr: 1 }}
              >
                Apply Filters
              </Button>
              <Button
                variant="outlined"
                onClick={resetFilters}
                disabled={loading}
```

```
                startIcon={<ClearAllIcon />}
              >
                Reset
              </Button>
            </Box>
          </Box>

          <Divider sx={{ mb: 3 }} />

          <Grid container spacing={2}>
            {/* Academic Year Filter */}
            <Grid item xs={12} md={4} lg={3}>
              <Paper elevation={0} sx={{ p: 2, bgcolor: 'white', height: '100%',
borderRadius: 1 }}>
                <Typography variant="subtitle2" color="text.secondary"
gutterBottom>
                  Academic Year
                </Typography>
                <FormControl fullWidth size="small">
                  <Select
                    name="academicYear"
                    value={filters.academicYear}
                    onChange={handleChange}
                    displayEmpty
                  >
                    <MenuItem value="">All Years</MenuItem>
                    <MenuItem value="1st Year">1st Year</MenuItem>
                    <MenuItem value="2nd Year">2nd Year</MenuItem>
                    <MenuItem value="3rd Year">3rd Year</MenuItem>
                    <MenuItem value="4th Year">4th Year</MenuItem>
                  </Select>
                </FormControl>
              </Paper>
            </Grid>

            {/* Industry Name Filter */}
            <Grid item xs={12} md={4} lg={3}>
              <Paper elevation={0} sx={{ p: 2, bgcolor: 'white', height: '100%',
borderRadius: 1 }}>
                <Typography variant="subtitle2" color="text.secondary"
gutterBottom>
                  Industry Name
                </Typography>
                <TextField
                  name="industryName"
                  placeholder="Search by industry"
                  fullWidth
```

```
                size="small"
                value={filters.industryName}
                onChange={handleChange}
              />
            </Paper>
          </Grid>

          {/* Principal Investigator Filter */}
          <Grid item xs={12} md={4} lg={3}>
            <Paper elevation={0} sx={{ p: 2, bgcolor: 'white', height: '100%',
borderRadius: 1 }}>
              <Typography variant="subtitle2" color="text.secondary"
gutterBottom>
                Principal Investigator
              </Typography>
              <TextField
                name="pi"
                placeholder="Search by PI name"
                fullWidth
                size="small"
                value={filters.pi}
                onChange={handleChange}
              />
            </Paper>
          </Grid>

          {/* Co-Principal Investigator Filter */}
          <Grid item xs={12} md={4} lg={3}>
            <Paper elevation={0} sx={{ p: 2, bgcolor: 'white', height: '100%',
borderRadius: 1 }}>
              <Typography variant="subtitle2" color="text.secondary"
gutterBottom>
                Co-Principal Investigator
              </Typography>
              <TextField
                name="copi"
                placeholder="Search by Co-PI name"
                fullWidth
                size="small"
                value={filters.copi}
                onChange={handleChange}
              />
            </Paper>
          </Grid>

          {/* Amount Range Filter */}
          <Grid item xs={12} md={8} lg={6}>
```

```
              <Paper elevation={0} sx={{ p: 2, bgcolor: 'white', borderRadius: 1
}}>
                <Typography variant="subtitle2" color="text.secondary"
gutterBottom>
                  Amount Range (₹)
                </Typography>
                <Box sx={{ display: 'flex', gap: 2 }}>
                  <TextField
                    name="minAmount"
                    placeholder="Minimum"
                    type="number"
                    fullWidth
                    size="small"
                    value={filters.minAmount}
                    onChange={handleChange}
                    InputProps={{
                      startAdornment: <InputAdornment
position="start">₹</InputAdornment>,
                    }}
                  />
                  <Box sx={{ display: 'flex', alignItems: 'center', px: 1
}}>to</Box>
                  <TextField
                    name="maxAmount"
                    placeholder="Maximum"
                    type="number"
                    fullWidth
                    size="small"
                    value={filters.maxAmount}
                    onChange={handleChange}
                    InputProps={{
                      startAdornment: <InputAdornment
position="start">₹</InputAdornment>,
                    }}
                  />
                </Box>
              </Paper>
            </Grid>
          </Grid>
        </Paper>

        {error && (
          <Box sx={{ mb: 2 }}>
            <Typography color="error">{error}</Typography>
          </Box>
        )}
```

```
        <Box sx={{ display: "flex", justifyContent: "space-between", mb: 2,
alignItems: "center" }}>
          <Typography variant="h6">
            {data.length} Projects Found
          </Typography>
          <Button
            variant="contained"
            color="success"
            onClick={downloadExcel}
            disabled={loading || data.length === 0}
          >
            Download Excel
          </Button>
        </Box>

        {loading ? (
          <Box sx={{ display: "flex", justifyContent: "center", my: 4 }}>
            <CircularProgress />
          </Box>
        ) : (
          <TableContainer component={Paper}>
            <Table>
              <TableHead>
                <TableRow sx={{ backgroundColor: '#1976d2' }}>
                  <TableCell sx={{ color: 'white', fontWeight: 'bold'
}}>Industry Name</TableCell>
                  <TableCell sx={{ color: 'white', fontWeight: 'bold' }}>Project
Title</TableCell>
                  <TableCell sx={{ color: 'white', fontWeight: 'bold'
}}>Principal Investor</TableCell>
                  <TableCell sx={{ color: 'white', fontWeight: 'bold'
}}>Co-Principal Investor</TableCell>
                  <TableCell sx={{ color: 'white', fontWeight: 'bold'
}}>Academic Year</TableCell>
                  <TableCell sx={{ color: 'white', fontWeight: 'bold' }}>Amount
Sanctioned</TableCell>
                  <TableCell sx={{ color: 'white', fontWeight: 'bold' }}>Amount
Received</TableCell>
                </TableRow>
              </TableHead>
              <TableBody>
                {data.length > 0 ? (
                  data.map((row, index) => (
                    <TableRow key={index}>
                      <TableCell>{row.industryName}</TableCell>
                      <TableCell>{row.projectTitle}</TableCell>
                      <TableCell>{row.pi}</TableCell>
```

```
                    <TableCell>{row.coPI}</TableCell>
                    <TableCell>{row.academicYear}</TableCell>
                    <TableCell>{row.amountSanctioned}</TableCell>
                    <TableCell>{row.amountReceived}</TableCell>
                  </TableRow>
                ))
              ) : (
                <TableRow>
                  <TableCell colSpan={7} align="center">
                    No data available
                  </TableCell>
                </TableRow>
              )}
            </TableBody>
          </Table>
        </TableContainer>
      )}
    </Container>
  </div>
 );
};

export default ViewData;
```

App.jsx :

```
import Login from "./Pages/Login";
import Signup from "./Pages/Signup";
import Form from "./Pages/Form";
import ForgotPassword from "./Pages/ForgotPassword";

import {
  BrowserRouter as Router,
  Routes,
  Route,
  Navigate,
} from "react-router-dom";
import "./App.css";
import ViewData from "./Pages/ViewData";
import OTP from "./Pages/OTP";
```

```
const ProtectedRoute = ({ children }) => {
  const token = localStorage.getItem("token");
  if (!token) {
    return <Navigate to="/" replace />;
  }
  return children;
};

function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Login />} />
        <Route path="/signup" element={<Signup />} />
        <Route path="/verifyotp" element={<OTP />} />
        <Route path="/forgotPsd" element={<ForgotPassword />} />

        <Route
          path="/form"
          element={
            <ProtectedRoute>
              <Form />
            </ProtectedRoute>
          }
        />

        <Route
          path="/view"
          element={
            <ProtectedRoute>
              <ViewData />
            </ProtectedRoute>
          }
        />

        <Route path="*" element={<Navigate to="/" replace />} />
      </Routes>
    </Router>
  );
}

export default App;
```

main.jsx :

```
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.jsx'

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <App />
  </StrictMode>,
)
```

Navbar.jsx :

```
import React, { useState, useRef, useEffect } from "react";
import { useNavigate, Link, useLocation } from "react-router-dom";
import "../CSS/navBar.css";

const NavBar = () => {
  const navigate = useNavigate();
  const location = useLocation();
  const [dropdownOpen, setDropdownOpen] = useState(false);
  const dropdownRef = useRef(null);
  const user = JSON.parse(localStorage.getItem("user"));

  const handleLogout = () => {
    localStorage.removeItem("token");
    localStorage.removeItem("user");
    navigate("/", { replace: true });
  };

  useEffect(() => {
    const handleClickOutside = (event) => {
      if (dropdownRef.current && !dropdownRef.current.contains(event.target)) {
        setDropdownOpen(false);
      }
    };

    document.addEventListener("mousedown", handleClickOutside);
    return () => document.removeEventListener("mousedown", handleClickOutside);
  }, []);
```

```
  return (
    <nav className="navbar">
      <div className="nav-brand">
        <h2>Consultancy Services</h2>
      </div>

      <div className="nav-items">
        <div className="nav-links">
          <Link
            to="/form"
            className={`nav-link ${
              location.pathname === "/form" ? "active" : ""
            }`}>
            Data Entry
          </Link>
          <Link
            to="/view"
            className={`nav-link ${
              location.pathname === "/view" ? "active" : ""
            }`}>
            View Data
          </Link>
        </div>

        <div className="user-menu" ref={dropdownRef}>
          <div
            className="user-icon-wrapper"
            onClick={() => setDropdownOpen(!dropdownOpen)}>
            {user?.email && <span className="user-email">{user.email}</span>}
            {dropdownOpen && (
              <div className="dropdown-content">
                <div className="user-info">
                  <span>{user?.email}</span>
                </div>
                <button onClick={handleLogout} className="logout-btn">
                  Logout
                </button>
              </div>
            )}
          </div>
        </div>
      </div>
    </nav>
  );
};

export default NavBar;
```

api_communicator.jsx :

```
import axios from "axios";

const API_BASE_URL = "http://localhost:8000/api";

export const loginUser = async (credentials) => {
    try {
        const response = await axios.post(`${API_BASE_URL}/login`, {
            email: credentials.email,
            password: credentials.psd
        });

        if (response.status === 200) {
            const { token, user } = response.data;
            localStorage.setItem('token', token);
            localStorage.setItem('user', JSON.stringify(user));
            return { success: true, data: response.data };
        }
    } catch (error) {
        return {
            success: false,
            error: error.response?.data?.message || "An error occurred during
login"
        };
    }
};

export const registerUser = async (credentials) => {
    try {
        const response = await axios.post(`${API_BASE_URL}/register`, {
            name: credentials.name,
            email: credentials.email,
            password: credentials.psd
        });

        if (response.status === 201) {
            return { success: true, data: response.data };
        }
    } catch (error) {
        return {
            success: false,
```

```
            error: error.response?.data?.message || "An error occurred during
registration"
        };
    }
};


export const forgotPsd = async (credentials) => {
    try {
        const response = await axios.put(`${API_BASE_URL}/forgotPsd`, {

            email: credentials.email,
            psd: credentials.psd
        });

        if (response.status === 200) {
            return { success: true, data: response.data };
        }
    } catch (error) {
        return {
            success: false,
            error: error.response?.data?.message || "An error occurred during
registration"
        };
    }
};

export const formsubmission = async (formData) => {
    try {
        const token = localStorage.getItem('token');
        const response = await axios.post(
            "http://localhost:8000/api/submit",
            formData,
            {
                headers: {
                    "Content-Type": "multipart/form-data",
                    "Authorization": `Bearer ${token}`
                },
            }
        );
        return response;
    } catch (error) {
        return {
            success: false,
            error: error.response?.data?.message || "An error occurred during
form submission"
        };
```

```
    }
};

export const generateOTP = async (email) => {
  try {
    const response = await axios.post(`${API_BASE_URL}/generateOTP`, { email });
    if (response.status === 200) {
      return { success: true, message: response.data.message };
    }
  } catch (error) {
    return {
      success: false,
      error: error.response?.data?.message || "Failed to send OTP",
    };
  }
};

export const verifyOtp = async (data) => {
  try {
    const response = await axios.post(`${API_BASE_URL}/verifyOTP`, data);
    if (response.status === 200) {
      return {
        success: true,
        message: response.data.message,
        status: 200,
      };
    }
  } catch (error) {
    return {
      success: false,
      error: error.response?.data?.message || "Failed to verify OTP",
    };
  }
};
```

Index.html:

```
<!doctype html>
<html lang="en">
  <head>
```

```
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Vite + React</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

CSS Files :

Login.css :

```
.login_container {
    display: flex;
    flex-direction: row;
    width: 60%;
    height: 80vh;
    margin: auto;
    margin-top: 50px;
    background-color: aqua;
    border-radius: 50px;
    overflow: hidden;
    transition:box-shadow .5s ease-in-out;

}
.login_container:hover{
    box-shadow: 5px 5px 20px rgb(0, 0, 0);
}

.brand {
    width: 150%; /* Adjust width for better balance */
    background-image: url("../assets/login_img.png");
    background-size: cover;
    background-position: center;
    background-repeat: no-repeat;
}

.login {
```

```
    width: 100%;
    display: flex;
    justify-content: center;
    align-items: center;
    background-color: rgba(82, 82, 78, 0.688);

}

.login form {
    display: flex;
    flex-direction: column;
    align-items: center;

}
.login form h2 {
   padding:0px 50px;
   background-color: white;
   border-radius: 10px;
   box-shadow: 2px 2px 2px rgb(0, 0, 0);

}

.login input {
    margin: 10px 0; /* Spacing between inputs */
    padding: 10px;
    border: none;
    border-radius: 5px;
    background-color: white;
    border: 1px solid rgb(0, 0, 0);
    width:250px;
}
.login input[type="submit"] {
   width:100%;
   background-color: rgba(0, 0, 0, 0.692);
   color: white;
    font-weight: bold;
    cursor: pointer;
    transition: background 0.2s ease-out;

}

.login input[type="submit"]:hover {
    background-color: rgba(255, 255, 255, 0.344);;
}
```

Navbar.css :

```
.navbar {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 1rem 2rem;
  background-color: #fff;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
  position: sticky;
  top: 0;
  z-index: 1000;
}

.nav-brand h2 {
  margin: 0;
  color: #333;
  font-size: 1.5rem;
}

.nav-items {
    display: flex;
}

.nav-links {
  display: flex;
  gap: 2;
}

.nav-link {
  text-decoration: none;
  color: #333;
  font-weight: 500;
  padding: 0.5rem 1rem;
  border-radius: 4px;
  transition: all 0.3s ease;
}

.nav-link:hover {
  background-color: #f0f0f0;
}

.nav-link.active {
  background-color: #007bff;
  color: white;
}
```

```css
.user-menu {
  position: relative;
}

.user-icon-wrapper {
  display: flex;
  align-items: center;
  gap: 0.5rem;
  cursor: pointer;
  padding: 0.5rem;
  border-radius: 4px;
  transition: background-color 0.3s;
}

.user-icon-wrapper:hover {
  background-color: #f0f0f0;
}

.user-icon {
  font-size: 24px;
  color: #333;
}

.user-email {
  font-size: 0.9rem;
  color: #666;
}

.dropdown-content {
  position: absolute;
  right: 0;
  top: 100%;
  background-color: #fff;
  min-width: 220px;
  box-shadow: 0 2px 10px rgba(0,0,0,0.1);
  border-radius: 4px;
  padding: 0.5rem;
  margin-top: 0.5rem;
}

.user-info {
  display: flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.75rem;
  border-bottom: 1px solid #eee;
  color: #333;
```

```css
}

.user-info .material-icons {
  font-size: 20px;
  color: #666;
}

.logout-btn {
  width: 100%;
  display: flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.75rem;
  margin-top: 0.5rem;
  border: none;
  background-color: transparent;
  color: #dc3545;
  cursor: pointer;
  transition: background-color 0.3s;
  border-radius: 4px;
}

.logout-btn:hover {
  background-color: #fff5f5;
}

.logout-btn .material-icons {
  font-size: 20px;
}

@media (max-width: 768px) {
  .navbar {
    padding: 1rem;
  }

  .nav-brand h2 {
    font-size: 1.2rem;
  }

  .nav-links {
    gap: 1rem;
  }

  .user-email {
    display: none;
  }
}
```

Backend Code :

UserController.js :

```
import User from "../Model/modelSchema.js";
import jwt from "jsonwebtoken";
import multer from "multer";
import path from "path";
import bcrypt from "bcryptjs";
import fs from "fs";
import { google } from "googleapis";
import { createFolder, uploadFile } from '../Configuration/googleDriveSetup.js';
import  dotenv  from 'dotenv';
import nodemailer from 'nodemailer';

dotenv.config();

export const getUserEmailRoute = async(req,res)=>{
  const authHeader = req.headers.authorization;

  if(!authHeader || !authHeader.startsWith("Bearer ")){
    return res.status(401).json({error: "Authorization Token missing"});
  }

  const token = authHeader.split(" ")[1];

  try {
    const decoded = jwt.verify(token, process.env.JWT_SECRET);
    const userId = decoded.id;

    const user = await User.findById(userId).select("email");

    if (!user) {
      return res.status(404).json({error:"user not found"});
    }

    return res.status(200).json({message: user.email});
  } catch (error) {
    console.error("Error decoding token or fetching user:", error);
    return res.status(500).json({error:"Something went wrong"});
```

```
  }
}

export const login = async (req, res) => {
  try {
    const { email, password } = req.body;

    if (!email || !password) {
      return res.status(400).json({
        success: false,
        message: "Please provide email and password",
      }
    )};

    const user = await User.findOne({ email });
    if (!user) {
      return res.status(401).json({
        success: false,
        message: "Invalid credentials",
      });
    }

    const isMatch = await user.matchPassword(password);
    if (!isMatch) {
      return res.status(401).json({
        success: false,
        message: "Invalid credentials",
      });
    }

    const token = jwt.sign({ id: user._id }, process.env.JWT_SECRET, {
      expiresIn: "1h",
    });

    res.status(200).json({
      success: true,
      token,
      user: {
        id: user._id,
        email: user.email,
      },
    });
  } catch (error) {
    res.status(500).json({
      success: false,
      message: "Error in login",
      error: error.message,
```

```
    });
  }
};

export const register = async (req, res) => {
  try {
    const { name, email, password } = req.body;

    const userExists = await User.findOne({ email });
    if (userExists) {
      return res.status(400).json({
        success: false,
        message: "User already exists",
      });
    }

    const user = await User.create({
      name,
      email,
      password,
    });

    res.status(201).json({
      success: true,
      message: "User created successfully",
      user: {
        id: user._id,
        email: user.email,
      },
    });
  } catch (error) {
    res.status(500).json({
      success: false,
      message: "Error in registration",
      error: error.message,
    });
  }
};


export const forgotPsd = async(req,res)=>{
  const {email,psd} = req.body;
  try{
    const user = await User.findOne({ email :email });
  if (!user) {
    return res.status(404).json({ success: false, message: "User not found" });
  }
```

```
  user.password = psd;
  await user.save();
  res.status(200).json({ success: true, message: "Password updated successfully"
});
  }

  catch(error){
    console.log(error);
    res.status(500).json({message:error});
  }



};

const otpStore = {};

export const generateOTP = async (req, res) => {
  try {
    const {email} = req.body;
    if (!email) {
      return res.status(400).json({ error: "Invalid authorization header" });
    }

    const transporter = nodemailer.createTransport({
      service: "gmail",
      auth: {
        user: process.env.EMAIL_USER,
        pass: process.env.EMAIL_PASS,
      },
    });

    const otp = Math.floor(100000 + Math.random() * 900000).toString();
    otpStore[email] = { otp, expiresAt: Date.now() + 5 * 60 * 1000 };

    const mailOptions = {
      from: process.env.EMAIL_USER,
      to: email,
      subject: "Your OTP Code",
      html: `<h3>Your OTP is: <b>${otp}</b></h3><p>It is valid for 5
minutes.</p>`,
    };

    await transporter.sendMail(mailOptions);
    res.json({ message: "OTP sent successfully" });
  } catch (error) {
```

```
    console.error("Error in generateOTP:", error);
    return res.status(500).json({ error: "Failed to send OTP" });
  }
};

export const verifyOtp = async (req, res) => {
  const { email, otp } = req.body;

  if (!email || !otp) {
    return res.status(400).json({ error: "Email and OTP are required" });
  }

  const record = otpStore[email];
  if (!record) {
    return res.status(400).json({ error: "OTP not found" });
  }

  if (Date.now() > record.expiresAt) {
    delete otpStore[email];
    return res.status(400).json({ error: "OTP expired" });
  }

  if (record.otp !== otp) {
    return res.status(400).json({ error: "Invalid OTP" });
  }

  delete otpStore[email];
  res.json({ message: "OTP verified successfully" });
};

const getUserEmail = async(authHeader)=>{

  if(!authHeader || !authHeader.startsWith("Bearer ")){
    throw new Error("Authorization token missing or invalid" );
  }

  const token = authHeader.split(" ")[1];

  try {
    const decoded = jwt.verify(token, process.env.JWT_SECRET); // ✅ decode
token
    const userId = decoded.id;

    const user = await User.findById(userId).select("email"); // Fetch only
email

    if (!user) {
```

```
      throw new Error("User not found");
    }

    return user.email;
  } catch (error) {
    console.error("Error decoding token or fetching user:", error);
    throw new Error("Erro in fetching email");
  }
}

const uploadDir = "uploads";
if (!fs.existsSync(uploadDir)) {
  fs.mkdirSync(uploadDir);
}

const storage = multer.diskStorage({
  destination: (req, file, cb) => {
    const projectTitle = req.body.projectTitle || "defaultProject";
    const projectFolder = path.join(uploadDir, projectTitle);
    if (!fs.existsSync(projectFolder)) {
      fs.mkdirSync(projectFolder, { recursive: true });
    }
    cb(null, projectFolder);
  },
  filename: (req, file, cb) => {
    cb(null, `${Date.now()}-${file.originalname}`);
  },
});

const fileFilter = (req, file, cb) => {
  if (file.fieldname === "billProof") {
    const allowedTypes = [".pdf", ".jpg", ".jpeg", ".png"];
    const ext = path.extname(file.originalname).toLowerCase();
    if (allowedTypes.includes(ext)) {
      cb(null, true);
    } else {
      cb(new Error("Invalid file type for bill proof. Only PDF, JPG, JPEG, PNG
allowed."));
    }
  } else if (file.fieldname === "agreement") {
    const ext = path.extname(file.originalname).toLowerCase();
    if (ext === ".pdf") {
      cb(null, true);
    } else {
      cb(new Error("Invalid file type for signed agreement. Only PDF
allowed."));
    }
```

```
  } else {
    cb(null, false);
  }
};

const upload = multer({
  storage,
  fileFilter,
}).fields([
  { name: "billProof", maxCount: 1 },
  { name: "agreement", maxCount: 1 },
]);

export const handleSubmit = async (req, res) => {
  upload(req, res, async (err) => {
    if (err) {
      return res.status(400).json({ error: err.message });
    }

    const {
      industryName,
      projectDuration,
      projectTitle,
      pi,
      coPI,
      academicYear,
      amountSanctioned,
      amountReceived,
      billDetails,
      studentDetails,
      projectSummary,
    } = req.body;

    if (!req.files || !req.files.billProof || !req.files.agreement) {
      return res.status(400).json({ error: "Both Bill Proof and Signed Agreement
files are required." });
    }

    const billProofFile = req.files.billProof[0];
    const agreementFile = req.files.agreement[0];

    try {
      const driveFolderId = await createFolder(projectTitle);
      console.log("Created Google Drive folder with ID:", driveFolderId);

      const billProofDriveId = await uploadFile(
        billProofFile.path,
```

```
        billProofFile.filename,
        billProofFile.mimetype,
        driveFolderId
      );

      const agreementDriveId = await uploadFile(
        agreementFile.path,
        agreementFile.filename,
        agreementFile.mimetype,
        driveFolderId
      );

      const billProofLink =
`https://drive.google.com/file/d/${billProofDriveId}/view`;
      const agreementLink =
`https://drive.google.com/file/d/${agreementDriveId}/view`;

      const auth = new google.auth.GoogleAuth({
        keyFile: "C:/Users/sriva/OneDrive/Desktop/sem/sem6/Internet
Programming/Consultancy-Service/Backend/Credentials/wise-bongo-451704-r2-11c606e
1ced8.json",
        scopes: ["https://www.googleapis.com/auth/spreadsheets"],
      });
      const client = await auth.getClient();
      const sheets = google.sheets({ version: "v4", auth: client });

      const spreadsheetId = process.env.GOOGLE_SPREADSHEET_ID;
      const range = "Sheet1!A1";

      const newRow = [
        industryName,
        projectDuration,
        projectTitle,
        pi,
        coPI,
        academicYear,
        amountSanctioned,
        amountReceived,
        billDetails,
        billProofLink,
        agreementLink,
        studentDetails,
        projectSummary,
      ];

      await sheets.spreadsheets.values.append({
        spreadsheetId,
```

```
        range,
        valueInputOption: "RAW",
        resource: {
          values: [newRow],
        },
      });

      const userEmail = await getUserEmail(req.headers.authorization)

      await notifyUser(userEmail, {
        industryName,
        projectDuration,
        projectTitle,
        pi,
        coPI,
        academicYear,
        amountSanctioned,
        amountReceived,
        studentDetails,
        projectSummary,
      });

      res.status(200).json({
        message: "Data successfully appended to Google Sheet and files uploaded
to Google Drive.",
        billProofLink,
        agreementLink,
      });
    } catch (error) {
      console.error("Error processing submission:", error);
      res.status(500).json({ error: "An error occurred while processing the
submission." });
    }
  });
};

export const notifyUser = async(toEmail, data) => {
  const {
    industryName,
    projectDuration,
    projectTitle,
    pi,
    coPI,
    academicYear,
    amountSanctioned,
    amountReceived,
    studentDetails,
```

```
    projectSummary,
  } = data;

  const transporter = nodemailer.createTransport({
    service: 'gmail',
    auth: {
      user: process.env.EMAIL_USER,
      pass: process.env.EMAIL_PASS
    }
  });

  const mailOptions = {
    from: process.env.EMAIL_USER,
    to: toEmail,
    subject: `📢 New Consultancy Project Entry: ${projectTitle}`,
    html: `
      <h3>New Project Submitted</h3>
      <p><strong>Industry Name:</strong> ${industryName}</p>
      <p><strong>Project Title:</strong> ${projectTitle}</p>
      <p><strong>PI:</strong> ${pi}</p>
      <p><strong>Co-PI:</strong> ${coPI}</p>
      <p><strong>Academic Year:</strong> ${academicYear}</p>
      <p><strong>Duration:</strong> ${projectDuration}</p>
      <p><strong>Sanctioned Amount:</strong> ₹${amountSanctioned}</p>
      <p><strong>Received Amount:</strong> ₹${amountReceived}</p>
      <p><strong>Student Details:</strong> ${studentDetails}</p>
      <p><strong>Project Summary:</strong> ${projectSummary}</p>`
  };

  try {
    const info = await transporter.sendMail(mailOptions);
    console.log("Notification sent successfully:", info.messageId);
  } catch (error) {
    console.log("An error occured",error);
  }
}

export const getSubmissions = async (req, res) => {

  try {
    const auth = new google.auth.GoogleAuth({
      keyFile: "C:/Users/sriva/OneDrive/Desktop/sem/sem6/Internet
Programming/Consultancy-Service/Backend/Credentials/wise-bongo-451704-r2-11c606e
1ced8.json",
      scopes: ["https://www.googleapis.com/auth/spreadsheets.readonly"],
    });
```

```
    const client = await auth.getClient();
    const sheets = google.sheets({ version: "v4", auth: client });

    const spreadsheetId = process.env.GOOGLE_SPREADSHEET_ID;
    const range = "Sheet1!A2:M"; // Skip header row (assuming headers are in row
1)

    const response = await sheets.spreadsheets.values.get({
      spreadsheetId,
      range,
    });

    const rows = response.data.values;

    if (!rows || rows.length === 0) {
      return res.status(200).json({ data: [], message: "No data found in the
sheet." });
    }

    const formattedData = rows.map((row) => ({
      industryName: row[0] || "",
      projectDuration: row[1] || "",
      projectTitle: row[2] || "",
      pi: row[3] || "",
      coPI: row[4] || "",
      academicYear: row[5] || "",
      amountSanctioned: row[6] || "",
      amountReceived: row[7] || "",
      billDetails: row[8] || "",
      billProofLink: row[9] || "",
      agreementLink: row[10] || "",
      studentDetails: row[11] || "",
      projectSummary: row[12] || "",
    }));

    res.status(200).json({ data: formattedData });
  } catch (error) {
    console.error("Error fetching data from Google Sheet:", error);
    res.status(500).json({ error: "An error occurred while fetching data from
the sheet." });
  }
};

export const getByPI = async (req, res) => {
  const { pi } = req.params;

  try {
```

```
    const auth = new google.auth.GoogleAuth({
      keyFile: "C:/Users/sriva/OneDrive/Desktop/sem/sem6/Internet
Programming/Consultancy-Service/Backend/Credentials/wise-bongo-451704-r2-11c606e
1ced8.json",
      scopes: ["https://www.googleapis.com/auth/spreadsheets.readonly"],
    });

    const client = await auth.getClient();
    const sheets = google.sheets({ version: "v4", auth: client });

    const spreadsheetId = process.env.GOOGLE_SPREADSHEET_ID;
    const range = "Sheet1!A2:M";

    const response = await sheets.spreadsheets.values.get({
      spreadsheetId,
      range,
    });

    const rows = response.data.values;

    if (!rows || rows.length === 0) {
      return res.status(200).json({ data: [], message: "No data found in the
sheet." });
    }

    let formattedData = rows.map((row) => ({
      industryName: row[0] || "",
      projectDuration: row[1] || "",
      projectTitle: row[2] || "",
      pi: row[3] || "",
      coPI: row[4] || "",
      academicYear: row[5] || "",
      amountSanctioned: row[6] || "",
      amountReceived: row[7] || "",
      billDetails: row[8] || "",
      billProofLink: row[9] || "",
      agreementLink: row[10] || "",
      studentDetails: row[11] || "",
      projectSummary: row[12] || "",
    }));

    const filteredData = formattedData.filter(
      (item) => item.pi.toLowerCase().includes(pi.toLowerCase())
    );

    res.status(200).json({ data: filteredData });
  } catch (error) {
```

```
    console.error("Error filtering data by PI:", error);
    res.status(500).json({ error: "An error occurred while fetching data by PI."
});
  }
};


export const getByCoPI = async (req, res) => {
  const { coPI } = req.params;

  try {
    const auth = new google.auth.GoogleAuth({
      keyFile: "C:/Users/sriva/OneDrive/Desktop/sem/sem6/Internet
Programming/Consultancy-Service/Backend/Credentials/wise-bongo-451704-r2-11c606e
1ced8.json",
      scopes: ["https://www.googleapis.com/auth/spreadsheets.readonly"],
    });

    const client = await auth.getClient();
    const sheets = google.sheets({ version: "v4", auth: client });

    const spreadsheetId = process.env.GOOGLE_SPREADSHEET_ID;
    const range = "Sheet1!A2:M"; // Skip header

    const response = await sheets.spreadsheets.values.get({
      spreadsheetId,
      range,
    });

    const rows = response.data.values;

    if (!rows || rows.length === 0) {
      return res.status(200).json({ data: [], message: "No data found in the
sheet." });
    }

    const formattedData = rows.map((row) => ({
      industryName: row[0] || "",
      projectDuration: row[1] || "",
      projectTitle: row[2] || "",
      pi: row[3] || "",
      coPI: row[4] || "",
      academicYear: row[5] || "",
      amountSanctioned: row[6] || "",
      amountReceived: row[7] || "",
      billDetails: row[8] || "",
      billProofLink: row[9] || "",
```

```
      agreementLink: row[10] || "",
      studentDetails: row[11] || "",
      projectSummary: row[12] || "",
    }));

    const filteredData = formattedData.filter((entry) =>
      entry.coPI.toLowerCase().includes(coPI.toLowerCase())
    );

    res.status(200).json({ data: filteredData });
  } catch (error) {
    console.error("Error filtering by Co-PI:", error);
    res.status(500).json({ error: "An error occurred while filtering by Co-PI."
});
  }
};

export const getBySanctionedAmountRange = async (req, res) => {
  const { minAmount, maxAmount } = req.params;

  if (!minAmount || !maxAmount) {
    return res.status(400).json({ error: "Please provide both minAmount and
maxAmount" });
  }

  try {

    const auth = new google.auth.GoogleAuth({
      keyFile: "C:/Users/sriva/OneDrive/Desktop/sem/sem6/Internet
Programming/Consultancy-Service/Backend/Credentials/wise-bongo-451704-r2-11c606e
1ced8.json",
      scopes: ["https://www.googleapis.com/auth/spreadsheets.readonly"],
    });

    const client = await auth.getClient();
    const sheets = google.sheets({ version: "v4", auth: client });

    const spreadsheetId = process.env.GOOGLE_SPREADSHEET_ID;
    const range = "Sheet1!A2:M";

    const response = await sheets.spreadsheets.values.get({
      spreadsheetId,
      range,
    });

    const rows = response.data.values;
```

```
    if (!rows || rows.length === 0) {
      return res.status(200).json({ data: [], message: "No data found in the
sheet." });
    }

    // Format the data from the sheet
    const formattedData = rows.map((row) => ({
      industryName: row[0] || "",
      projectDuration: row[1] || "",
      projectTitle: row[2] || "",
      pi: row[3] || "",
      coPI: row[4] || "",
      academicYear: row[5] || "",
      amountSanctioned: row[6] || "",
      amountReceived: row[7] || "",
      billDetails: row[8] || "",
      billProofLink: row[9] || "",
      agreementLink: row[10] || "",
      studentDetails: row[11] || "",
      projectSummary: row[12] || "",
    }));


    const filteredData = formattedData.filter((item) => {
      const sanctionedAmount =
parseFloat(item.amountSanctioned.replace(/[^0-9.]/g, ''));


      return sanctionedAmount >= parseFloat(minAmount) && sanctionedAmount <=
parseFloat(maxAmount);
    });

    res.status(200).json({ data: filteredData });

  } catch (error) {
    console.error("Error fetching or filtering data:", error);
    res.status(500).json({ error: "An error occurred while fetching or filtering
data." });
  }
};

export const getByIndustryName = async (req, res) => {
  const { industryName } = req.params;

  try {
    const auth = new google.auth.GoogleAuth({
```

```
      keyFile: "C:/Users/sriva/OneDrive/Desktop/sem/sem6/Internet
Programming/Consultancy-Service/Backend/Credentials/wise-bongo-451704-r2-11c606e
1ced8.json",
      scopes: ["https://www.googleapis.com/auth/spreadsheets.readonly"],
    });

    const client = await auth.getClient();
    const sheets = google.sheets({ version: "v4", auth: client });

    const spreadsheetId = process.env.GOOGLE_SPREADSHEET_ID;
    const range = "Sheet1!A2:M";

    const response = await sheets.spreadsheets.values.get({
      spreadsheetId,
      range,
    });

    const rows = response.data.values;

    if (!rows || rows.length === 0) {
      return res.status(200).json({ data: [], message: "No data found in the
sheet." });
    }

    const formattedData = rows.map((row) => ({
      industryName: row[0] || "",
      projectDuration: row[1] || "",
      projectTitle: row[2] || "",
      pi: row[3] || "",
      coPI: row[4] || "",
      academicYear: row[5] || "",
      amountSanctioned: row[6] || "",
      amountReceived: row[7] || "",
      billDetails: row[8] || "",
      billProofLink: row[9] || "",
      agreementLink: row[10] || "",
      studentDetails: row[11] || "",
      projectSummary: row[12] || "",
    }));

    const filteredData = formattedData.filter((entry) => {
      // Ensure industryName exists before calling toLowerCase()
      return entry.industryName &&
entry.industryName.toLowerCase().includes(industryName.toLowerCase());
    });

    res.status(200).json({ data: filteredData });
```

```
  } catch (error) {
    console.error("Error filtering by Industry Name:", error); // Updated error
message
    res.status(500).json({ error: "An error occurred while filtering by Industry
Name." }); // Updated error message
  }
};
```

modelSchema.js :

```
import mongoose from "mongoose";
import bcrypt from "bcryptjs";

const userSchema = new mongoose.Schema({
    name:{
        type: String,
        required: [true, "Please provide a name"],
        trim: true
    },
    email: {
        type: String,
        required: [true, "Please provide an email"],
        unique: true,
    },
    password: {
        type: String,
        required: [true, "Please provide a password"],
        minlength: 6
    }
}, { timestamps: true });

userSchema.pre('save', async function(next) {
    if (!this.isModified('password')) return next();
    this.password = await bcrypt.hash(this.password, 12);
    next();
});

userSchema.methods.matchPassword = async function(enteredPassword) {
    return await bcrypt.compare(enteredPassword, this.password);
```

```
};
const User = mongoose.model('User', userSchema);

export default User;
```

Routes - > index.js :

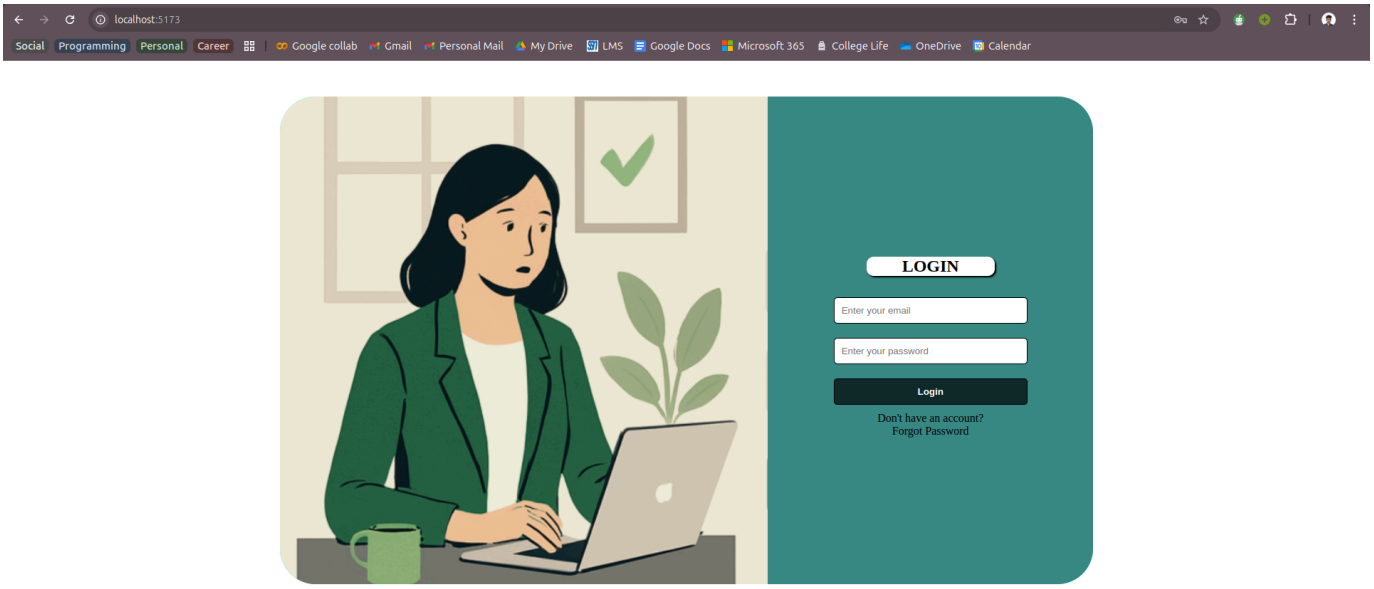
```
import express from 'express';
import { handleSubmit, getUserEmailRoute,login,generateOTP, verifyOtp,
register,forgotPsd,getSubmissions,getByPI,getByCoPI,getBySanctionedAmountRange,
getByIndustryName} from '../Controller/userController.js';


const router = express.Router();
router.post('/login', login);
router.post('/register', register);
router.post('/submit',handleSubmit);
router.put("/forgotPsd",forgotPsd);
router.get("/user/email",getUserEmailRoute);
router.get("/getData",getSubmissions);
router.get("/getDataByPI/:pi",getByPI);
router.get("/getDataBycoPI/:coPI",getByCoPI);
router.get("/getDataByIndustryName/:industryName",getByIndustryName);
router.post("/generateOTP", generateOTP);
router.post("/verifyOTP", verifyOtp);


router.get("/getDataBySanctionAmt/:minAmount/:maxAmount",getBySanctionedAmountRa
nge);



export default router;
```

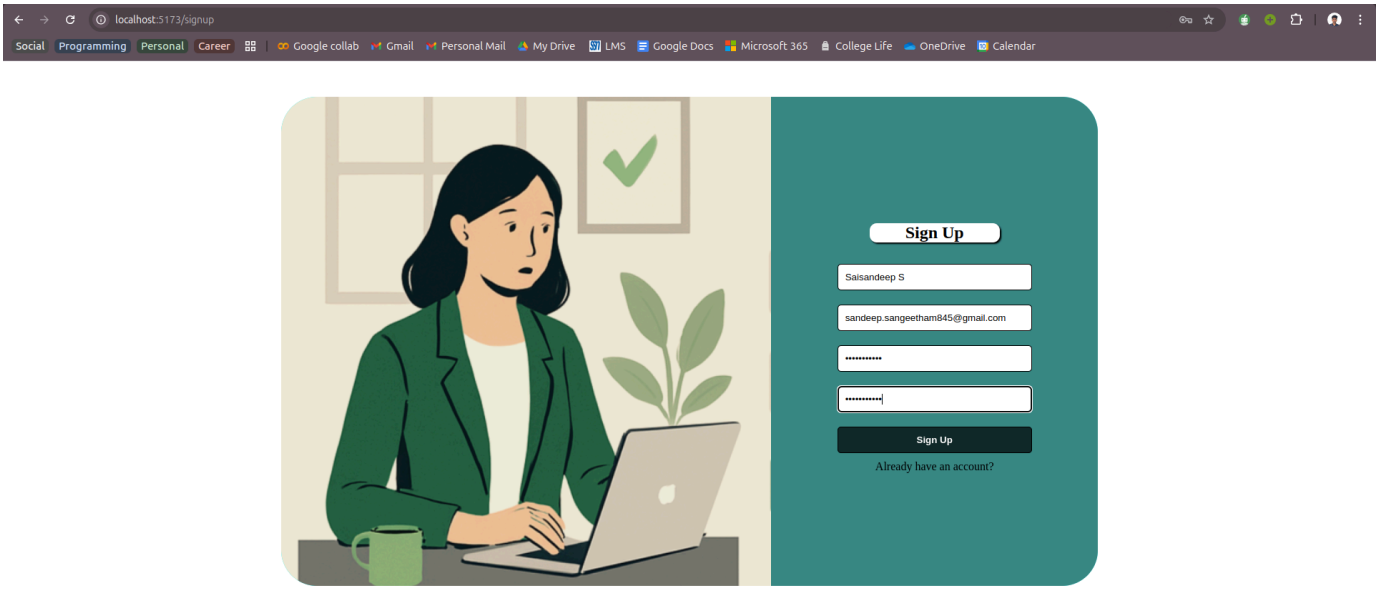
server.js :

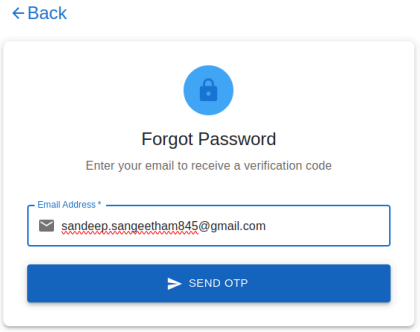
```
import express from 'express';
import cors from 'cors';
import dotenv from 'dotenv';
import routes from './Route/index.js';
import bodyParser from 'body-parser';
import morgan from 'morgan';
import mongoose from 'mongoose';

const app = express();

dotenv.config();
```

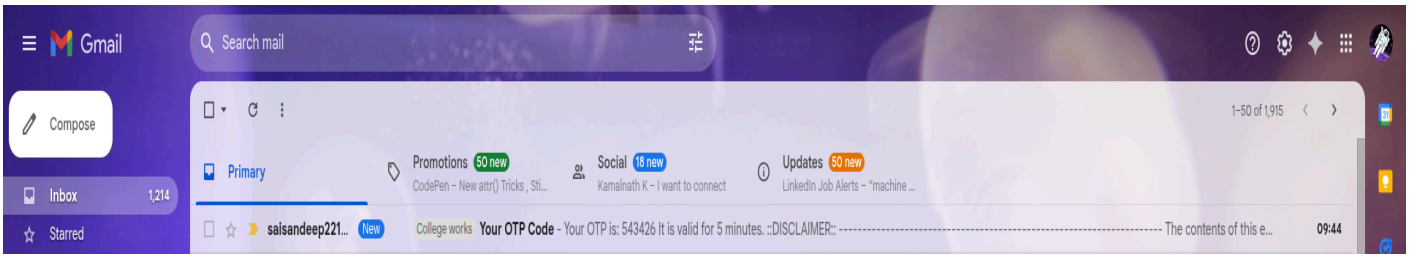
```
app.use(cors({origin: "http://localhost:5173", credentials: true}));
app.use(bodyParser.json());
app.use(morgan('dev'));

app.use('/api', routes);

app.use((err, req, res,next) => {
  console.error(err.stack);
  res.status(500).json({ message: 'Something broke!' });
});

const PORT = process.env.PORT || 5000;
const MongoURI = process.env.MONGO_URI;

mongoose.connect(MongoURI)
  .then(() => {
    console.log("MongoDB connected successfully");
    app.listen(PORT, () => {
      console.log(`Server is running on port ${PORT}`);
    });
  })
  .catch((err) => {
    console.error('MongoDB connection error:', err);
  });

export default app;
```

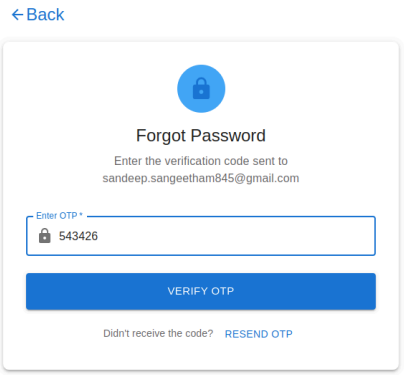
Outputs :

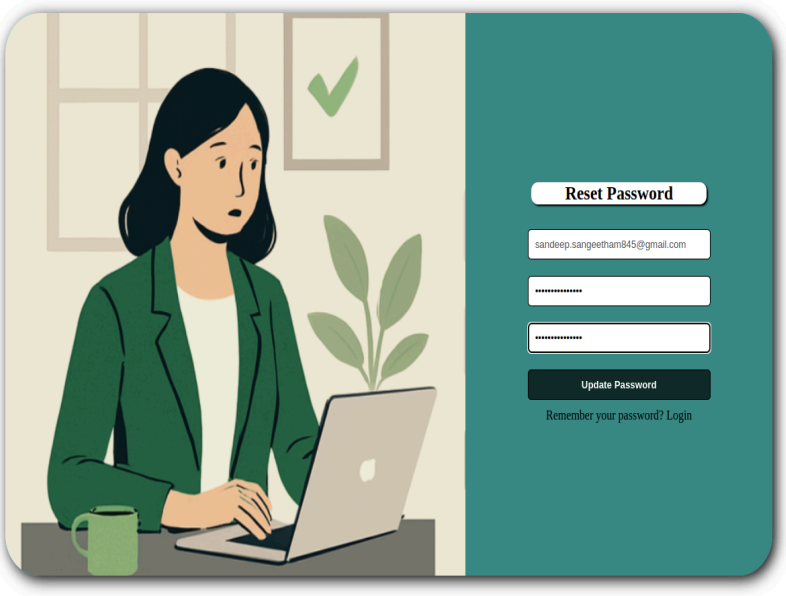

localhost:5173
Social
Programming
Personal
Career
Google collab
Gmail
Personal Mail
My Drive
LMS
Google Docs
Microsoft 365
College Life
OneDrive
Calendar
LOGIN
Enter your email
Enter your password
Login
Don't have an account?
Forgot Password

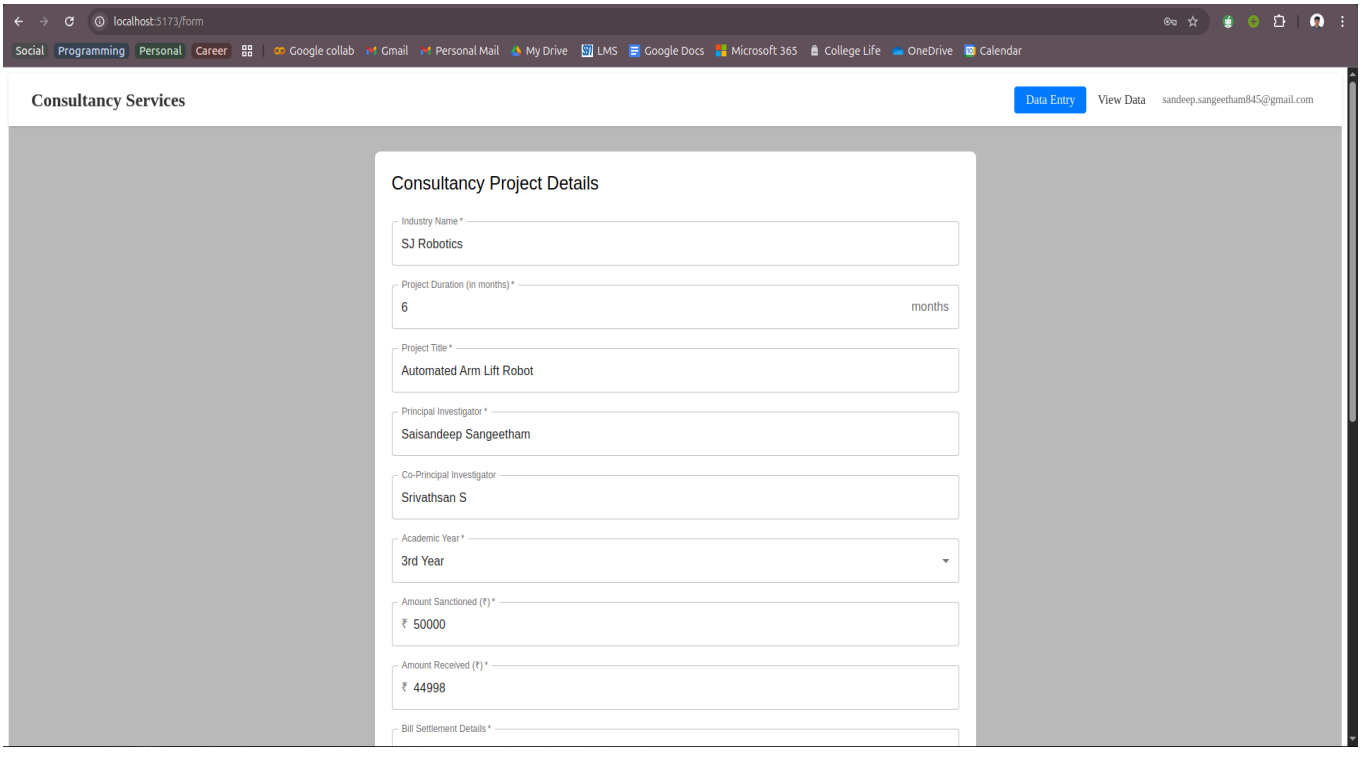

localhost:5173/signup
Social
Programming
Personal
Career
Google collab
Gmail
Personal Mail
My Drive
LMS
Google Docs
Microsoft 365
College Life
OneDrive
Calendar
Sign Up
Salsandeep S
sandeep.sangeetham845@gmail.com
Sign Up
Already have an account?

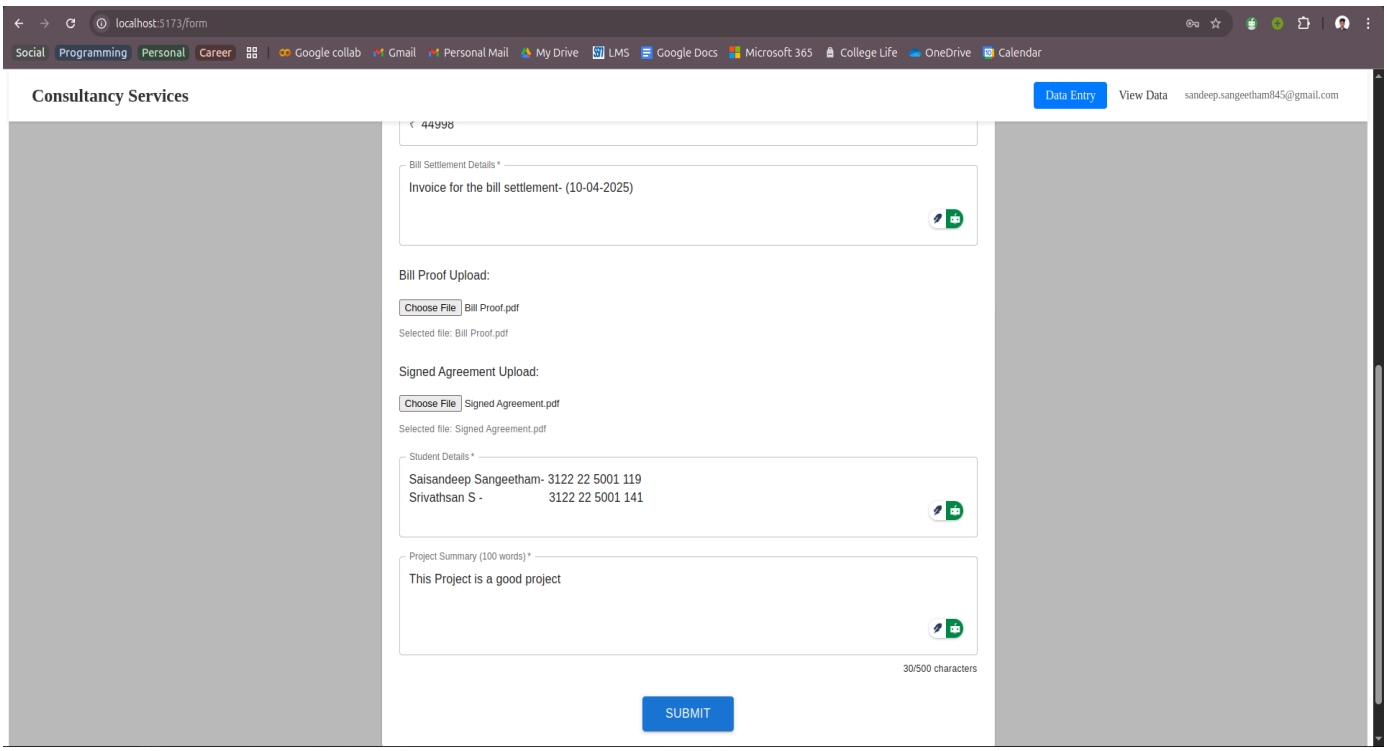

←Back
Forgot Password
Enter your email to receive a verification code
Email Address *
sandeep.sangeetham845@gmail.com
SEND OTP

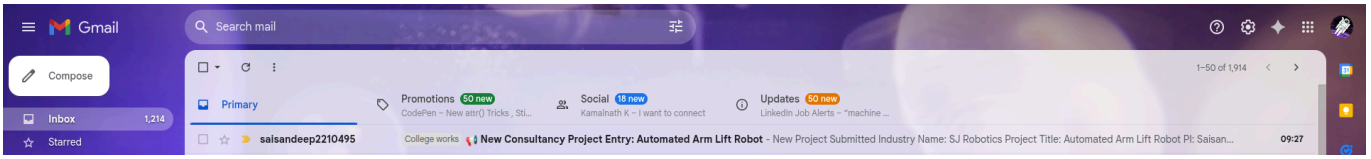

Gmail
Search mail
Compose
Inbox 1,214
Starred
1–50 of 1,915
Primary
Promotions 50 new
CodePen – New attr() Tricks , Sti...
Social 18 new
Kamalnath K – I want to connect
Updates 50 new
LinkedIn Job Alerts – "machine ...
saisandeep221... New
College works Your OTP Code - Your OTP is: 543426 It is valid for 5 minutes. ::DISCLAIMER:: ---------- The contents of this e...
09:44

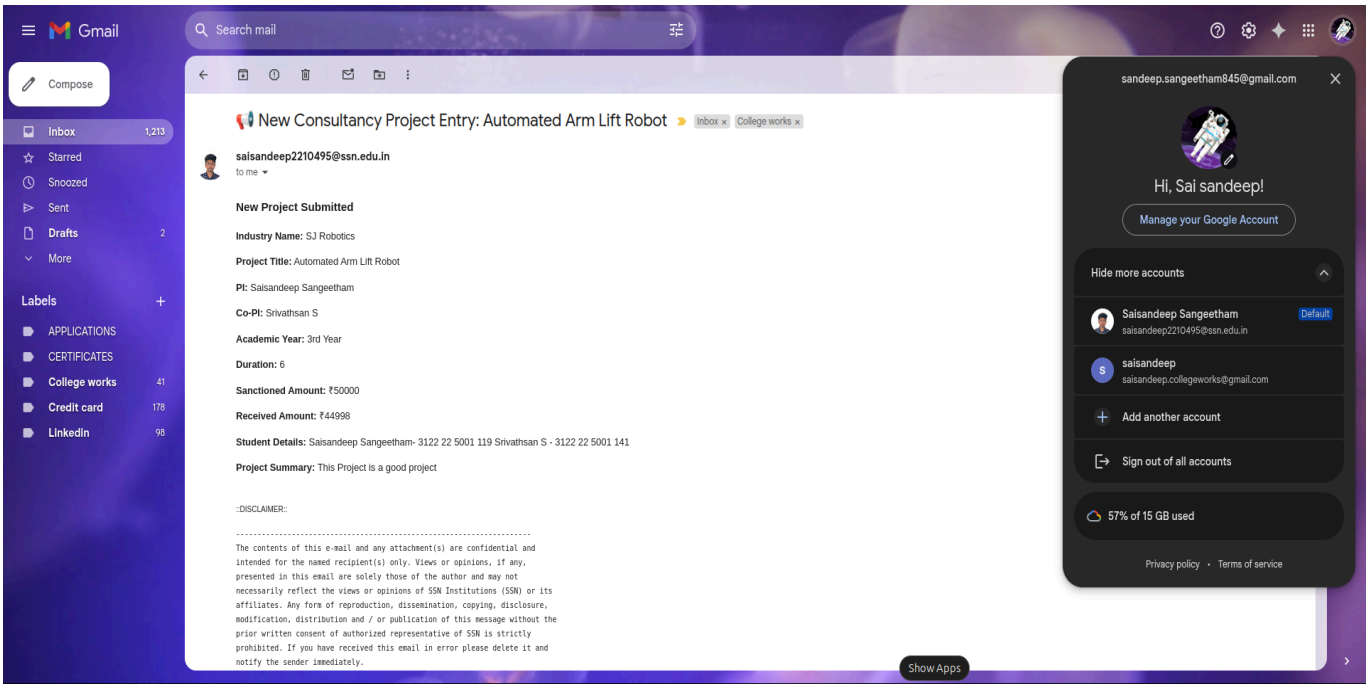

←Back
Forgot Password
Enter the verification code sent to
sandeep.sangeetham845@gmail.com
Enter OTP *
543426
VERIFY OTP
Didn't receive the code? RESEND OTP

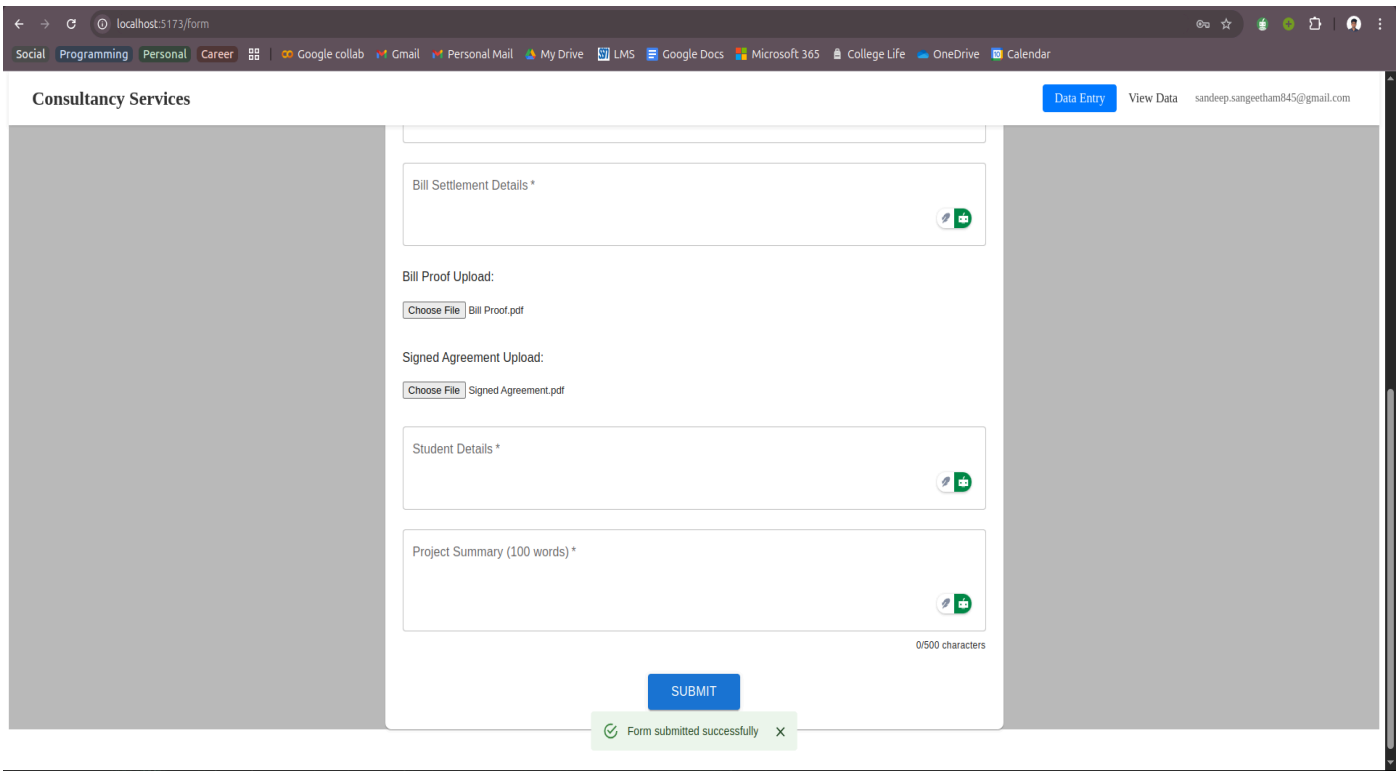

Reset Password
sandeep.sangeetham845@gmail.com
Update Password
Remember your password? Login

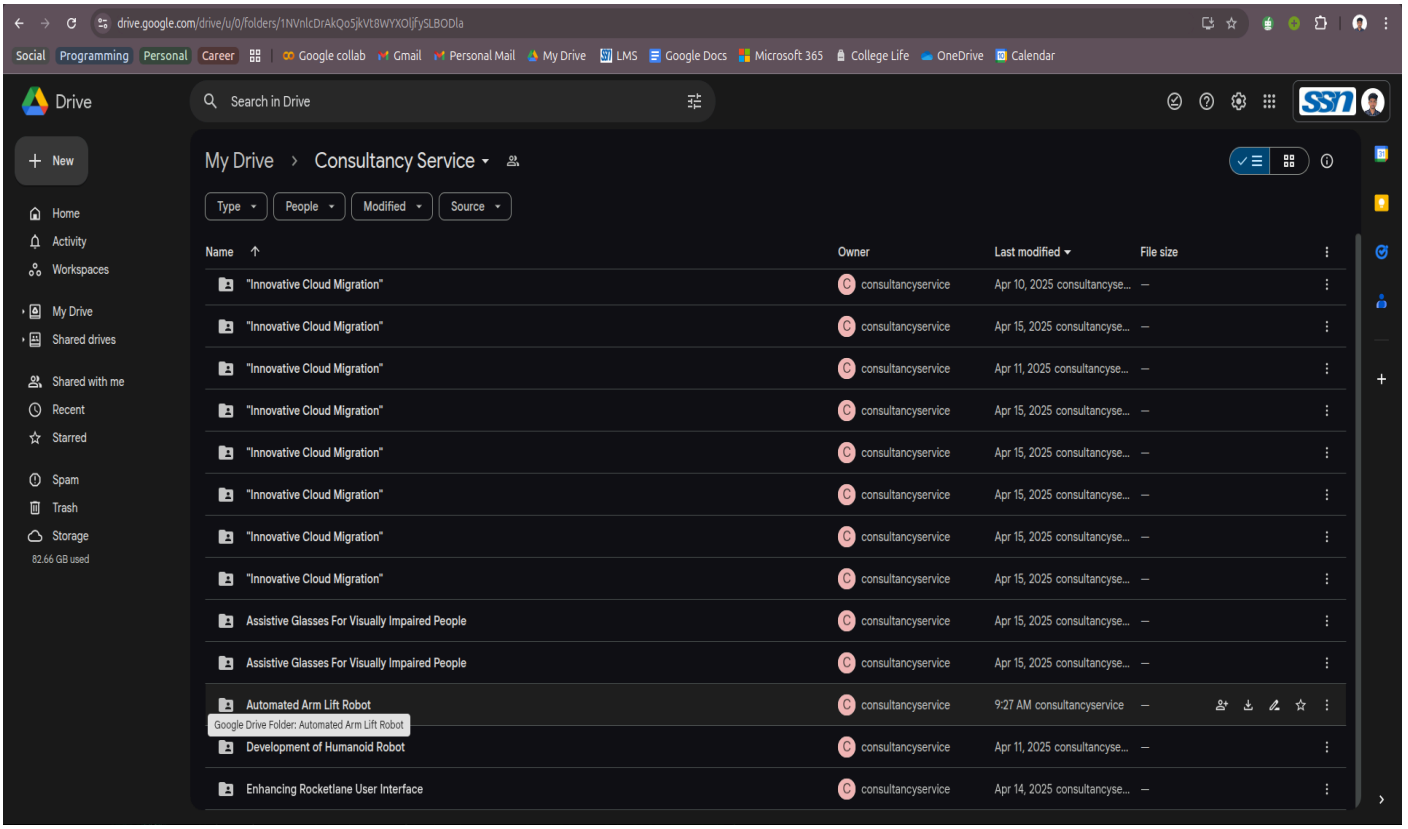

localhost:5173/form
Social
Programming
Personal
Career
Google collab
Gmail
Personal Mail
My Drive
LMS
Google Docs
Microsoft 365
College Life
OneDrive
Calendar
Consultancy Services
Data Entry
View Data
sandeep.sangeetham845@gmail.com
Consultancy Project Details
Industry Name *
SJ Robotics
Project Duration (in months) *
6
months
Project Title *
Automated Arm Lift Robot
Principal Investigator *
Saisandeep Sangeetham
Co-Principal Investigator
Srivathsan S
Academic Year *
3rd Year
Amount Sanctioned (₹) *
₹ 50000
Amount Received (₹) *
₹ 44998
Bill Settlement Details *

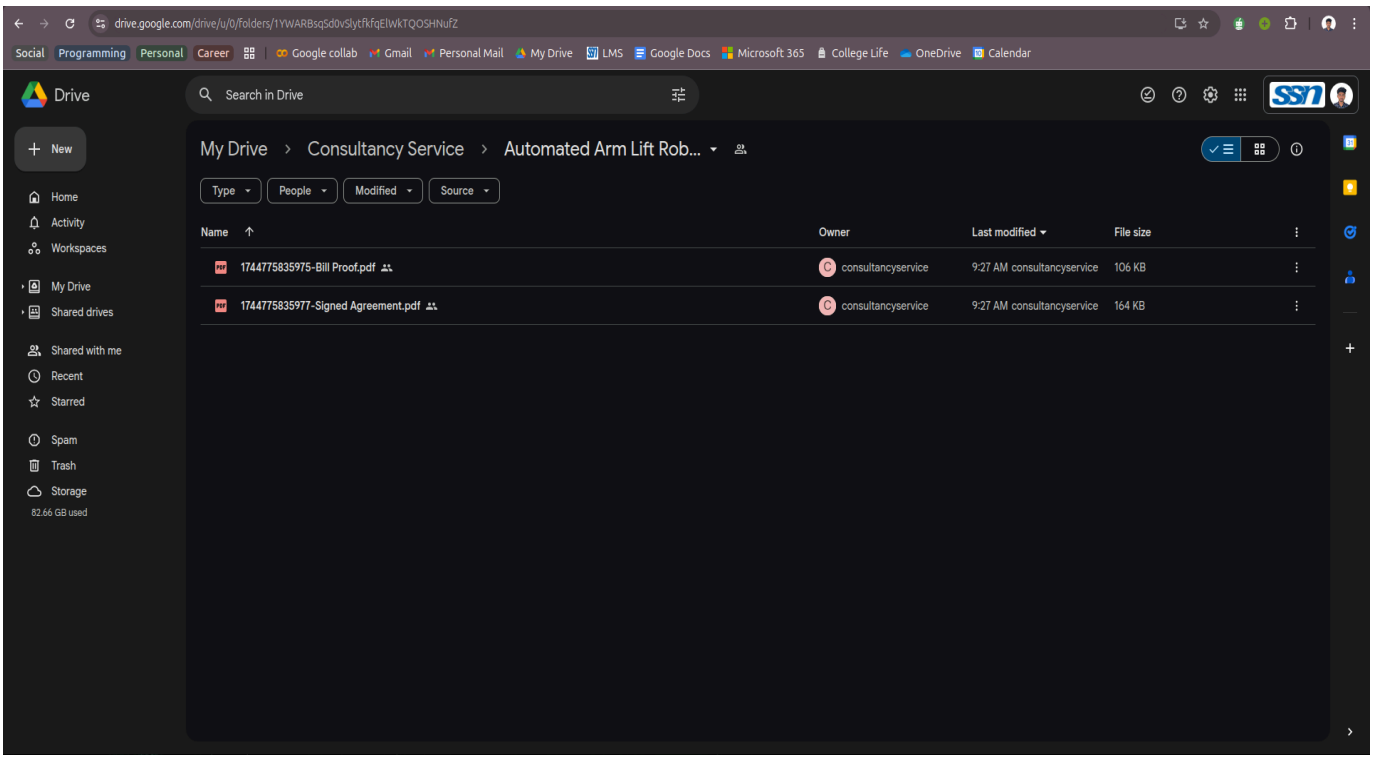

localhost:5173/form
Consultancy Services
Data Entry
View Data
sandeep.sangeetham845@gmail.com
₹ 44998
Bill Settlement Details *
Invoice for the bill settlement- (10-04-2025)
Bill Proof Upload:
Choose File Bill Proof.pdf
Selected file: Bill Proof.pdf
Signed Agreement Upload:
Choose File Signed Agreement.pdf
Selected file: Signed Agreement.pdf
Student Details *
Saisandeep Sangeetham- 3122 22 5001 119
Srivathsan S - 3122 22 5001 141
Project Summary (100 words) *
This Project is a good project
30/500 characters
SUBMIT

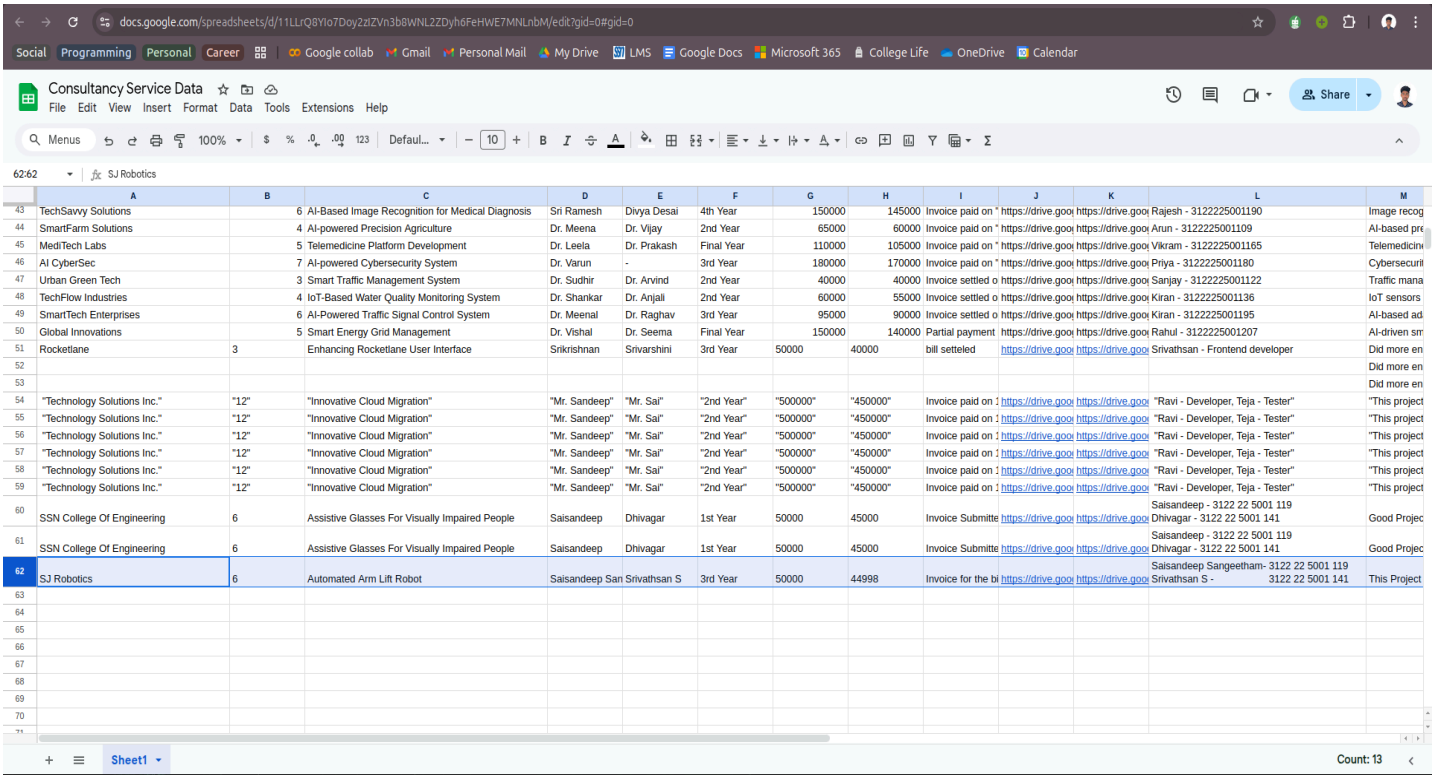

Gmail
Search mail
Compose
Inbox 1,214
Starred
Primary
Promotions 60 new
Social 18 new
Updates 60 new
saisandeep2210495
New Consultancy Project Entry: Automated Arm Lift Robot - New Project Submitted Industry Name: SJ Robotics Project Title: Automated Arm Lift Robot PI: Saisan...
09:27


Gmail
Search mail
Compose
Inbox 1,213
Starred
Snoozed
Sent
Drafts 2
More
Labels
APPLICATIONS
CERTIFICATES
College works 41
Credit card 178
Linkedin 98
New Consultancy Project Entry: Automated Arm Lift Robot
saisandeep2210495@ssn.edu.in
to me
New Project Submitted
Industry Name: SJ Robotics
Project Title: Automated Arm Lift Robot
PI: Saisandeep Sangeetham
Co-PI: Srivathsan S
Academic Year: 3rd Year
Duration: 6
Sanctioned Amount: ₹50000
Received Amount: ₹44998
Student Details: Saisandeep Sangeetham- 3122 22 5001 119 Srivathsan S - 3122 22 5001 141
Project Summary: This Project is a good project
::DISCLAIMER::
The contents of this e-mail and any attachment(s) are confidential and intended for the named recipient(s) only. Views or opinions, if any, presented in this email are solely those of the author and may not necessarily reflect the views or opinions of SSN Institutions (SSN) or its affiliates. Any form of reproduction, dissemination, copying, disclosure, modification, distribution and / or publication of this message without the prior written consent of authorized representative of SSN is strictly prohibited. If you have received this email in error please delete it and notify the sender immediately.
sandeep.sangeetham845@gmail.com
Hi, Sai sandeep!
Manage your Google Account
Hide more accounts
Saisandeep Sangeetham
saisandeep2210495@ssn.edu.in
Default
saisandeep
saisandeep.collegeworks@gmail.com
Add another account
Sign out of all accounts
57% of 15 GB used
Privacy policy • Terms of service
Show Apps

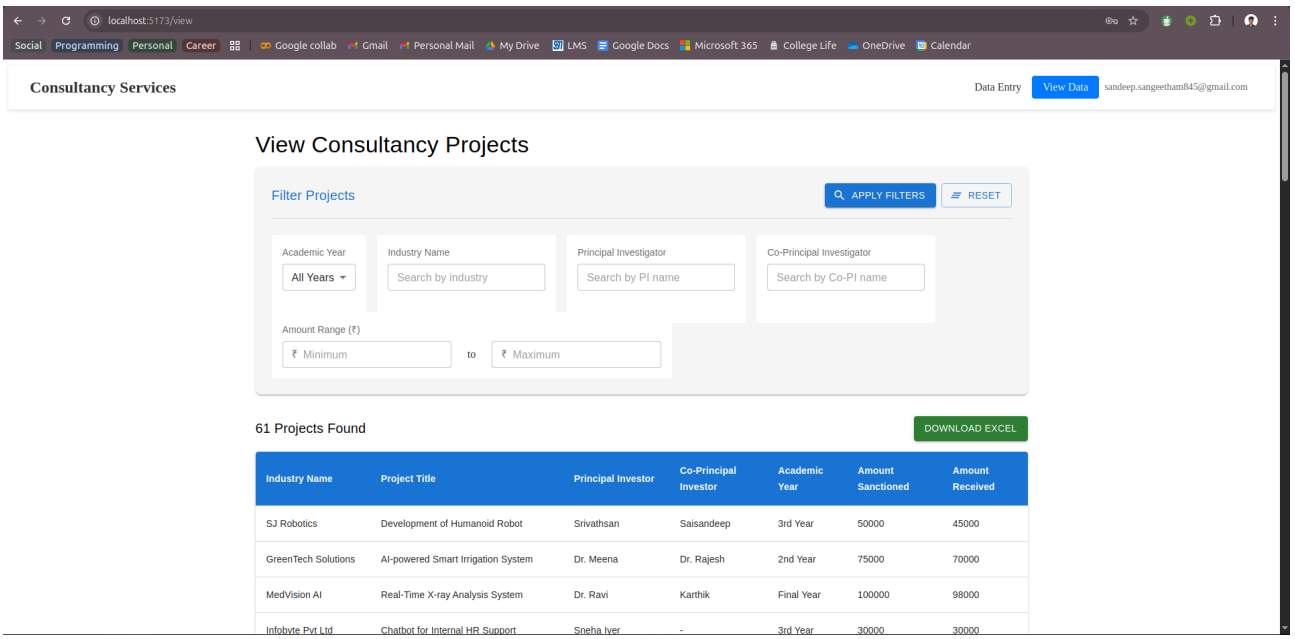

localhost:5173/form
Consultancy Services
Data Entry
View Data
sandeep.sangeetham845@gmail.com
Bill Settlement Details *
Bill Proof Upload:
Choose File
Bill Proof.pdf
Signed Agreement Upload:
Choose File
Signed Agreement.pdf
Student Details *
Project Summary (100 words) *
0/500 characters
SUBMIT
Form submitted successfully

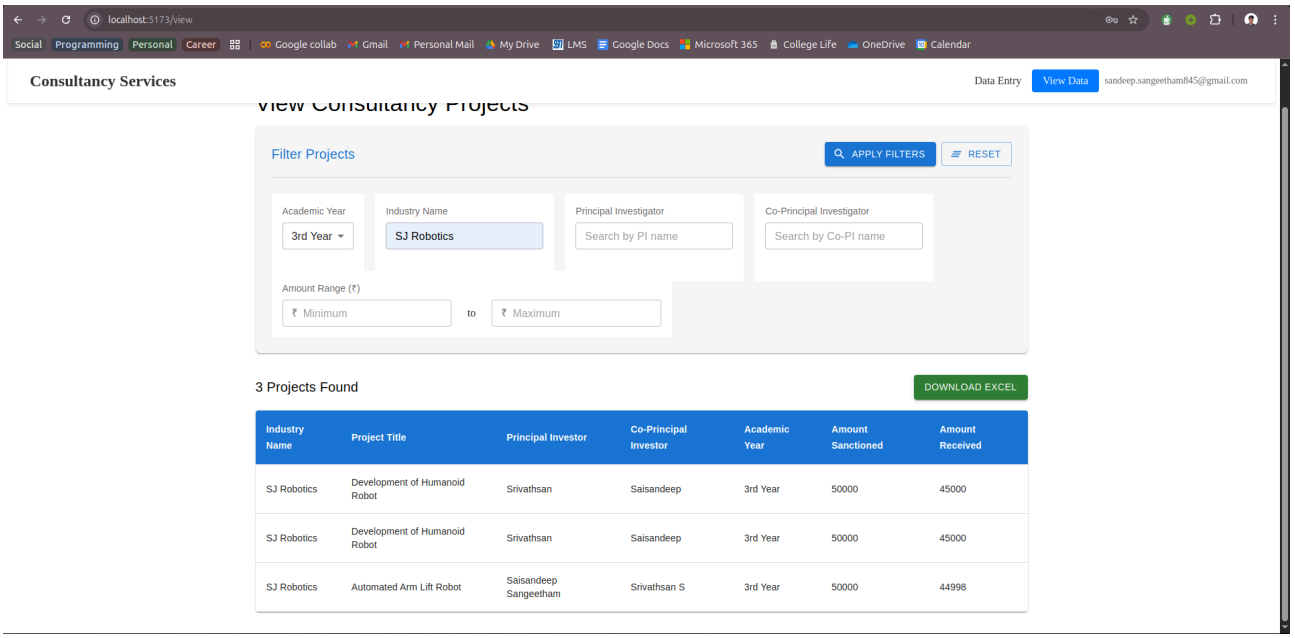

drive.google.com/drive/u/0/folders/1NVnlcDrAkQo5jkVt8WYXOljfySLBODla
Drive
Search in Drive
New
My Drive
Consultancy Service
Type
People
Modified
Source
Name
Owner
Last modified
File size
"Innovative Cloud Migration"
Assistive Glasses For Visually Impaired People
Automated Arm Lift Robot
Google Drive Folder: Automated Arm Lift Robot
Development of Humanoid Robot
Enhancing Rocketlane User Interface
consultancyservice

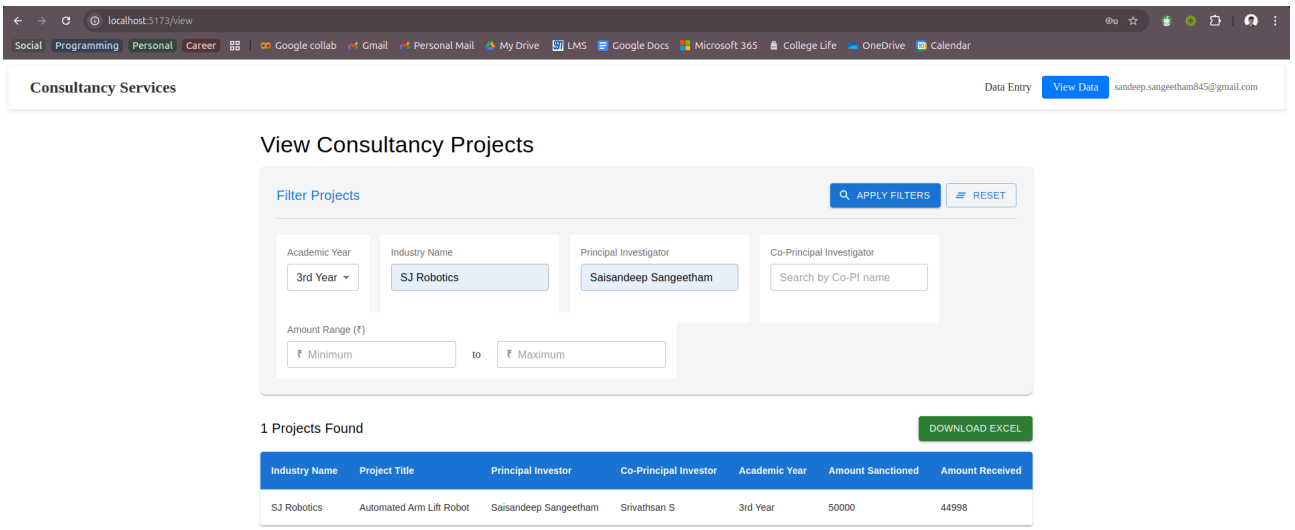


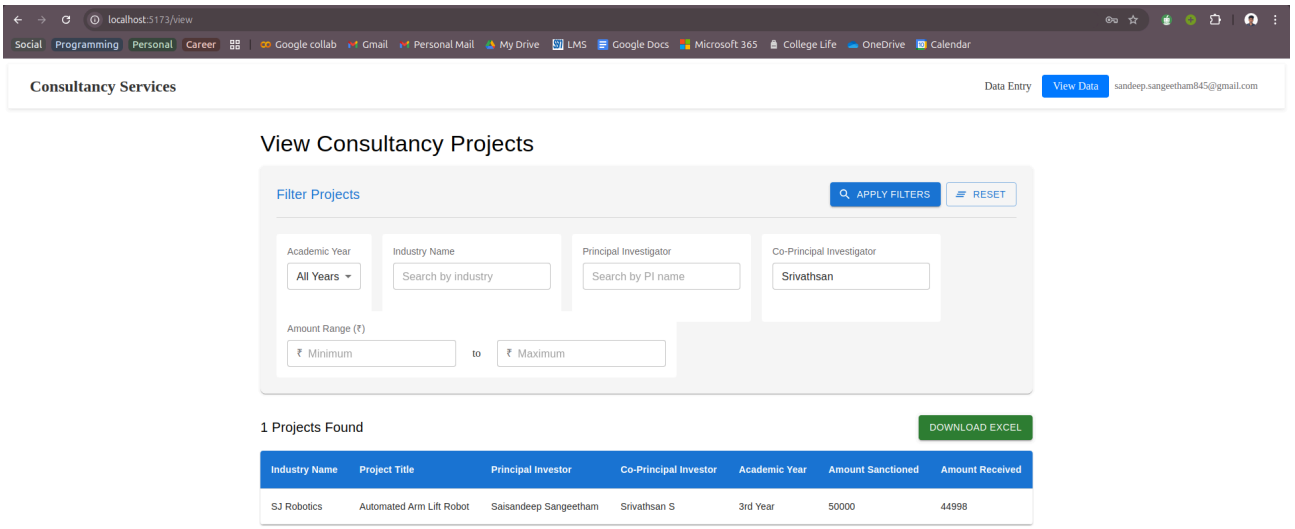


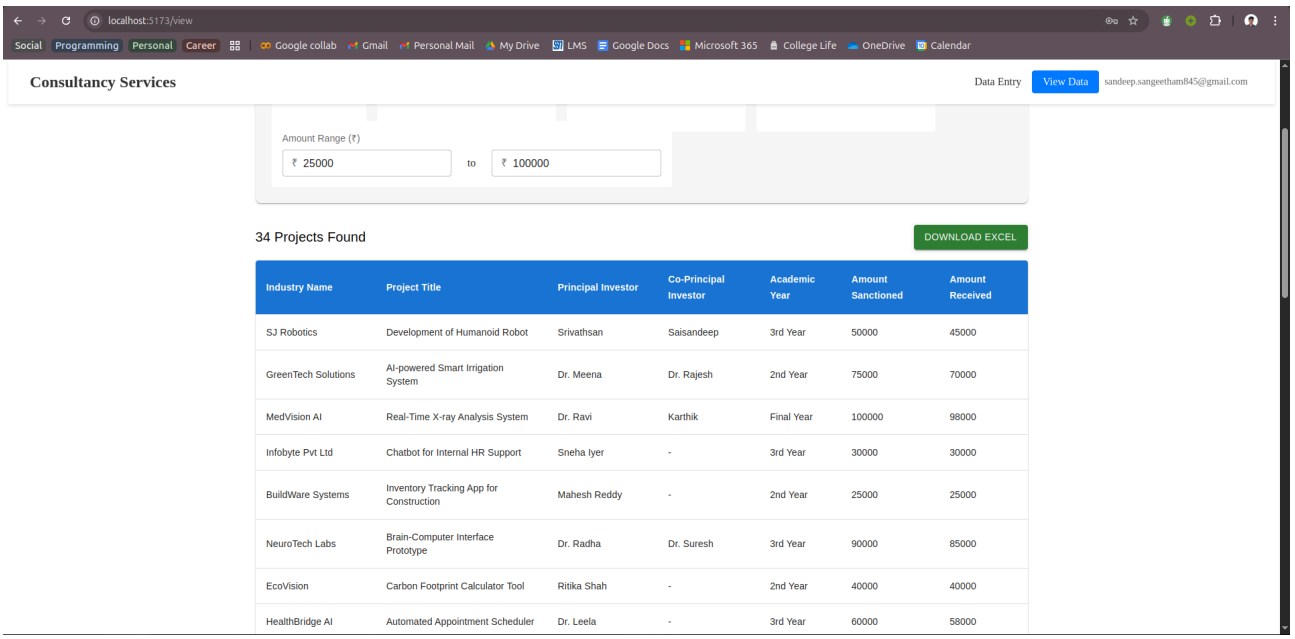
	A	B	C	D	E	F	G	H	I	J	K	L	M
43	TechSavvy Solutions	6	AI-Based Image Recognition for Medical Diagnosis	Sri Ramesh	Divya Desai	4th Year	150000	145000	Invoice paid on "	https://drive.goo	https://drive.goo	Rajesh - 3122225001190	Image recog
44	SmartFarm Solutions	4	AI-powered Precision Agriculture	Dr. Meena	Dr. Vijay	2nd Year	65000	60000	Invoice paid on "	https://drive.goo	https://drive.goo	Arun - 3122225001109	AI-based pre
45	MediTech Labs	5	Telemedicine Platform Development	Dr. Leela	Dr. Prakash	Final Year	110000	105000	Invoice paid on "	https://drive.goo	https://drive.goo	Vikram - 3122225001165	Telemedicin
46	AI CyberSec	7	AI-powered Cybersecurity System	Dr. Varun	-	3rd Year	180000	170000	Invoice paid on "	https://drive.goo	https://drive.goo	Priya - 3122225001180	Cybersecurit
47	Urban Green Tech	3	Smart Traffic Management System	Dr. Sudhir	Dr. Arvind	2nd Year	40000	40000	Invoice settled o	https://drive.goo	https://drive.goo	Sanjay - 3122225001122	Traffic mana
48	TechFlow Industries	4	IoT-Based Water Quality Monitoring System	Dr. Shankar	Dr. Anjali	2nd Year	60000	55000	Invoice settled o	https://drive.goo	https://drive.goo	Kiran - 3122225001136	IoT sensors
49	SmartTech Enterprises	6	AI-Powered Traffic Signal Control System	Dr. Meenal	Dr. Raghav	3rd Year	95000	90000	Invoice settled o	https://drive.goo	https://drive.goo	Kiran - 3122225001195	AI-based ad
50	Global Innovations	5	Smart Energy Grid Management	Dr. Vishal	Dr. Seema	Final Year	150000	140000	Partial payment	https://drive.goo	https://drive.goo	Rahul - 3122225001207	AI-driven sm
51	Rocketlane	3	Enhancing Rocketlane User Interface	Srikrishnan	Srivarshini	3rd Year	50000	40000	bill setteled	https://drive.goo	https://drive.goo	Srivathsan - Frontend developer	Did more en
52													Did more en
53													Did more en
54	"Technology Solutions Inc."	"12"	"Innovative Cloud Migration"	"Mr. Sandeep"	"Mr. Sai"	"2nd Year"	"500000"	"450000"	Invoice paid on !	https://drive.goo	https://drive.goo	"Ravi - Developer, Teja - Tester"	"This project
55	"Technology Solutions Inc."	"12"	"Innovative Cloud Migration"	"Mr. Sandeep"	"Mr. Sai"	"2nd Year"	"500000"	"450000"	Invoice paid on !	https://drive.goo	https://drive.goo	"Ravi - Developer, Teja - Tester"	"This project
56	"Technology Solutions Inc."	"12"	"Innovative Cloud Migration"	"Mr. Sandeep"	"Mr. Sai"	"2nd Year"	"500000"	"450000"	Invoice paid on !	https://drive.goo	https://drive.goo	"Ravi - Developer, Teja - Tester"	"This project
57	"Technology Solutions Inc."	"12"	"Innovative Cloud Migration"	"Mr. Sandeep"	"Mr. Sai"	"2nd Year"	"500000"	"450000"	Invoice paid on !	https://drive.goo	https://drive.goo	"Ravi - Developer, Teja - Tester"	"This project
58	"Technology Solutions Inc."	"12"	"Innovative Cloud Migration"	"Mr. Sandeep"	"Mr. Sai"	"2nd Year"	"500000"	"450000"	Invoice paid on !	https://drive.goo	https://drive.goo	"Ravi - Developer, Teja - Tester"	"This project
59	"Technology Solutions Inc."	"12"	"Innovative Cloud Migration"	"Mr. Sandeep"	"Mr. Sai"	"2nd Year"	"500000"	"450000"	Invoice paid on !	https://drive.goo	https://drive.goo	"Ravi - Developer, Teja - Tester"	"This project
60	SSN College Of Engineering	6	Assistive Glasses For Visually Impaired People	Saisandeep	Dhivagar	1st Year	50000	45000	Invoice Submitte	https://drive.goo	https://drive.goo	Saisandeep - 3122 22 5001 119 Dhivagar - 3122 22 5001 141	Good Projec
61	SSN College Of Engineering	6	Assistive Glasses For Visually Impaired People	Saisandeep	Dhivagar	1st Year	50000	45000	Invoice Submitte	https://drive.goo	https://drive.goo	Saisandeep - 3122 22 5001 119 Dhivagar - 3122 22 5001 141	Good Projec
62	SJ Robotics	6	Automated Arm Lift Robot	Saisandeep San	Srivathsan S	3rd Year	50000	44998	Invoice for the bi	https://drive.goo	https://drive.goo	Saisandeep Sangeetham- 3122 22 5001 119 Srivathsan S - 3122 22 5001 141	This Project
63													
64													
65													
66													
67													
68													
69													
70													


localhost:5173/view
Social
Programming
Personal
Career
Google collab
Gmail
Personal Mail
My Drive
LMS
Google Docs
Microsoft 365
College Life
OneDrive
Calendar
Consultancy Services
Data Entry
View Data
sandeep.sangeetham845@gmail.com
View Consultancy Projects
Filter Projects
APPLY FILTERS
RESET
Academic Year
All Years
Industry Name
Search by industry
Principal Investigator
Search by PI name
Co-Principal Investigator
Search by Co-PI name
Amount Range (₹)
₹ Minimum
to
₹ Maximum
61 Projects Found
DOWNLOAD EXCEL
Industry Name
Project Title
Principal Investor
Co-Principal Investor
Academic Year
Amount Sanctioned
Amount Received
SJ Robotics
Development of Humanoid Robot
Srivathsan
Saisandeep
3rd Year
50000
45000
GreenTech Solutions
AI-powered Smart Irrigation System
Dr. Meena
Dr. Rajesh
2nd Year
75000
70000
MedVision AI
Real-Time X-ray Analysis System
Dr. Ravi
Karthik
Final Year
100000
98000
Infobyte Pvt Ltd
Chatbot for Internal HR Support
Sneha Iyer
-
3rd Year
30000
30000

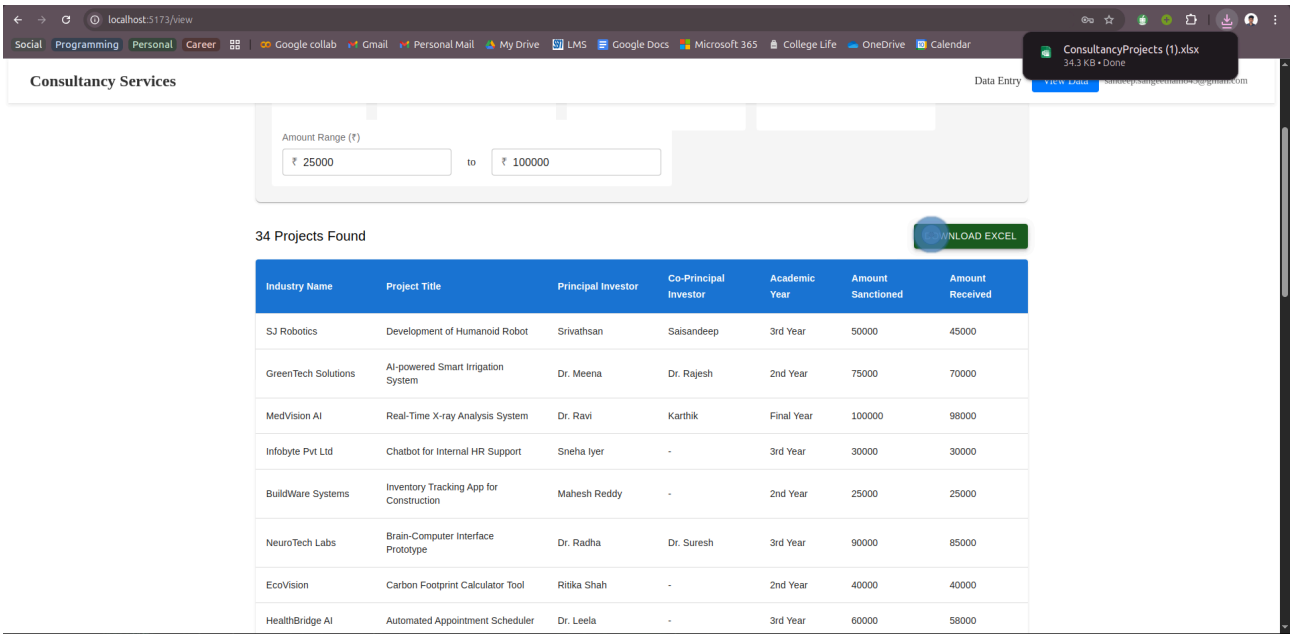

localhost:5173/view
Social
Programming
Personal
Career
Google collab
Gmail
Personal Mail
My Drive
LMS
Google Docs
Microsoft 365
College Life
OneDrive
Calendar
Consultancy Services
Data Entry
View Data
sandeep.sangeetham845@gmail.com
Filter Projects
APPLY FILTERS
RESET
Academic Year
3rd Year
Industry Name
SJ Robotics
Principal Investigator
Search by PI name
Co-Principal Investigator
Search by Co-PI name
Amount Range (₹)
₹ Minimum
to
₹ Maximum
3 Projects Found
DOWNLOAD EXCEL
Industry Name
Project Title
Principal Investor
Co-Principal Investor
Academic Year
Amount Sanctioned
Amount Received
SJ Robotics
Development of Humanoid Robot
Srivathsan
Saisandeep
3rd Year
50000
45000
SJ Robotics
Development of Humanoid Robot
Srivathsan
Saisandeep
3rd Year
50000
45000
SJ Robotics
Automated Arm Lift Robot
Saisandeep Sangeetham
Srivathsan S
3rd Year
50000
44998


localhost:5173/view
Social
Programming
Personal
Career
Google collab
Gmail
Personal Mail
My Drive
LMS
Google Docs
Microsoft 365
College Life
OneDrive
Calendar
Consultancy Services
Data Entry
View Data
sandeep.sangeetham845@gmail.com
View Consultancy Projects
Filter Projects
APPLY FILTERS
RESET
Academic Year
3rd Year
Industry Name
SJ Robotics
Principal Investigator
Saisandeep Sangeetham
Co-Principal Investigator
Search by Co-PI name
Amount Range (₹)
₹ Minimum
to
₹ Maximum
1 Projects Found
DOWNLOAD EXCEL
Industry Name
Project Title
Principal Investor
Co-Principal Investor
Academic Year
Amount Sanctioned
Amount Received
SJ Robotics
Automated Arm Lift Robot
Saisandeep Sangeetham
Srivathsan S
3rd Year
50000
44998

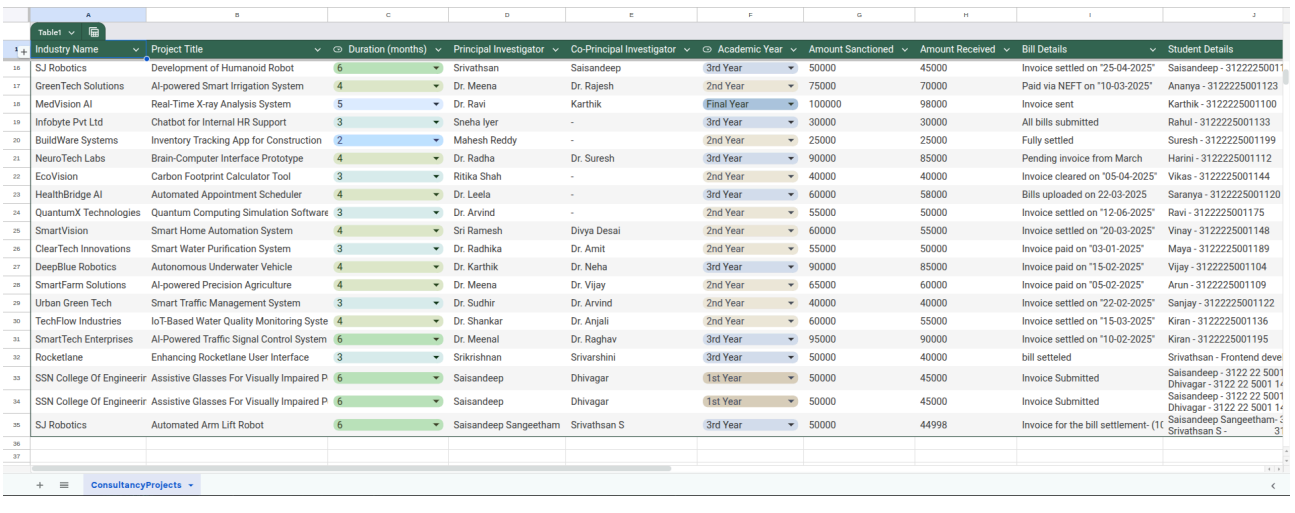

localhost:5173/view
Social
Programming
Personal
Career
Google collab
Gmail
Personal Mail
My Drive
LMS
Google Docs
Microsoft 365
College Life
OneDrive
Calendar
Consultancy Services
Data Entry
View Data
sandeep.sangeetham845@gmail.com
View Consultancy Projects
Filter Projects
APPLY FILTERS
RESET
Academic Year
All Years
Industry Name
Search by industry
Principal Investigator
Search by PI name
Co-Principal Investigator
Srivathsan
Amount Range (₹)
₹ Minimum
to
₹ Maximum
1 Projects Found
DOWNLOAD EXCEL
Industry Name
Project Title
Principal Investor
Co-Principal Investor
Academic Year
Amount Sanctioned
Amount Received
SJ Robotics
Automated Arm Lift Robot
Saisandeep Sangeetham
Srivathsan S
3rd Year
50000
44998



Industry Name	Project Title	Principal Investor	Co-Principal Investor	Academic Year	Amount Sanctioned	Amount Received
SJ Robotics	Development of Humanoid Robot	Srivathsan	Saisandeep	3rd Year	50000	45000
GreenTech Solutions	AI-powered Smart Irrigation System	Dr. Meena	Dr. Rajesh	2nd Year	75000	70000
MedVision AI	Real-Time X-ray Analysis System	Dr. Ravi	Karthik	Final Year	100000	98000
Infobyte Pvt Ltd	Chatbot for Internal HR Support	Sneha Iyer	-	3rd Year	30000	30000
BuildWare Systems	Inventory Tracking App for Construction	Mahesh Reddy	-	2nd Year	25000	25000
NeuroTech Labs	Brain-Computer Interface Prototype	Dr. Radha	Dr. Suresh	3rd Year	90000	85000
EcoVision	Carbon Footprint Calculator Tool	Ritika Shah	-	2nd Year	40000	40000
HealthBridge AI	Automated Appointment Scheduler	Dr. Leela	-	3rd Year	60000	58000



Industry Name	Project Title	Principal Investor	Co-Principal Investor	Academic Year	Amount Sanctioned	Amount Received
SJ Robotics	Development of Humanoid Robot	Srivathsan	Saisandeep	3rd Year	50000	45000
GreenTech Solutions	AI-powered Smart Irrigation System	Dr. Meena	Dr. Rajesh	2nd Year	75000	70000
MedVision AI	Real-Time X-ray Analysis System	Dr. Ravi	Karthik	Final Year	100000	98000
Infobyte Pvt Ltd	Chatbot for Internal HR Support	Sneha Iyer	-	3rd Year	30000	30000
BuildWare Systems	Inventory Tracking App for Construction	Mahesh Reddy	-	2nd Year	25000	25000
NeuroTech Labs	Brain-Computer Interface Prototype	Dr. Radha	Dr. Suresh	3rd Year	90000	85000
EcoVision	Carbon Footprint Calculator Tool	Ritika Shah	-	2nd Year	40000	40000
HealthBridge AI	Automated Appointment Scheduler	Dr. Leela	-	3rd Year	60000	58000



Industry Name	Project Title	Duration (months)	Principal Investigator	Co-Principal Investigator	Academic Year	Amount Sanctioned	Amount Received	Bill Details	Student Details
SJ Robotics	Development of Humanoid Robot	6	Srivathsan	Saisandeep	3rd Year	50000	45000	Invoice settled on "25-04-2025"	Saisandeep - 3122225001
GreenTech Solutions	AI-powered Smart Irrigation System	4	Dr. Meena	Dr. Rajesh	2nd Year	75000	70000	Paid via NEFT on "10-03-2025"	Ananya - 3122225001123
MedVision AI	Real-Time X-ray Analysis System	5	Dr. Ravi	Karthik	Final Year	100000	98000	Invoice sent	Karthik - 3122225001100
Infobyte Pvt Ltd	Chatbot for Internal HR Support	3	Sneha Iyer	-	3rd Year	30000	30000	All bills submitted	Rahul - 3122225001133
BuildWare Systems	Inventory Tracking App for Construction	2	Mahesh Reddy	-	2nd Year	25000	25000	Fully settled	Suresh - 3122225001199
NeuroTech Labs	Brain-Computer Interface Prototype	4	Dr. Radha	Dr. Suresh	3rd Year	90000	85000	Pending invoice from March	Harini - 3122225001112
EcoVision	Carbon Footprint Calculator Tool	3	Ritika Shah	-	2nd Year	40000	40000	Invoice cleared on "05-04-2025"	Vikas - 3122225001144
HealthBridge AI	Automated Appointment Scheduler	4	Dr. Leela	-	3rd Year	60000	58000	Bills uploaded on 22-03-2025	Saranya - 31222250011201
QuantumX Technologies	Quantum Computing Simulation Software	3	Dr. Arvind	-	2nd Year	55000	50000	Invoice settled on "12-06-2025"	Ravi - 3122225001175
SmartVision	Smart Home Automation System	4	Sri Ramesh	Divya Desai	2nd Year	60000	55000	Invoice settled on "20-03-2025"	Vinay - 3122225001148
ClearTech Innovations	Smart Water Purification System	3	Dr. Radhika	Dr. Amit	2nd Year	55000	50000	Invoice paid on "03-01-2025"	Maya - 3122225001189
DeepBlue Robotics	Autonomous Underwater Vehicle	4	Dr. Karthik	Dr. Neha	3rd Year	90000	85000	Invoice paid on "15-02-2025"	Vijay - 3122225001104
SmartFarm Solutions	AI-powered Precision Agriculture	4	Dr. Meena	Dr. Vijay	2nd Year	65000	60000	Invoice paid on "05-02-2025"	Arun - 3122225001109
Urban Green Tech	Smart Traffic Management System	3	Dr. Sudhir	Dr. Arvind	2nd Year	40000	40000	Invoice settled on "22-02-2025"	Sanjay - 3122225001122
TechFlow Industries	IoT-Based Water Quality Monitoring Syste	4	Dr. Shankar	Dr. Anjali	2nd Year	60000	55000	Invoice settled on "15-03-2025"	Kiran - 3122225001136
SmartTech Enterprises	AI-Powered Traffic Signal Control System	6	Dr. Meenal	Dr. Raghav	3rd Year	95000	90000	Invoice settled on "10-02-2025"	Kiran - 3122225001195
Rocketlane	Enhancing Rocketlane User Interface	3	Srikrishnan	Srivarshini	3rd Year	50000	40000	bill setteled	Srivathsan - Frontend devel
SSN College Of Engineerir	Assistive Glasses For Visually Impaired P	6	Saisandeep	Dhivagar	1st Year	50000	45000	Invoice Submitted	Saisandeep - 3122 22 5001 Dhivagar - 3122 22 5001 14
SSN College Of Engineerir	Assistive Glasses For Visually Impaired P	6	Saisandeep	Dhivagar	1st Year	50000	45000	Invoice Submitted	Saisandeep - 3122 22 5001 Dhivagar - 3122 22 5001 14
SJ Robotics	Automated Arm Lift Robot	6	Saisandeep Sangeetham	Srivathsan S	3rd Year	50000	44998	Invoice for the bill settlement- (1(	Saisandeep Sangeetham - 3 Srivathsan S - 31

Learning Outcome:

By the end of this project, we learnt:

Design and develop a full-stack web application using React.js and Node.js with Express, following modern development practices.

Implement user authentication and security features using bcryptjs and JWT, while selectively integrating MongoDB for secure credential management.

Perform CRUD operations on Excel files using the xlsx and file-saver libraries, enabling structured data handling without a traditional database.

Integrate file upload functionalities using Multer and manage file system operations through Node.js core modules.

Develop and style responsive UIs using Material UI, Emotion, and MUI Icons, promoting consistent and modern design principles.

Utilize routing and state management in a React application using react-router-dom and appropriate component state techniques.

Facilitate user interaction with data through advanced filtering, searching, and Excel download features.

Automate communication workflows via email notifications using the Gmail API through the googleapis library.

Apply middleware for enhanced backend functionality, including logging (morgan), body parsing, and CORS configuration.