WEB TECH PROJECT
Driving Licence Management

Team: 9

Section: J

Name: Shreya Kundu

SRN: PES1UG23CS918

Name: Sripriya Srinivas

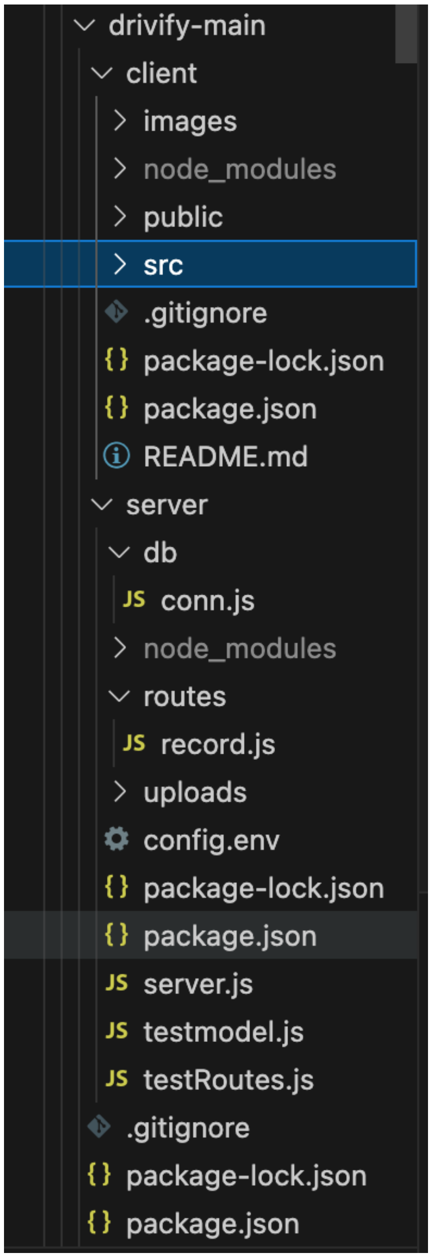
SRN: PES1UG23CS597

Name: Surabhi

SRN: PES1UG23CS616

Client 5
src 5
api.js 5
App.css 6
App.js 7
App.test.js 8
ApplicationDL.js 8
ApplicationForm.js 14
BookAppointment.js 19
ConfirmDL.js 22
form_styles.css 24
header_col.css 25
Home.css 26
Home.js 26
index.css 29
index.js 29
Instructions.js 30
Payment.js 31
PrintApplication.js 32
PrintDL.js 34
Question.js 38
QuestionBubble.js 38
Quiz.css 39
Quiz.js 42
reportWebVitals.js 46
SelectState.css 46
SelectState.js 49
SetupTests.js 53

Submission.js 53
Test.css 53
Test.js 55
Timer.js 56
Server 57
db 57
conn.js 57
Routes 58
record.js 58
server.js 62
testmodel.js 63
testRoutes.js 64
Screenshots of the output 65



```
drivify-main
  client
    images
    node_modules
    public
    src
    .gitignore
    package-lock.json
    package.json
    README.md
  server
    db
      conn.js
    node_modules
    routes
      record.js
    uploads
    config.env
    package-lock.json
    package.json
    server.js
    testmodel.js
    testRoutes.js
  .gitignore
  package-lock.json
  package.json
```

The screenshot of the directories inside the root-directory

The code for the project:

src folder of client:

Client

src

api.js

```
export const saveApplication = async (formData) => {
  try {
    const response = await fetch("http://localhost:5000/api/save-application", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify(formData),
    });

    if (!response.ok) {
      throw new Error('Network response was not ok');
    }

    return await response.json(); // Return the JSON response
  } catch (error) {
    console.error('Error saving application:', error);
    throw error; // Rethrow error for handling in the caller
  }
};

export const getApplication = async (applicationId) => {
  try {
    const response = await fetch("http://localhost:5000/api/get-
application?applicationId="+applicationId, {
      method: "GET",
      headers: {
        "Content-Type": "application/json",
      },
    });

    if (!response.ok) {
      throw new Error('Network response was not ok');
    }

    return await response.json(); // Return the JSON response
  } catch (error) {
    console.error('Error getting application:', error);
    throw error; // Rethrow error for handling in the caller
  }
};
```

App.css

```
.App {
  text-align: center;
}

.App-logo {
  height: 40vmin;
  pointer-events: none;
}

@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
  }
}

.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}
```

```
.App-link {
  color: #61dafb;
}

@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}
```

App.js

```
import React from 'react';
import SelectState from './SelectState';
import { BrowserRouter as Router, Route, Routes, Navigate } from "react-router-dom";
import Home from "./Home";
import Quiz from "./Quiz"
import Instructions from "./Instructions";
import Apply from "./ApplicationForm";
import Payment from "./Payment";
import ApplicationDL from './ApplicationDL';
import ConfirmDL from './ConfirmDL';
import Print from "./PrintApplication"
import BookAppointment from './BookAppointment';
import PrintDL from './PrintDL';

const App = () => {
    return (
    <Router>
      <Routes>
        <Route path="/" element={<SelectState />} />
        <Route path="/home" element={<Home />} />
        <Route path="/apply" element={<Apply />} />
        <Route path="/instructions" element={<Instructions />} />
        <Route path="/payment" element={<Payment />} />
        <Route path="/quiz" element={<Quiz/>}/>
        <Route path="*" element={<Navigate to="/" replace />} />
        <Route path="/ApplicationDL" element={<ApplicationDL />} />
        <Route path="/ConfirmDL" element={<ConfirmDL />} />
        <Route path="/print" element={<Print />}/>
        <Route path="/bookdl" element={<BookAppointment />}/>
        <Route path="/printdl" element={<PrintDL />}/>
      </Routes>
    </Router>
    );
};

export default App;
```

App.test.js

```
import { render, screen } from '@testing-library/react';
import App from './App';

test('renders learn react link', () => {
  render(<App />);
  const linkElement = screen.getByText(/learn react/i);
  expect(linkElement).toBeInTheDocument();
});
```

ApplicationDL.js

```
import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import validator from 'validator';
import Tesseract from 'tesseract.js';
import './form_styles.css';
import './header_col.css';

const Form = () => {
    const navigate = useNavigate();
    const [formData, setFormData] = useState({
        fullName: '',
        dob: '',
        address: '',
        aadhaarfile: null,
        aadhaar: '',
        city: '',
        pincode: '',
        email: '',
        phno: '',
        learnersLicense: '',
        bloodGroup: 'O',
        nationality: 'Indian',
        vehicleType: 'twoWheeler',
    });

    const [errors, setErrors] = useState({});
    const [extractedText, setExtractedText] = useState('');

    const handleInputChange = (e) => {
        const { name, value } = e.target;
        setFormData({ ...formData, [name]: value });
    };

    const handleFileChange = (e) => {
        const { name, files } = e.target;
        setFormData({ ...formData, [name]: files[0] });
        extractText(files[0]);
    };

    //validate the form details
    const validateForm = () => {
        const newErrors = {};

        if (!formData.fullName) newErrors.fullName = "Full Name is required";
        if (!validator.isDate(formData.dob)) newErrors.dob = "Invalid date of birth";
        if (!formData.address) newErrors.address = "Address is required";
        if (!validator.isLength(formData.aadhaar, { min: 12, max: 12 })) newErrors.aadhaar = 
"Aadhaar must be 12 digits";
        if (!validator.isLength(formData.pincode, { min: 6, max: 6 })) newErrors.pincode = 
"Pincode must be 6 digits";
        if (!validator.isEmail(formData.email)) newErrors.email = "Invalid email address";
        if (!validator.isLength(formData.phno, { min: 10, max: 10 })) newErrors.phno = "Phone 
number must be 10 digits";

        setErrors(newErrors);
        return Object.keys(newErrors).length === 0;
    };

    // Function to extract text using Tesseract
    const extractText = (file) => {
        if (!file) return;

        Tesseract.recognize(
          file,
          'eng',
          {
            logger: (m) => console.log(m),
          }
        ).then(({ data: { text } }) => {
          setExtractedText(text);
          extractAadhaarNumber(text);
        });
    };

    // Function to extract Aadhaar number using regex
    const extractAadhaarNumber = (text) => {
        const aadhaarPattern = /\b\d{4}\s\d{4}\s\d{4}\b/;
        const match = text.match(aadhaarPattern);
        if (match) {
          setFormData({ ...formData, aadhaar: match[0].replace(/\s/g, '') }); // Set extracted
Aadhaar number
        }
    };

    const handleSubmit = async (e) => {
        e.preventDefault();
        if (!validateForm()) return;

        try {
          const formDataToSend = new FormData();

          Object.keys(formData).forEach((key) => {
            if (key !== "aadhaarfile") {
              formDataToSend.append(key, formData[key]);
            }
          });

          if (formData.aadhaarfile) {
            formDataToSend.append("aadhaarfile", formData.aadhaarfile);
          }

          const response = await fetch('http://localhost:5000/record', {
            method: 'POST',
            body: formDataToSend,
          });

          if (response.ok) {
            alert("Application submitted successfully");
            const data = await response.json();
            const message = data.message;
            console.log(message);
            navigate('/ConfirmDL', { state: { message } });
          } else {
            alert("Failed to submit application");
          }
        } catch (error) {
          console.error(error);
          alert("An error occurred. Please try again.");
        }
    };

    return (
        <div className='formstyle'>
          <form onSubmit={handleSubmit}>
            <label>Full Name:</label>
            <input type="text" name="fullName" value={formData.fullName}
onChange={handleInputChange} required />
            {errors.fullName && <p style={{ color: 'red' }}>{errors.fullName}</p>}
            <br /><br />

            <label>Date of Birth:</label>
            <input type="date" name="dob" value={formData.dob}
onChange={handleInputChange} required />
            {errors.dob && <p style={{ color: 'red' }}>{errors.dob}</p>}
            <br /><br />

            <label>Address:</label>
            <textarea name="address" rows="2" cols="30" value={formData.address}
onChange={handleInputChange} required></textarea>
            {errors.address && <p style={{ color: 'red' }}>{errors.address}</p>}
            <br /><br />

            <label>Aadhaar Card Upload:</label>
            <input type="file" name="aadhaarfile" onChange={handleFileChange} required />
            {formData.aadhaarfile && <p>Selected file: {formData.aadhaarfile.name}</p>}
            <br /><br />

            <label>Aadhaar No.:</label>
            <input
              type="text"
              name="aadhaar"
              value={formData.aadhaar}
              onChange={handleInputChange}
              pattern="[0-9]{12}"
              placeholder="12 digits"
              required
            />
            {errors.aadhaar && <p style={{ color: 'red' }}>{errors.aadhaar}</p>}
            <br /><br />

            <label>Learner's License Number:</label>
            <input
              type="text"
              name="learnersLicense"
              value={formData.learnersLicense}
              onChange={handleInputChange}
              required
            />
            {errors.aadhaar && <p style={{ color: 'red' }}>{errors.aadhaar}</p>}
            <br /><br />

            <label>City:</label>
            <input type="text" name="city" value={formData.city}
onChange={handleInputChange} required />
            {errors.city && <p style={{ color: 'red' }}>{errors.city}</p>}
            <br /><br />

            <label>Pincode:</label>
            <input type="text" name="pincode" pattern="[0-9]{6}" placeholder="6 digits"
value={formData.pincode} onChange={handleInputChange} required />
            {errors.pincode && <p style={{ color: 'red' }}>{errors.pincode}</p>}
            <br /><br />

            <label>Email:</label>
            <input type="email" name="email" value={formData.email}
onChange={handleInputChange} required />
            {errors.email && <p style={{ color: 'red' }}>{errors.email}</p>}
            <br /><br />
```

```
          <label>Phone No.:</label>
          <input type="text" name="phno" pattern="[0-9]{10}" placeholder="10 digits"
value={formData.phno} onChange={handleInputChange} required />
          {errors.phno && <p style={{ color: 'red' }}>{errors.phno}</p>}
          <br /><br />

          <label>Blood Group:</label>
          <select name="bloodGroup" value={formData.bloodGroup}
onChange={handleInputChange}>
            <option value="O">O</option>
            <option value="A">A</option>
            <option value="B">B</option>
            <option value="AB">AB</option>
          </select>
          {errors.bloodGroup && <p style={{ color: 'red' }}>{errors.bloodGroup}</p>}
          <br /><br />

          <label>Nationality:</label>
          <select name="nationality" value={formData.nationality}
onChange={handleInputChange}>
            <option value="Indian">Indian</option>
            <option value="British">British</option>
            <option value="American">American</option>
            <option value="Canadian">Canadian</option>
            <option value="Russian">Russian</option>
            <option value="French">French</option>
          </select>
          {errors.nationality && <p style={{ color: 'red' }}>{errors.nationality}</p>}
          <br /><br />

          <label>Vehicle Type:</label>
          <select name="vehicleType" value={formData.vehicleType}
onChange={handleInputChange}>
            <option value="twoWheeler">Two-Wheeler</option>
            <option value="car">Car</option>
            <option value="truck">Truck</option>
          </select>
          {errors.vehicleType && <p style={{ color: 'red' }}>{errors.vehicleType}</p>}
          <br /><br />

          <button type="submit">Submit Application</button>
        </form>
```

```
    </div>
  );
};

const ApplicationDL = () => {
  return (
    <div>
      <div className='head'>
        <header>
          <div>
            <img src="/dl.svg" alt="dl" style={{ marginLeft: '0px', width: "100px", height:
"100px", fill: 'white' }} />
            <figcaption style={{ marginLeft: '12px', marginTop: '0px', color:
'white' }}><b><i>Drivify</i></b></figcaption>
            <h1 style={{ color: 'white', textAlign: 'center' }}>Application Form For DL</h1>
          </div>
        </header>
      </div>
      <div>
        <Form />
      </div>
    </div>
  );
};
export default ApplicationDL;
```

ApplicationForm.js

```
import React, { useState } from 'react';
import { useNavigate} from 'react-router';
import { saveApplication}from './api';
import './header_col.css';
import './form_styles.css';

const ApplicationForm = () => {
 const navigate = useNavigate(); // Initialize the navigate hook
  const [formData, setFormData] = useState({
   fullName: "",
   dob: "",
   address: "",
    aadhar: "",
    city: "",
    pincode:"",
    email: "",
    phno: "",
    bloodGroup: "O",
    nationality: "Indian",
    vehicleType: "twoWheeler",
  });


 const handleInputChange = (event) => {
  const { name, value } = event.target;
  setFormData((prevData) => ({
   ...prevData,
   [name]: value,
  }));
 };

 const handleSubmit = async(event) => {
  event.preventDefault();

  // Validation: Check if all fields are filled
  const {
   fullName,
   dob,
   address,
   aadhar,
   city,
   pincode,
   email,
   phno,
   bloodGroup,
   nationality,
   vehicleType
  } = formData;

  if (!fullName || !dob || !address || !aadhar || !city ||!pincode || !email || !phno || !bloodGroup
|| !nationality || !vehicleType) {
   alert("Please fill all the fields before submitting the form.");
   return;  // Prevents submission if any field is empty
  }
  // Check if the Aadhar number is valid (12 digits)
  if (aadhar.length !== 12 || isNaN(aadhar)) {
   alert("Please enter a valid 12-digit Aadhar number.");
   return;
  }

  // Check if phone number is valid (10 digits)
  if (phno.length !== 10 || isNaN(phno)) {
   alert("Please enter a valid 10-digit phone number.");
   return;
  }
  if (pincode.length !== 6  || isNaN(pincode)) {
   alert("Please enter a valid 6-digit pincode number.");
   return;
  }
  try {
   const res = await saveApplication(formData); // Call the API function
      if (res) {
        alert(` Application saved successfully! ID: ${res.id}` );
      }

      } catch (error) {
      console.error("Error:", error);
      alert("An error occurred. Please try again.");
     }
  // If all validations pass, generate an application ID and navigate to the payment page
  // Clear form fields after submission
  setFormData({
   fullName: "",
   dob: "",
   address: "",
   aadhar: "",
   city: "",
   pincode:"",
   email: "",
   phno: "",
   bloodGroup: "O",
   nationality: "Indian",
   vehicleType: "twoWheeler",
  });

  navigate('/payment');  // Navigate to the payment page after successful form submission
 };
 return (
  <div>
      <div className='head'>
      <header>
        <figure>
          <img src="/dl.svg" alt="dl"
style={{marginLeft:'0px',width:"100px",height:"100px",fill:'white'}}/>
          <figcaption
style={{marginLeft:'12px',marginUp:'0px'}}><b><i>Drivify</i></b></figcaption>
          <h1 style={{color:'white',textAlign:'center'}}>Application Form for LL</h1>
        </figure>
      </header>
      </div>
  <section>
    <br/>
    <div className='formstyle'>
      <br/>
    <form onSubmit={handleSubmit}>
      <label>
        Full Name:
        </label>
        <input type="text" name="fullName" value={formData.fullName}
onChange={handleInputChange} required/>
      <br />
      <br/>
      <label >
        Date of Birth: </label>
        <input type="date" name="dob" value={formData.dob}
onChange={handleInputChange} required />
      <br/>
      <br />
      <label >
        Address:</label>
        <textarea
          name="address"  // Correct the name here
          rows="2"
          cols="30"
          value={formData.address}  // Bind the value to formData.address
          onChange={handleInputChange}  // Ensure handleInputChange is working
          required>
         </textarea>
         <br/>
    <br/>
    <label >
     Aadhar no.:</label>
     <input type="text" name="aadhar" pattern="[0-9]{12}" placeholder='12 digits'
value={formData.aadhar} onChange={handleInputChange} required ></input>
    <br/>
    <br />
    <label >
     City: </label>
     <input type="text" name="city" value={formData.city}
onChange={handleInputChange} required ></input>
    <br/>
    <br/>
    <label >
     Pincode:</label>
     <input type="text" name="pincode" pattern="[0-9]{6}" placeholder='6 digits'
value={formData.pincode} onChange={handleInputChange} required ></input>
    <br/>
    <br />
    <label >
     Email: </label>
     <input
      type="email"
      name="email"  // Make sure this is "email"
      value={formData.email}  // Bind it to formData.email
      onChange={handleInputChange}  // Handle the change
      required
     />
    <br/>
    <br/>
    <label >
     Phone no.: </label>
     <input type="text" name="phno" pattern="[0-9]{10}" placeholder='10
digits'value={formData.phno} onChange={handleInputChange} required ></input>
    <br/>
    <br/>
    <label >
     Blood group: </label>
     <select name="bloodGroup" value={formData.bloodGroup}
onChange={handleInputChange}>
      <option value="O">O</option>
      <option value="A">A</option>
      <option value="B">B</option>
```

```
          <option value="AB">AB</option>
        </select>
      <br/>
      <br />
      <label >
        Nationality: </label>
        <select name="nationality" value={formData.nationality}
onChange={handleInputChange}>
          <option value="Indian">Indian</option>
          <option value="British">British</option>
          <option value="American">American</option>
          <option value="Canadian">Canadian</option>
          <option value="Russian">Russian</option>
          <option value="French">French</option>
        </select>
      <br/>
      <br />
      <label >
        Vehicle Type: </label>
        <select name="vehicleType" value={formData.vehicleType}
onChange={handleInputChange}>
          <option value="twoWheeler">Two-Wheeler</option>
          <option value="car">Car</option>
          <option value="truck">Truck</option>
        </select>
      <br/><br/>
      <button type="submit" onClick={handleSubmit}>Submit Application</button>
     </form>
     </div>
    </section>
    </div>
  );
};

export default ApplicationForm;
```

BookAppointment.js

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import './form_styles.css';
import './header_col.css';

const BookAppointment = () => {
  const navigate = useNavigate();

  const [applicationNumber, setApplicationNumber] = useState(''); // Input for application
number
  const [tracks] = useState(["Track A", "Track B", "Track C"]); // Available tracks
  const [selectedTrack, setSelectedTrack] = useState('');
  const [appointmentDate, setAppointmentDate] = useState('');
  const [successMessage, setSuccessMessage] = useState('');
  const [error, setError] = useState(null);

  const handleSubmit = async (e) => {
    e.preventDefault();
    if (!appointmentDate || !selectedTrack || !applicationNumber) {
      setError("Please enter application number, select a date, and track.");
      return;
    }

    try {
      const response = await fetch("http://localhost:5000/record/appointment", {
        method: "POST",
        headers: { "Content-Type": "application/json" },
        body: JSON.stringify({
          applicationNumber,
          appointmentDate,
          testTrack: selectedTrack,
        }),
      });

      if (response.ok) {
        setSuccessMessage("Appointment booked successfully!");
        setError(null);
        setTimeout(() => navigate("/confirmDL", { state: { message: applicationNumber } }),
3000); // Redirect after success
      } else {
        const data = await response.json();
        setError(data.error || "Error booking appointment.");
      }
    } catch (err) {
      console.error(err);
      setError("Error booking appointment.");
    }
  };

  return (
    <div>
      <div className="head">
        <header>
          <div>
            <img src="/dl.svg" alt="dl" style={{ marginLeft: '0px', width: "100px", height:
"100px", fill: 'white' }} />
            <figcaption style={{ marginLeft: '12px', marginTop: '0px', color:
'white' }}><b><i>Drivify</i></b></figcaption>
            <h1 style={{ color: 'white', textAlign: 'center' }}>Book Your Appointment</h1>
          </div>
        </header>
      </div>

      <form onSubmit={handleSubmit} style={{width: "30%", margin: "5% 20% 5% 30%"}}
className='formstyle'>
          <label>Application Number:</label>
          <input type="text" value={applicationNumber} onChange={(e) =>
setApplicationNumber(e.target.value)} required />

          <label>Select Date:</label>
            <input type="date" value={appointmentDate} onChange={(e) =>
setAppointmentDate(e.target.value)} required />

          <label>Select Track:</label>
            <select value={selectedTrack} onChange={(e) =>
setSelectedTrack(e.target.value)} required>
              <option value="" disabled>Select a track</option>
              {tracks.map((track, index) => (
                <option key={index} value={track}>
                  {track}
                </option>
              ))}
            </select>

        <button type="submit">Book Appointment</button>
      </form>

      {error && <p style={{ color: 'red' }}>{error}</p>}
      {successMessage && <p style={{ color: 'green' }}>{successMessage}</p>}
    </div>
  );
};

export default BookAppointment;
```

ConfirmDL.js

```
import React, { useEffect, useState } from 'react';
import { useLocation } from 'react-router-dom';
import './form_styles.css';
import './header_col.css';

const ConfirmDL = () => {
  const location = useLocation();
  const applicationNumber = location.state.message || ''; // Access application number
from state
  console.log(applicationNumber);

  const [applicationData, setApplicationData] = useState(null); // State to hold the fetched
data
  const [error, setError] = useState(null);

  // Fetch application details using the application number
  useEffect(() => {
    const fetchApplicationData = async () => {
      try {
        const response = await
fetch(`http://localhost:5000/record/application/${applicationNumber}`);
        if (response.ok) {
          const data = await response.json();
          setApplicationData(data);
        } else {
          setError('Unable to fetch application details. Please try again later.');
        }
      } catch (err) {
        console.error(err);
        setError('An error occurred while fetching application details.');
      }
    };
    if (applicationNumber) {
      fetchApplicationData();
    }
  }, [applicationNumber]);

  if (error) {
    return <div style={{ color: 'red', textAlign: 'center' }}>{error}</div>;
  }

  if (!applicationData) {
    return <div style={{ textAlign: 'center' }}>Loading application details...</div>;
  }

  // Render the application details
  return (
    <div>
      <div className='head'>
        <header>
          <div>
            <img src="/dl.svg" alt="dl" style={{ marginLeft: '0px', width: "100px", height:
"100px", fill: 'white' }} />
            <figcaption style={{ marginLeft: '12px', marginTop: '0px', color:
'white' }}><b><i>Drivify</i></b></figcaption>
            <h1 style={{ color: 'white', textAlign: 'center' }}>Application Confirmed</h1>
          </div>
        </header>
      </div>
      <div className="formstyle">
        <h2>Application Details</h2>
        <p><strong>Application Number:</strong>
{applicationData.applicationNumber}</p>
        <p><strong>Full Name:</strong> {applicationData.fullName}</p>
        <p><strong>Date of Birth:</strong> {applicationData.dob}</p>
        <p><strong>Address:</strong> {applicationData.address}</p>
        <p><strong>Aadhaar Number:</strong> {applicationData.aadhaar}</p>
        <p><strong>City:</strong> {applicationData.city}</p>
        <p><strong>Pincode:</strong> {applicationData.pincode}</p>
        <p><strong>Email:</strong> {applicationData.email}</p>
        <p><strong>Phone Number:</strong> {applicationData.phno}</p>
        <p><strong>Blood Group:</strong> {applicationData.bloodGroup}</p>
        <p><strong>Nationality:</strong> {applicationData.nationality}</p>
        <p><strong>Vehicle Type:</strong> {applicationData.vehicleType}</p>
        <h3>Booking Details</h3>
        {applicationData.appointmentDate && applicationData.testTrack ? (
          <>
            <p><strong>Booking Date:</strong> {applicationData.appointmentDate}</p>
            <p><strong>Track:</strong> {applicationData.testTrack}</p>
          </>
        ) : (
          <p><strong>Booking Status:</strong> Not booked yet</p>
        )}
      </div>
    </div>
  );
};

export default ConfirmDL;
```

form_styles.css

```
.formstyle {
  padding: 5% 5%;
  border: 1px solid #6a6a6a;
  border-radius: 10px;
  display: flex;
  flex-direction: column;
  align-items: center;
  margin: 5% 20% ;
  width: 50%
}

.formstyle label {
  margin: 8px 0;
}

.formstyle input[type="file"]{
  border: 0px;
}
.formstyle input,
.formstyle select,
.formstyle textarea {
  width: 100%;
  padding: 12px;
  margin-bottom: 10px;
  border-radius: 4px;
  border: 1px solid #ccc;
  box-sizing: border-box; /* Ensures padding and border don't affect width */
}

.formstyle button {
  background-color: #000000;
  color: white;
  padding: 10px 20px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

.formstyle button:hover {
  background-color: #393939;
}
```

header_col.css

```
.head {
  margin-top: 0%;
background: linear-gradient(-45deg, #ffb820, #ff7f08, #bd16ec, #c300ff);
background-size: 400% 400%;
animation: gradient 15s ease infinite;
height: 50vh;
}

@keyframes gradient {
0% {
background-position: 0% 50%;
}
  25%{
    background-position: 75% 25%;
  }
50% {
background-position: 100% 50%;
}
  75%{
    background-position: 75% 25%;
  }
100% {
background-position: 0% 50%;
}
}
```

Home.css

```
.steps{
   display: flex;
   padding: 0% 5% 5% 5%; /*top,right,bottom,left*/
}
.stepbox{
   height:250;
   width:250;
   background-color: rgb(255, 255, 255);
   margin: 3% 3% 3% 3%; /*top, right, bottom, and left*/
   padding: 5% 5% 5% 5% ;
   border-radius: 10px;
   transition: transform 0.3s ease-in-out, box-shadow 0.3s ease-in-out;
}
.stepbox b{
   font-weight: bolder;
   font-size: larger;
}

.stepbox:hover{
   box-shadow: inset 0 0 0 1000px rgba(245, 161, 152, 0.5);
   transform: scale(1.2);
}
.faq
{
   background-color: rgb(226, 43, 43);
   color:white;
}
```

Home.js

```
import React from 'react';
import { useNavigate } from 'react-router-dom'; // Import useNavigate
import './Home.css';
const HomePage = () => {
  const navigate = useNavigate(); // Initialize the navigate hook

  const handleApplyClick = () => {
    navigate('/instructions'); // Navigate to the instructions page
  };
  const handlePrintClick=()=>{
    navigate('/print');// Navigate to printll page
  };
  const handleQuizClick=()=>
  {
    navigate('/quiz');
  };
  const handleApplyDLClick=()=>
  {
    navigate('/ApplicationDL');
  };
  const handleBookDLClick=()=>
  {
    navigate('/bookdl');
  };
  const handlePrintDLClick=()=>
  {
    navigate('/printdl')
  };
  return (
    <div>
      <div className='head'>
        <header>
          <figure>
            <img src="/dl.svg" alt="dl" style={{ marginLeft: '0px', width: "100px", height:
"100px", fill: 'white' }} />
            <figcaption
style={{marginLeft:'12px',marginUp:'0px'}}><b><i>Drivify</i></b></figcaption>
            <h1 style={{ color: 'white', textAlign: 'center' }}>Home</h1>
          </figure>
        </header>
      </div>
      <main>
        <main>
          <h1 style={{marginLeft:'500px'}}>Welcome to Drivify!</h1>
```

```
            <p style={{marginLeft:'380px',fontSize:'25px'}}>Your one-stop solution for
managing driving licences.</p>
            <div style={{alignItems:'center'}} className="steps">
              <label style={{fontSize:'25px'}}>Learner's Licence: </label><br/>
              <div className='stepbox' onClick={handleApplyClick}><img src='/form.png'
alt="Form logo" style={{height:120,width:120}}></img><p
style={{fontSize:'25px'}}>Apply</p>{/* Navigate on click */}</div>
              <div className='stepbox' onClick={handlePrintClick}><img src='/print.png'
alt="Print logo" style={{height:120,width:120}}></img><p style={{fontSize:'25px'}}>Print
Forms</p></div>
              <div className='stepbox' onClick={handleQuizClick}><img
src='/test_home.png'  alt="Test logo" style={{height:120,width:120}}></img><p
style={{fontSize:'25px'}}>Take Test</p></div>
            </div>
            <div style={{alignItems:'center'}} className="steps">
              <label style={{fontSize:'25px'}}>Driving Licence: </label><br/>
              <div className='stepbox'onClick={handleApplyDLClick}><img src='/form.png'
alt="Form logo" style={{height:120,width:120}}></img><p
style={{fontSize:'25px'}}>Apply</p>{/* Navigate on click */}</div>
              <div className='stepbox' onClick={handleBookDLClick}><img
src='/calendar.png'  alt="Calendar logo" style={{height:120,width:120}}></img><p
style={{fontSize:'25px'}}>Book Appointment</p><br/></div> {/* Navigate on click */}
              <div className='stepbox' onClick={handlePrintDLClick}><img src='/print.png'
alt="Print logo" style={{height:120,width:120}}></img><p style={{fontSize:'25px'}}>Print
Forms</p></div>
            </div>
            <section>
              <marquee className="faq" style={{fontSize:'25px'}}><b>Frequently Asked
Questions:</b> What documents do I need to apply?    <b>User Testimonials:</b> "Drivify
made the application process so easy!"</marquee>
            </section>

          </main>
        </main>
      </div>
  );
};
 export default HomePage;
```

index.css

```
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
    sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}

code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}
```

index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
 // If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

Instructions.js

```
import React from 'react';
import { useNavigate } from 'react-router-dom'; // Import useNavigate for navigation
import './header_col.css';
import './form_styles.css';

const Instructions = () => {
  const navigate = useNavigate(); // Initialize useNavigate hook for redirection

  const handleOkClick = () => {
    navigate('/apply'); // Redirects to the application form page
  };

  return (
    <div>
      <div className='head'>
        <header>
          <figure>
            <img src="/dl.svg" alt="dl" style={{ marginLeft: '0px', width: "100px", height:
"100px", fill: 'white' }} />
            <figcaption
style={{marginLeft:'12px',marginUp:'0px'}}><b><i>Drivify</i></b></figcaption>
            <h1 style={{ color: 'white', textAlign: 'center' }}>Application for Learner's
Licence</h1>
          </figure>
        </header>
      </div>
      <br/>
      <div className='formstyle'>
        <br/>
        <h2>Instructions</h2><br/>
        <ol style={{textAlign:'left',paddingLeft: '20px', listStylePosition: 'inside'}}>
          <li style={{fontSize:'20px'}}>Fill the application details</li><br/>
          <li style={{fontSize:'20px'}}>Fee payment</li><br/>
          <li style={{fontSize:'20px'}}>Verify the payment status</li><br/>
          <li style={{fontSize:'20px'}}>Print the application</li><br/>
        </ol>
        <button onClick={handleOkClick} style={{ padding: '10px 20px', marginTop: '20px',
fontSize: '16px' }}>
          OK
        </button>
      </div>
    </div>
  );
};

export default Instructions;
```

Payment.js

```
import React from "react";
import './form_styles.css';
import './header_col.css';

const Payment=()=>
{
  return(
    <div>
      <div className='head'>
        <header>
          <figure>
            <img src="/dl.svg" alt="dl"
style={{marginLeft:'0px',width:"100px",height:"100px",fill:'white'}}/>
            <figcaption
style={{marginLeft:'12px',marginUp:'0px'}}><b><i>Drivify</i></b></figcaption>
            <h1 style={{color:'white',textAlign:'center'}}>Payment</h1>
          </figure>
        </header>
      </div>
      <br/>
      <div className='formstyle'><br/>
        <h3 style={{fontSize:20}}>Pay by :</h3><br/>
        <ul style={{textAlign:'left',paddingLeft: '20px', listStylePosition: 'inside'}}>
          <li style={{fontSize:20}}>G Pay</li><br/>
          <li style={{fontSize:20}}>PhonePe</li><br/>
          <li style={{fontSize:20}}>Net Banking</li><br/>
          <li style={{fontSize:20}}>Credit/Debit Card</li><br/>
        </ul>
      </div>
    </div>
  )
}
export default Payment;
```

PrintApplication.js

```
import React, { useState } from "react";
import { getApplication } from "./api"; // Assuming getApplication is defined in api.js
import './header_col.css';
import './form_styles.css';
const Print = () => {

  const [formData, setFormData] = useState({
    fullName: "",
  });
  const [application, setApplication] = useState(null); // State to store application details
  const [error, setError] = useState(null); // State to store any errors

  const handleInputChange = (event) => {
    const { name, value } = event.target;
    setFormData((prevData) => ({
      ...prevData,
      [name]: value,
    }));
  };

  const handleSubmit = async (event) => {
    event.preventDefault();

    // Destructure to get the fullName
    const { fullName } = formData;

    // Validation
    if (!fullName) {
      alert("Please fill full name before submitting the form.");
      return;
    }

    try {
      // Fetch application data by full name (assuming `getApplication` accepts full name or
application ID)
      const applicationData = await getApplication(fullName);

      if (applicationData) {
        setApplication(applicationData); // Set application data to display on the page
        setError(null);
```

```
   } else {
    setApplication(null);
    setError("No application found for the given name.");
   }
  } catch (error) {
   console.error("Error fetching application:", error);
   setError("Failed to retrieve application. Please try again.");
  }

  // Reset the form data
  setFormData({
   fullName: "",
  });
 };

 return (
  <div>
   <div className='head'>
    <header>
     <figure>
      <img
       src="/dl.svg"
       alt="dl"
       style={{ marginLeft: "0px", width: "100px", height: "100px" }}
      />
      <figcaption style={{ marginLeft: "12px", marginTop: "0px" }}>
       <b><i>Drivify</i></b>
      </figcaption>
      <h1 style={{ color: "white", textAlign: "center" }}>
       Print Application Form for LL
      </h1>
     </figure>
    </header>
   </div>
   <br />
   <div ><br/>
    <form onSubmit={handleSubmit} className="formstyle">
     <label style={{fontSize:"25px",marginLeft:"10px"}}>Full Name: </label>
     <input style={{fontSize:"25px"}}
      type="text"
      name="fullName"
      value={formData.fullName}
      onChange={handleInputChange}
```

```
          required
        />
        <br /><br/>
        <button style={{fontSize:20}}type="submit">Submit</button>
      </form>
    </div>
    <br />

    {/* Display application details if available */}
    {application && (
      <div style={{ border: "1px solid #ddd", padding: "20px", marginTop:
"20px" ,marginLeft:"450px", width:"400px"}}>
        <h2>Application Details</h2>
        <p><strong>ID:</strong> {application._id}</p>
        <p><strong>Name:</strong> {application.fullName}</p>
        <p><strong>Date of Birth:</strong> {application.dob}</p>
        <p><strong>Address:</strong> {application.address}</p>
        <p><strong>Aadhar:</strong> {application.aadhar}</p>
        <p><strong>City:</strong> {application.city}</p>
        <p><strong>Pincode:</strong> {application.pincode}</p>
        <p><strong>Email:</strong> {application.email}</p>
        <p><strong>Phone Number:</strong> {application.phno}</p>
        <p><strong>Blood Group:</strong> {application.bloodGroup}</p>
        <p><strong>Aadhar:</strong> {application.aadhar}</p>
        <p><strong>Nationality:</strong> {application.nationality}</p>
        <p><strong>Vehicle Type:</strong> {application.vehicleType}</p>
      </div>
    )}

    {/* Display error message if no application found or error occurs */}
    {error && <p style={{ color: "red" ,marginLeft: "450px",fontSize:"25px"}}>{error}</p>}
   </div>
  );
};

export default Print;
```

PrintDL.js

```
import React, { useState} from 'react';
import './form_styles.css';
import './header_col.css';
```

```
const PrintDL = () => {
  const [applicationNumber, setApplicationNumber] = useState(''); // State to store the
input application number
  const [applicationData, setApplicationData] = useState(null); // State to hold the fetched
data
  const [error, setError] = useState(null);

  const handleInputChange = (e) => {
    setApplicationNumber(e.target.value); // Update the application number on input
change
  };

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

    if (!applicationNumber) {
      setError('Please enter an application number.');
      return;
    }

    // Fetch application details using the application number
    try {
      const response = await
fetch(`http://localhost:5000/record/application/${applicationNumber}`);
      if (response.ok) {
        const data = await response.json();
        setApplicationData(data);
        setError(null); // Clear any previous error
      } else {
        setError('Unable to fetch application details. Please check the application number
and try again.');
      }
    } catch (err) {
      console.error(err);
      setError('An error occurred while fetching application details.');
    }
  };

  const handlePrint = () => {
    window.print(); // Trigger print dialog
  };
  if (error) {
    return <div style={{ color: 'red', textAlign: 'center' }}>{error}</div>;
  }

  if (!applicationData) {
    return (
      <div>
        <div className='head'>
          <header>
            <div>
              <img src="/dl.svg" alt="dl" style={{ marginLeft: '0px', width: "100px", height:
"100px", fill: 'white' }} />
              <figcaption style={{ marginLeft: '12px', marginTop: '0px', color:
'white' }}><b><i>Drivify</i></b></figcaption>
              <h1 style={{ color: 'white', textAlign: 'center' }}>Print Details</h1>
            </div>
          </header>
        </div>
        {/* Input form for application number */}
        <div className="formstyle" style={{ textAlign: 'center', marginTop: '50px' }}>
          <h2>Enter Application Number</h2>
          <form onSubmit={handleSubmit}>
            <input
              type="text"
              value={applicationNumber}
              onChange={handleInputChange}
              required
              style={{ padding: '10px', fontSize: '16px' }}
            />
            <button type="submit" style={{ padding: '10px 20px', fontSize: '16px',
marginLeft: '10px' }}>
              Fetch Details
            </button>
          </form>
        </div>
      </div>
    );
  }

  // Render the application details and print option after data is fetched
  return (
    <div>
      <div className='head'>
```

```
        <header>
          <div>
            <img src="/dl.svg" alt="dl" style={{ marginLeft: '0px', width: "100px", height:
"100px", fill: 'white' }} />
            <figcaption style={{ marginLeft: '12px', marginTop: '0px', color:
'white' }}><b><i>Drivify</i></b></figcaption>
            <h1 style={{ color: 'white', textAlign: 'center' }}>Print Details</h1>
          </div>
        </header>
      </div>
      <div className="formstyle">
        <h2>Application Details</h2>
        <p><strong>Application Number:</strong>
{applicationData.applicationNumber}</p>
        <p><strong>Full Name:</strong> {applicationData.fullName}</p>
        <p><strong>Date of Birth:</strong> {applicationData.dob}</p>
        <p><strong>Address:</strong> {applicationData.address}</p>
        <p><strong>Aadhaar Number:</strong> {applicationData.aadhaar}</p>
        <p><strong>City:</strong> {applicationData.city}</p>
        <p><strong>Pincode:</strong> {applicationData.pincode}</p>
        <p><strong>Email:</strong> {applicationData.email}</p>
        <p><strong>Phone Number:</strong> {applicationData.phno}</p>
        <p><strong>Blood Group:</strong> {applicationData.bloodGroup}</p>
        <p><strong>Nationality:</strong> {applicationData.nationality}</p>
        <p><strong>Vehicle Type:</strong> {applicationData.vehicleType}</p>

        {/* Check if booking is done */}
        <h3>Booking Details</h3>
        {applicationData.appointmentDate && applicationData.testTrack ? (
          <>
            <p><strong>Booking Date:</strong> {applicationData.appointmentDate}</p>
            <p><strong>Track:</strong> {applicationData.testTrack}</p>
          </>
        ) : (
          <p><strong>Booking Status:</strong> Not booked yet</p>
        )}

        {/* Button to print the page */}
        <button onClick={handlePrint} style={{ marginTop: '20px' }}>Print Details</button>
      </div>
    </div>
  );
};
```

```
export default PrintDL;
```

Question.js

```
import React from 'react';
 const Question = ({ question, index, selectedAnswer, onAnswerChange }) => {
  return (
    <div className="question">
      <p>{index + 1}. {question.text}</p>
      {question.options.map((option, idx) => (
        <label key={idx}>
          <input
            type="radio"
            name={`question-${question._id}`}
            value={option}
            checked={selectedAnswer === option}
            onChange={() => onAnswerChange(question._id, option)}
          />
          {option}
        </label>
      ))}
    </div>
  );
};

export default Question;
```

QuestionBubble.js

```
import React from 'react';
const QuestionBubble = ({ number, isAnswered, onClick }) => {
  return (
    <button
      className={`bubble ${isAnswered ? 'answered' : ''}`}
      onClick={onClick}
    >
      {number}
  </button>
 );
};

export default QuestionBubble; import React from 'react';

const QuestionBubble = ({ number, isAnswered, onClick }) => {
 return (
  <button
   className={`bubble ${isAnswered ? 'answered' : ''}`}
   onClick={onClick}
  >
   {number}
  </button>
 );
};

export default QuestionBubble;
```

Quiz.css

```
.App {
  font-family: Arial, sans-serif;
  text-align: center;
  background-color: #f4f4f9;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
 }

 .App-header {
  background-color: #0056a2;
  color: white;
  padding: 20px;
 }

 .test-header {
  display: flex;
  justify-content: space-between;
  align-items: center;
```

```
}

.test-body {
  display: flex;
  justify-content: space-between;
  padding: 20px;
}

.question-section {
  flex: 3;
  text-align: left;
  margin-right: 20px;
  background: white;
  padding: 20px;
  border-radius: 10px;
  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}

.navigation-section {
  flex: 1;
  display: flex;
  flex-direction: column;
  justify-content: flex-start;
  align-items: center;
}

.bubble-navigation {
  display: grid;
  grid-template-columns: repeat(5, 1fr); /* 5 bubbles per row */
  gap: 10px; /* Space between bubbles */
  width: 100%;
  max-width: 300px; /* Adjust to fit exactly two rows */
  margin-bottom: 20px;
}

.bubble {
  width: 40px;
  height: 40px;
  border-radius: 50%;
  display: flex;
  justify-content: center;
  align-items: center;
  background-color: #e4e4e4;
  color: #333;
  font-weight: bold;
  cursor: pointer;
  border: 1px solid #ccc;
  font-size: 16px;
}

.bubble.answered {
  background-color: #4caf50;
  color: white;
}

.bubble:hover {
  background-color: #0056a2;
  color: white;
}

.end-test {
  background-color: #ff4d4d;
  color: white;
  border: none;
  padding: 10px 20px;
  font-size: 16px;
  cursor: pointer;
  border-radius: 5px;
  margin-top: 20px;
}

.end-test:hover {
  background-color: #cc0000;
}

.question-image {
  display: block;
  margin: 10px auto;
  max-width: 100%;
  height: auto;
  border-radius: 5px;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}
```

Quiz.js

```
import React, { useState } from 'react';
import './Quiz.css';
import QuestionBubble from './QuestionBubble';
import Timer from './Timer';

const Quiz = () => {
 const questions = [
  {
   _id: '1',
   text: 'Near a pedestrian crossing, when pedestrians are waiting to cross the road, you
should...',
   options: [
    'Sound horn and proceed',
    'Slow down, sound horn and pass',
    'Stop the vehicle and wait till the pedestrians cross the road and then proceed',
   ],
  },
  {
   _id: '2',
   text: 'The following sign represents...',
   image: '/q2.png', // Image URL relative to public folder
   options: ['Stop', 'No parking', 'Hospital ahead'],
  },
  {
   _id: '3',
   text: 'You are approaching a narrow bridge. Another vehicle is about to enter the bridge
from opposite side. You should...',
   options: [
    'Increase the speed and try to cross the bridge as fast as possible',
    'Put on the head light and pass the bridge',
    'Wait till the other vehicle crosses the bridge and then proceed',
   ],
  },
  {
   _id: '4',
   text: 'The following sign represents...',
   image: '/q4.png', // Image URL relative to public folder
   options: ['Keep left', 'There is no road to the left', 'Compulsory turn left'],
  },
  {
   _id: '5',
   text: 'When a vehicle is involved in an accident causing injury to any person...',
    options: [
      'Take the vehicle to the nearest police station and report the accident',
      'Stop the vehicle and report to the police station',
      'Take all reasonable steps to secure medical attention for the injured and report to the
nearest police station within 24 hours',
    ],
  },
  {
    _id: '6',
    text: 'The following sign represents...',
    image: '/q6.png', // Image URL relative to public folder
    options: ['Give way', 'Hospital ahead', 'Traffic island ahead'],
  },
  {
    _id: '7',
    text: 'On a road designated as one way...',
    options: [
      'Parking is prohibited',
      'Overtaking is prohibited',
      'Should not drive in reverse gear',
    ],
  },
  {
    _id: '8',
    text: 'The following sign represents...',
    image: '/q8.png', // Image URL relative to public folder
    options: ['No entry', 'One way', 'Speed limit ends'],
  },
  {
    _id: '9',
    text: 'You can overtake a vehicle...',
    options: [
      'Through the right side of the vehicle',
      'Through the left side',
      'Through the left side, if the road is wide',
    ],
  },
  {
    _id: '10',
    text: 'The following sign represents...',
    image: '/q10.png', // Image URL relative to public folder
    options: ['Right turn prohibited', 'Sharp curve to the right', 'U-turn prohibited'],
  },
];

const [currentQuestionIndex, setCurrentQuestionIndex] = useState(0);
const [answers, setAnswers] = useState({});
const [applicationId] = useState('PES120210001');

const handleAnswer = (questionId, answer) => {
 setAnswers({ ...answers, [questionId]: answer });
};

const handleEndTest = () => {
 const payload = {
  applicationId,
  answers,
 };

 fetch('http://localhost:5000/api/tests/submit', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify(payload),
 })
  .then((res) => {
   if (res.ok) {
    alert('Test submitted successfully!');
    window.location.reload(); // Refresh the page
   } else {
    alert('Error submitting test');
   }
  })
  .catch((err) => console.error(err));
};

const handleNavigate = (index) => {
 setCurrentQuestionIndex(index);
};

return (
 <div className="App">
  <header className="App-header">
   <div className="test-header">
    <h1>Driving Licence Test</h1>
    <Timer duration={20 * 60} onTimeUp={handleEndTest} />
   </div>
```

```
</header>
<div className="test-body">
 <div className="question-section">
  {questions.length > 0 && (
   <div>
    <h3>
     {currentQuestionIndex + 1}. {questions[currentQuestionIndex].text}
    </h3>
    {questions[currentQuestionIndex].image && (
     <img
      src={questions[currentQuestionIndex].image}
      alt={`Question ${currentQuestionIndex + 1}`}
      className="question-image"
     />
    )}
    {questions[currentQuestionIndex].options.map((option, index) => (
     <div key={index}>
      <label>
       <input
        type="radio"
        name={`question-${currentQuestionIndex}`}
        value={option}
        onChange={() =>
         handleAnswer(questions[currentQuestionIndex]._id, option)
        }
        checked={answers[questions[currentQuestionIndex]._id] === option}
       />
       {option}
      </label>
     </div>
    ))}
   </div>
  )}
 </div>
 <div className="navigation-section">
  <div className="bubble-navigation">
   {questions.map((question, index) => (
    <QuestionBubble
     key={question._id}
     number={index + 1}
     isAnswered={Boolean(answers[question._id])}
     onClick={() => handleNavigate(index)}
    />
```

```
          ))}
        </div>
        <button className="end-test" onClick={handleEndTest}>
          End Test
        </button>
      </div>
    </div>
  </div>
 );
};

export default Quiz;
```

reportWebVitals.js

```
const reportWebVitals = onPerfEntry => {
  if (onPerfEntry && onPerfEntry instanceof Function) {
    import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {
      getCLS(onPerfEntry);
      getFID(onPerfEntry);
      getFCP(onPerfEntry);
      getLCP(onPerfEntry);
      getTTFB(onPerfEntry);
    });
  }
};

export default reportWebVitals;
```

SelectState.css

```
.top {
    background-image: url("/images/DLHome2.png");
    height: 600px;
    background-repeat: no-repeat;
    background-size: cover;
    background-position: center;
    box-shadow: inset 0 0 0 1000px rgba(0, 0, 0, 0.5);
    color: white;
}
```

```
.header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 10px 20px;
}

.logo {
  display: flex;
  align-items: center;
}

.nav {
  display: flex;
  gap: 20px;
}
.nav button:hover{
  font-weight: bolder;
}

.link {
  color: white;
  text-decoration: none;
  font-size: 20px;
  background: none;
  border: none;
  cursor: pointer;
}

.main {
  margin: auto;
  text-align: center;
}

.dropdown-container {
  margin: auto;
}

.dropdown {
  padding: 10px;
  font-size: 16px;
}
```

```
.dl {
  display: block;
  padding-left: 50px;
}

.steps{
  display: flex;
  padding: 0% 5% 5% 5%; /*top,right,bottom,left*/
}

.stepbox{
  background-color: rgb(255, 255, 255);
  margin: 3% 3% 3% 3%; /*top, right, bottom, and left*/
  padding: 5% 5% 5% 5% ;
  border-radius: 10px;
  transition: transform 0.3s ease-in-out, box-shadow 0.3s ease-in-out;
}
.stepbox b{
  font-weight: bolder;
  font-size: larger;
}

.stepbox:hover{
  box-shadow: inset 0 0 0 1000px rgba(196, 196, 196, 0.5);
  transform: scale(1.2);
}

.about{
  background-image: url("/images/about2.jpg");
  box-shadow: inset 0 0 0 1000px rgba(0, 0, 0, 0.7);
  color: white;
}

.about img{
  border-radius: 50%;
}

.objective{
  margin: 3% 3% 3% 3%; /*top, right, bottom, and left*/
  padding: 5% 2% 5% 5% ;
  flex-basis: 33.33%; /* Initial width of 33.33% */
  flex-grow: 1;
}
```

SelectState.js

```
import React, { useState, useRef, useEffect } from 'react';
import { useNavigate} from 'react-router-dom';
import './SelectState.css';
import 'aos/dist/aos.css';
import Aos from 'aos';

const Steps = React.forwardRef((props, ref) => {
  return (
    <div ref={ref} style={{paddingLeft: '50px', backgroundColor: 'rgb(243, 243, 243)'}}>
      <h1>Steps To Apply</h1>
      <div className="steps">
        <div className='stepbox'>
          <img src="./apply.png" alt="apply" width="70" />
          <br/>
          <b>Apply</b>
          <br/>
          <p>Select your state</p>
          <p>Fill the application form</p>
          <p>Get your application number</p>
        </div>
        <div className='stepbox'>
          <img src="./test.png" alt="test" width="70" />
          <br/>
          <b>Test</b>
          <br/>
          <p>Refer to our videos for the test</p>
          <p>Login to the test</p>
          <p>Give the test</p>
        </div>
        <div className='stepbox'>
          <img src="./complete.png" alt="test" width="70" />
          <br/>
          <b>Get Your License</b>
          <br/>
          <p>Wait for test results</p>
          <p>Get your Driving License!</p>
        </div>
      </div>
      </div>
   );
});


const About = React.forwardRef((props, ref) => {
   return (
      <div ref={ref} className="about">
         <h1>About Us</h1>
         <div>
            <div style={{display: 'flex'}} data-aos="fade-up" data-aos-delay="300">
               <div className='objective'>
                  <img src="/1.png" alt="vision" height="50px"/>
                  <h3>Vision</h3>
                  <p>To improve the quality of service delivery to the citizen and the quality of
work environment of the RTOs.</p>
               </div>
               <div className='objective' data-aos="fade-up" data-aos-delay="500">
                  <img src="/2.png" alt="mission" height="50px"/>
                  <h3>Mission</h3>
                  <p>To automate all Vehicle Registration and Driving License related activities in
transport authorities of country with introduction of smart card technology to handle
issues like inter state transport vehicle movement and to create state and national level
registers of vehicles/DL information</p>
               </div>
               <div className='objective' data-aos="fade-up" data-aos-delay="700">
                   <img src="/3.png" alt="obj" height="50px"/>
                  <h3>Objectives</h3>
                     <p>Better services to Transport Department as well as citizen</p>
                     <p>Instant access of Vehicle/DL information to other government
departments</p>
               </div>

            </div>
         </div>
      </div>
   );
});

const Contact = React.forwardRef((props, ref) => {
   return (
      <div ref={ref} className="dl">
         <h1>Contact</h1>
```

```
        <div style={{ display: 'flex', padding:'5% 2% 5% 5%' }}>
          <div data-aos="fade-right">
            <img src="./ContactUs.jpg" alt="car" width="200" />
          </div>
          <div style={{padding:'0% 2% 0% 5%' }} data-aos="fade-left" >
            <h2>Phone</h2>
            <p>+91 0123456789</p>
            <h2>Email</h2>
            <p>abcd@xyz.com</p>
          </div>
        </div>
      </div>
  );
});

const SelectState = () => {
  const [selectedState, setSelectedState] = useState('Select State');
  const states = ['Karnataka', 'Maharashtra', 'Tamil Nadu', 'Kerala', 'Other'];
  const navigate = useNavigate();
  const aboutRef = useRef(null);
  const contactRef = useRef(null);
  const stepsRef = useRef(null);

  useEffect(() => {
    Aos.init({ duration: 600 });
  }, []);

  const handleStateChange = (event) => {
    setSelectedState(event.target.value);
    if (event.target.value !== 'Select State') {
      navigate('/home');
    }
  };

  const scrollToSteps = () => {
    if (stepsRef.current) {
      stepsRef.current.scrollIntoView({ behavior: 'smooth' });
    }
  };
  const scrollToAbout = () => {
    if (aboutRef.current) {
      aboutRef.current.scrollIntoView({ behavior: 'smooth' });
    }
  };

  const scrollToContact = () => {
    if (contactRef.current) {
      contactRef.current.scrollIntoView({ behavior: 'smooth' });
    }
  };

  return (
    <div>
      <div className="top">
        <header className="header">
          <div className="logo">
            <img src="/dl.svg" alt="Drivify Logo" />
            <h1>Drivify</h1>
          </div>
          <nav className="nav">
            <button onClick={scrollToSteps} className="link">Steps to Apply</button>
            <button onClick={scrollToAbout} className="link">About</button>
            <button onClick={scrollToContact} className="link">Contact</button>
          </nav>
        </header>

        <div className="main">
          <h1 style={{ fontSize: '40px' }}>Get Your Driving License!</h1>
          <h2>Select State</h2>
          <div className="dropdown-container">
            <select className="dropdown" value={selectedState}
onChange={handleStateChange}>
              <option value="Select State" disabled>Select State</option>
              {states.map((state) => (
                <option key={state} value={state}>{state}</option>
              ))}
            </select>
          </div>
        </div>
      </div>
      <div data-aos="fade-up">
        <Steps ref={stepsRef} />
      </div>
      <div data-aos="fade-up" data-aos-delay="200">
        <About ref={aboutRef} />
      </div>
      <div>
        <Contact ref={contactRef} />
      </div>
    </div>
  );
};

export default SelectState;
```

SetupTests.js

```
// jest-dom adds custom jest matchers for asserting on DOM nodes.
// allows you to do things like:
// expect(element).toHaveTextContent(/react/i)
// learn more: https://github.com/testing-library/jest-dom
import '@testing-library/jest-dom';
```

Submission.js

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

const SubmissionSchema = new mongoose.Schema({
 applicationId: { type: String, required: true },
 answers: { type: Array, required: true },
});

module.exports = mongoose.model('Submission', SubmissionSchema);
```

Test.css

```
.test-container {
  max-width: 800px;
  margin: auto;
  padding: 20px;
  background-color: #f9f9f9;
  border-radius: 10px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
 }

 header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 20px;
 }

 .timer {
  font-size: 18px;
  font-weight: bold;
  color: #d9534f;
 }

 .questions-container {
  margin-bottom: 20px;
 }

 .question {
  margin-bottom: 15px;
 }

 .question p {
  font-size: 16px;
  font-weight: bold;
 }

 label {
  display: block;
  margin: 5px 0;
 }

 footer {
  text-align: center;
 }

 .submit-button {
  padding: 10px 20px;
  background-color: #5cb85c;
  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  font-size: 16px;
 }
```

```
 .submit-button:hover {
  background-color: #4cae4c;
 }
```

Test.js

```
import React, { useState, useEffect } from 'react';
import axios from 'axios';
import Timer from './Timer';
import Question from './Question';
import './Test.css';

const Test = () => {
 const [questions, setQuestions] = useState([]);
 const [answers, setAnswers] = useState({});
 const [applicationId] = useState('USER_APP_ID'); // Replace with dynamic user ID if
available

 useEffect(() => {
  // Fetch questions when the component mounts
  axios.get('/api/tests/questions').then((response) => {
   setQuestions(response.data);
  });
 }, []);

 const handleAnswerChange = (questionId, answer) => {
  setAnswers((prevAnswers) => ({
   ...prevAnswers,
   [questionId]: answer,
  }));
 };

 const handleSubmit = () => {
  axios.post('/api/tests/submit', { applicationId, answers }).then((response) => {
   alert(response.data.message || 'Test submitted successfully!');
   window.location.reload(); // Reload to reset
  });
 };

 return (
  <div className="test-container">
```

```
   <header>
    <h1>Online Quiz</h1>
    <Timer duration={20 * 60} onTimeUp={handleSubmit} />
   </header>
   <div className="questions-container">
    {questions.map((question, index) => (
     <Question
      key={question._id}
      question={question}
      index={index}
      selectedAnswer={answers[question._id]}
      onAnswerChange={handleAnswerChange}
     />
    ))}
   </div>
   <footer>
    <button className="submit-button" onClick={handleSubmit}>
     End Test
    </button>
   </footer>
  </div>
 );
};

export default Test;
```

Timer.js

```
import React, { useEffect, useState } from 'react';

const Timer = ({ duration, onTimeUp }) => {
 const [timeLeft, setTimeLeft] = useState(duration);

 useEffect(() => {
  const timer = setInterval(() => {
   setTimeLeft((prev) => {
    if (prev <= 1) {
     clearInterval(timer);
     onTimeUp();
     return 0;
    }
```

```
        return prev - 1;
      });
    }, 1000);

    return () => clearInterval(timer);
  }, [onTimeUp]);

  const formatTime = (seconds) => {
    const minutes = Math.floor(seconds / 60);
    const secs = seconds % 60;
    return `${minutes.toString().padStart(2, '0')}:${secs.toString().padStart(2, '0')}`;
  };

  return <div className="timer">Time Left: {formatTime(timeLeft)}</div>;
};

export default Timer;
```

Server

db

conn.js

```
const { MongoClient, ServerApiVersion } = require("mongodb");

const uri = "mongodb://localhost:27017/Drivify";
const client = new MongoClient(uri, {
  serverApi: {
    version: ServerApiVersion.v1,
    strict: true,
    deprecationErrors: true,
  }
});

async function connectDb() {
  try {
    await client.connect();
    client.db("test").command({ ping: 1 });
    console.log("Successfully connected to MongoDB");
  } catch (err) {
    console.log(err);
  }
}

let db = client.db("driver");

// Exporting the db connection object
module.exports = db;  // Export the db object

// Or if you prefer to export the connectDb function:
module.exports.connectDb = connectDb;
```

Routes

record.js

```
const express = require("express");

// This will help us connect to the database
const db = require("../db/conn.js");

// This helps convert the id from string to ObjectId for the _id.
const { ObjectId } = require("mongodb");

// For file upload
const multer = require("multer");
const path = require("path");

// Router is an instance of the express router.
// We use it to define our routes.
// The router will be added as a middleware and will take control of requests starting with
path /record.
const router = express.Router();

// Use timestamp (13-digit) for application number
const applicationNumber = Date.now().toString();

// Set up multer storage options
const storage = multer.diskStorage({
  destination: (req, file, cb) => {
    cb(null, 'uploads/');  // Folder where files will be stored
```

```
 },
 filename: (req, file, cb) => {
  cb(null, applicationNumber + path.extname(file.originalname));  //
path.extname(file.originalname) for .png
 }
});

const upload = multer({ storage });

// This section will help you get a list of all the records.
router.get("/", async (req, res) => {
 let collection = await db.collection("records");
 let results = await collection.find({}).toArray();
 res.status(200).send(results);
});

// This section will help you get a single record by id
router.get("/application/:applicationNumber", async (req, res) => {
 try {
  let collection = await db.collection("records");
  const query = { applicationNumber: req.params.applicationNumber };
  const result = await collection.findOne(query);

  if (!result) return res.status(404).send({ error: "Application not found" });
  res.status(200).send(result);
 } catch (err) {
  console.error(err);
  res.status(500).send("Error fetching application details");
 }
});

// This section will help you create a new record.
router.post("/", upload.single("aadhaarfile"), async (req, res) => {
 try {
  // Collecting form data along with the uploaded file path
  const newDocument = {
   applicationNumber,
   fullName: req.body.fullName,
   dob: req.body.dob,
   address: req.body.address,
   aadhaarfile: req.file ? req.file.path : null, // Save file path in DB
   aadhaar: req.body.aadhaar,
   learnersLicense: req.body.learnersLicense,
   city: req.body.city,
   pincode: req.body.pincode,
   email: req.body.email,
   phno: req.body.phno,
   bloodGroup: req.body.bloodGroup,
   nationality: req.body.nationality,
   vehicleType: req.body.vehicleType,
  };

  let collection = await db.collection("records");
  let result = await collection.insertOne(newDocument);
  res.status(200).send({ message: applicationNumber });
 } catch (err) {
  console.error(err);
  res.status(500).send("Error adding record");
 }
});

// This section will help you update a record by id.
router.patch("/:id", async (req, res) => {
 try {
  const query = { _id: new ObjectId(req.params.id) };
  const updates = {
   $set: {
    name: req.body.name,
    position: req.body.position,
    level: req.body.level,
   },
  };

  let collection = await db.collection("records");
  let result = await collection.updateOne(query, updates);
  res.status(200).send(result);
 } catch (err) {
  console.error(err);
  res.status(500).send("Error updating record");
 }
});

// This section will help you delete a record
router.delete("/:id", async (req, res) => {
 try {
  const query = { _id: new ObjectId(req.params.id) };
```

```
    const collection = db.collection("records");
    let result = await collection.deleteOne(query);

    res.status(200).send(result);
  } catch (err) {
    console.error(err);
    res.status(500).send("Error deleting record");
  }
});

// POST Route to book an appointment
router.post("/appointment", async (req, res) => {
  try {
    const { applicationNumber, appointmentDate, testTrack } = req.body;

    if (!applicationNumber || !appointmentDate || !testTrack) {
      return res.status(400).send({ error: "Missing required fields" });
    }

    const collection = await db.collection("records");
    const result = await collection.updateOne(
      { applicationNumber },
      { $set: { appointmentDate, testTrack } }
    );

    if (result.matchedCount === 0) {
      return res.status(404).send({ error: "Application not found" });
    }

    res.status(200).send({ message: "Appointment booked successfully" });
  } catch (err) {
    console.error(err);
    res.status(500).send("Error booking appointment");
  }
});

// Correct export of the router
module.exports = router;
```

server.js

```
const express = require('express');
const mongodb = require('mongodb');
const cors = require('cors');
const bodyParser = require('body-parser');
const mongoose = require('mongoose');
const testRoutes = require('./testRoutes.js'); // Import your router
const records =require('./routes/record.js');
const app = express();
const PORT = 5000;
const MongoClient = mongodb.MongoClient;

app.use(cors());
app.use(bodyParser.json());

const url = 'mongodb://localhost:27017'; // MongoDB URL
const dbName = 'drivingLicenceDB'; // Database name
let db;

// Connect to MongoDB
MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true })
  .then(client => {
    console.log('Connected to Database');
    db = client.db(dbName);
  })
  .catch(error => console.error('Connection Error:', error));

// API to save application data
app.post('/api/save-application', async (req, res) => {
  try {
    const applicationData = req.body;
    console.log('Hooray! ');
    console.log(applicationData);
    const result = await db.collection('applications').insertOne(applicationData); //
Collection created if it doesn't exist
    res.status(201).json({ message: 'Application saved', id: result.insertedId });
  } catch (error) {
    console.error('Error inserting application:', error);
    res.status(500).json({ error: 'Failed to save application' });
  }
});
```

```
app.get('/api/get-application', async(req, res) => {
  try {
    console.log('Hooray Get! ');
    const name = req.query.applicationId;
    console.log(name);
    const result = await db.collection('applications').findOne({fullName:name}); //
Collection created if it doesn't exist
    console.log(result);
    res.status(201).json(result);
  } catch (error) {
    console.error('Error finding application:', error);
    res.status(500).json({ error: 'Failed to find application' });
  }

});
app.use('/api/tests', testRoutes);

// MongoDB connection
mongoose
 .connect('mongodb://127.0.0.1:27017/mern_test_db', {
  useNewUrlParser: true,
  useUnifiedTopology: true,
 })
 .then(() => console.log('Connected to MongoDB'))
 .catch((error) => console.error('MongoDB connection error:', error));

app.use(express.json());
app.use("/record", records);
// Start the server
app.listen(PORT, () => {
  console.log(`Server running on http://localhost:${PORT}`);
});
```

testmodel.js

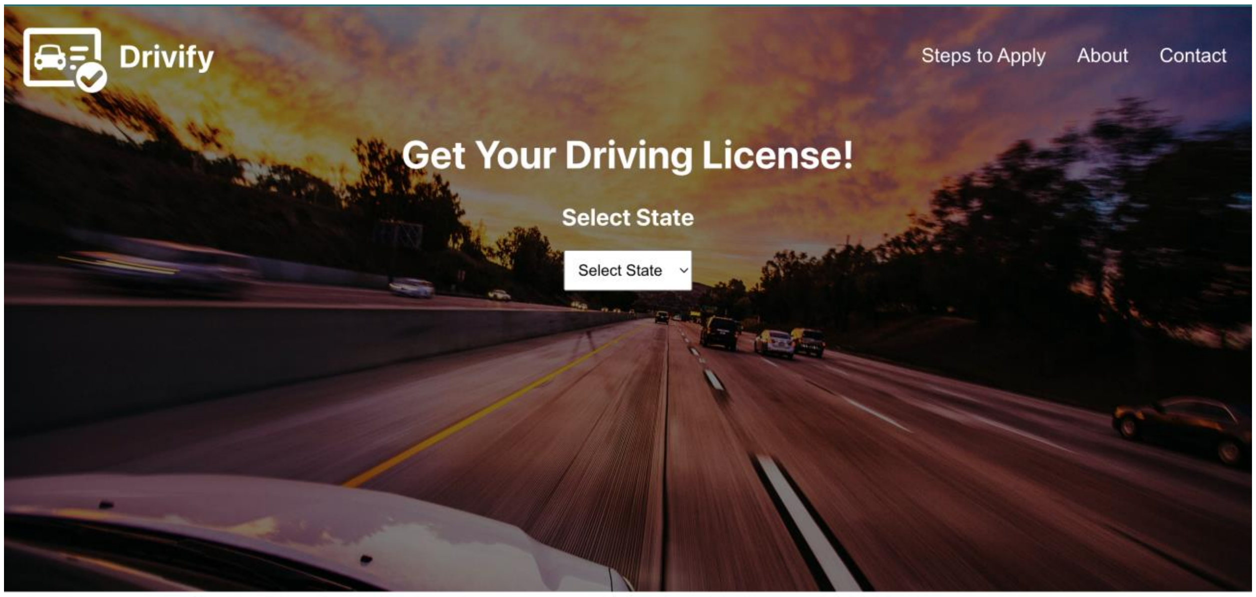
```
const mongoose = require('mongoose');

const testSchema = new mongoose.Schema({
 applicationId: {
  type: String,
  required: true,
 },
 answers: {
  type: Array,
  required: true,
 },
 createdAt: {
  type: Date,
  default: Date.now,
 },
});

const Test = mongoose.model('Test', testSchema);
 module.exports = Test;
```

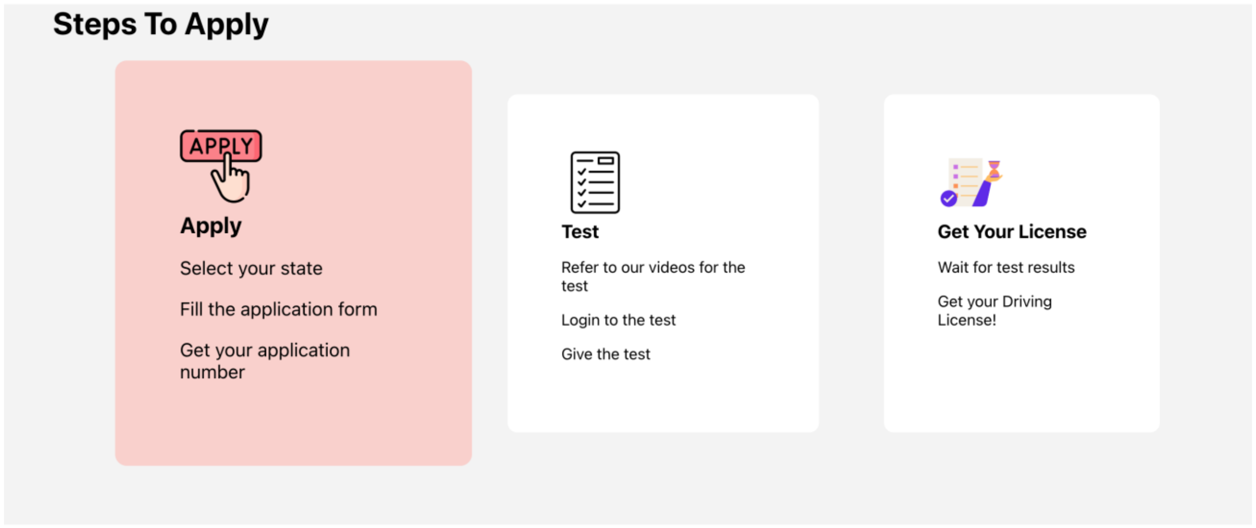
testRoutes.js

```
const express = require('express');
const router = express.Router(); // Initialize the router
const Test = require('./testmodel.js'); // Import your Mongoose model if needed

// Route to handle test submissions
router.post('/submit', async (req, res) => {
 try {
  const { applicationId, answers } = req.body;

  if (!applicationId || !answers) {
   return res.status(400).json({ message: 'Application ID and answers are required' });
  }

  // Save the test data to the database (if using MongoDB)
  const testSubmission = new Test({
   applicationId,
   answers,
  });

  await testSubmission.save();
  res.status(200).json({ message: 'Test submitted successfully' });
 } catch (error) {
  console.error(error);
  res.status(500).json({ message: 'An error occurred while submitting the test' });
 }
});
```

```
module.exports = router; // Export the router
```

Screenshots of the output





Steps To Apply

Apply

Select your state

Fill the application form

Get your application number

Test

Refer to our videos for the test

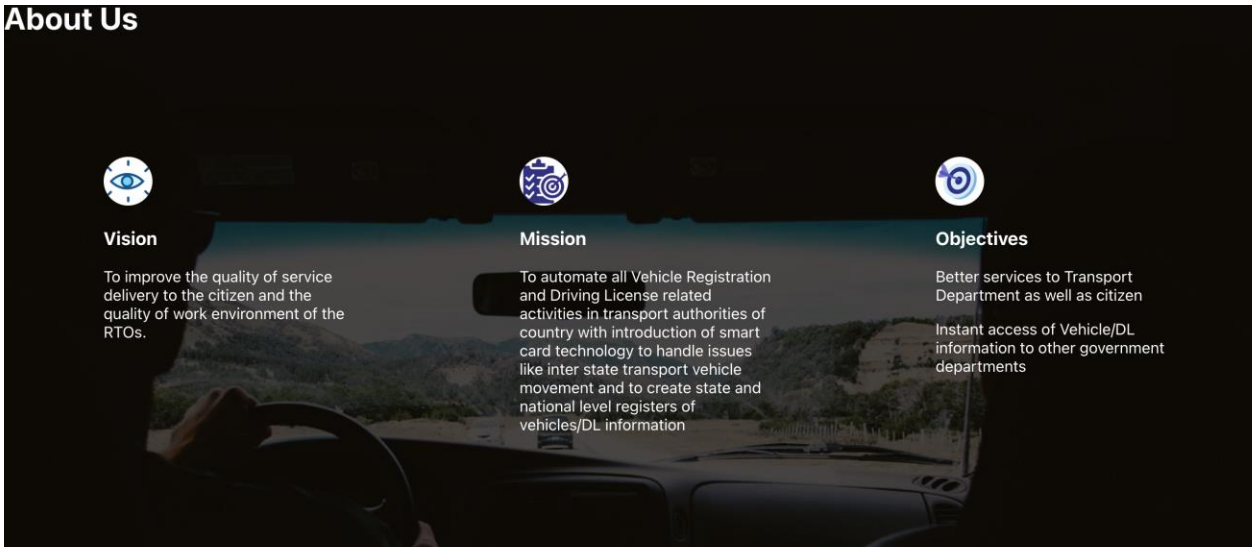
Login to the test

Give the test

Get Your License

Wait for test results

Get your Driving License!



About Us

Vision

To improve the quality of service delivery to the citizen and the quality of work environment of the RTOs.

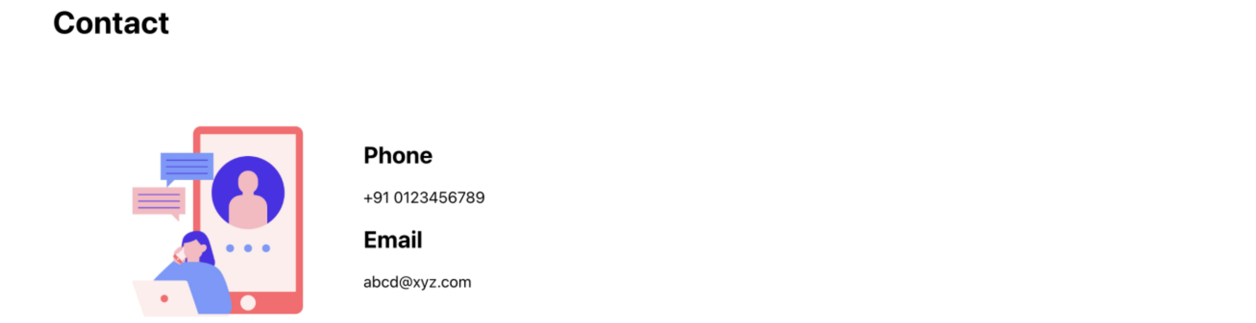
Mission

To automate all Vehicle Registration and Driving License related activities in transport authorities of country with introduction of smart card technology to handle issues like inter state transport vehicle movement and to create state and national level registers of vehicles/DL information

Objectives

Better services to Transport Department as well as citizen

Instant access of Vehicle/DL information to other government departments



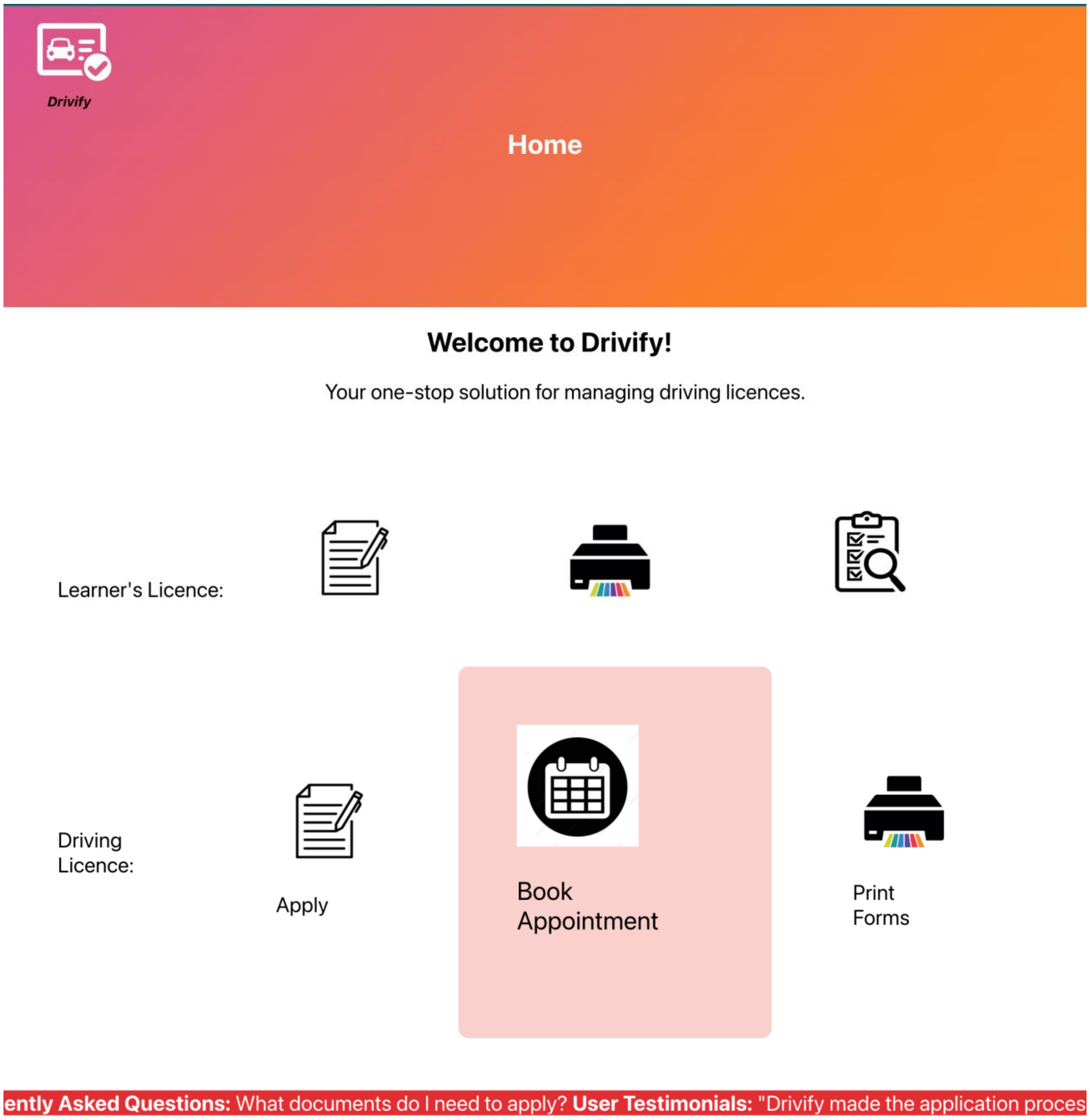
Contact

Phone

+91 0123456789

Email

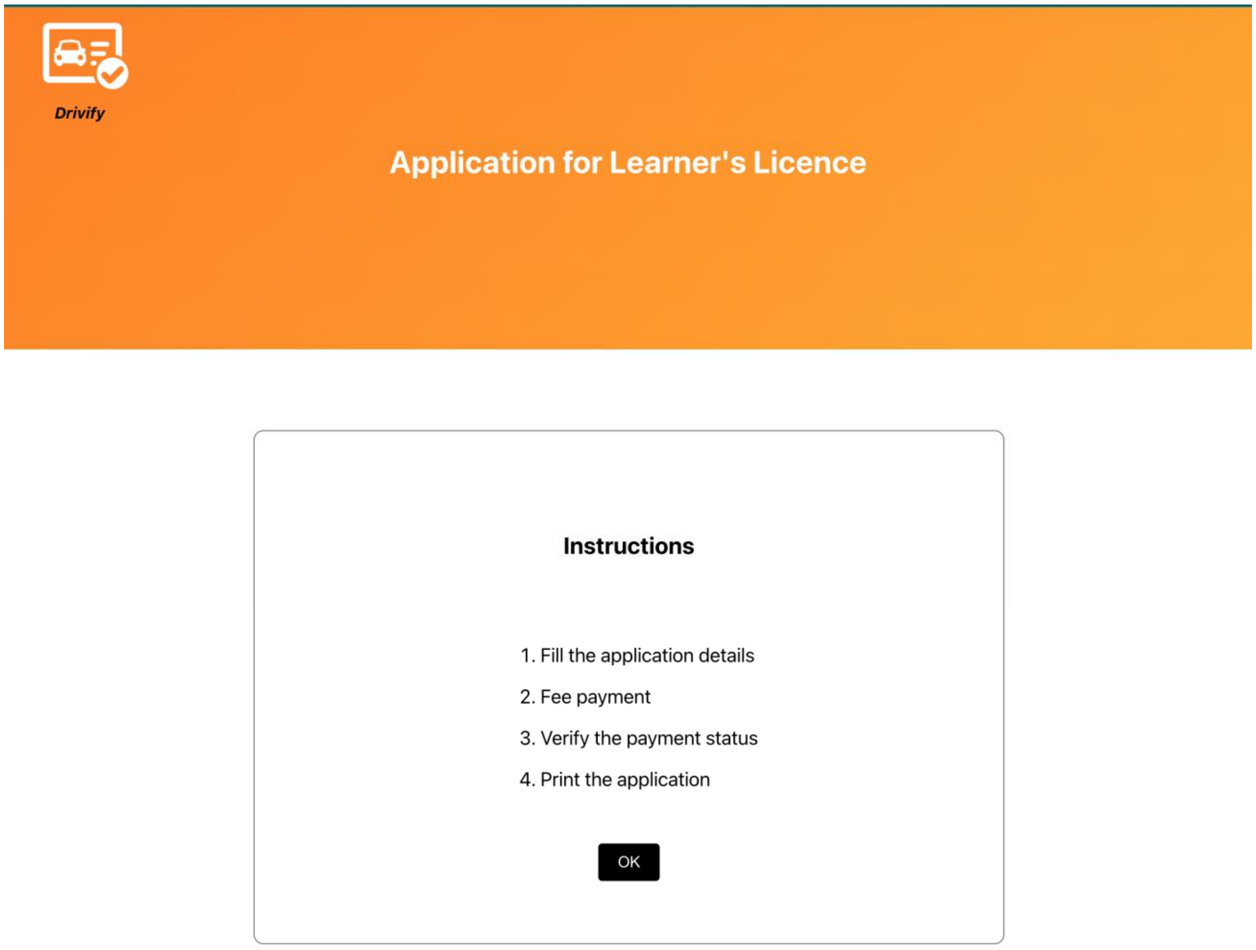
abcd@xyz.com



Welcome to Drivify!

Your one-stop solution for managing driving licences.

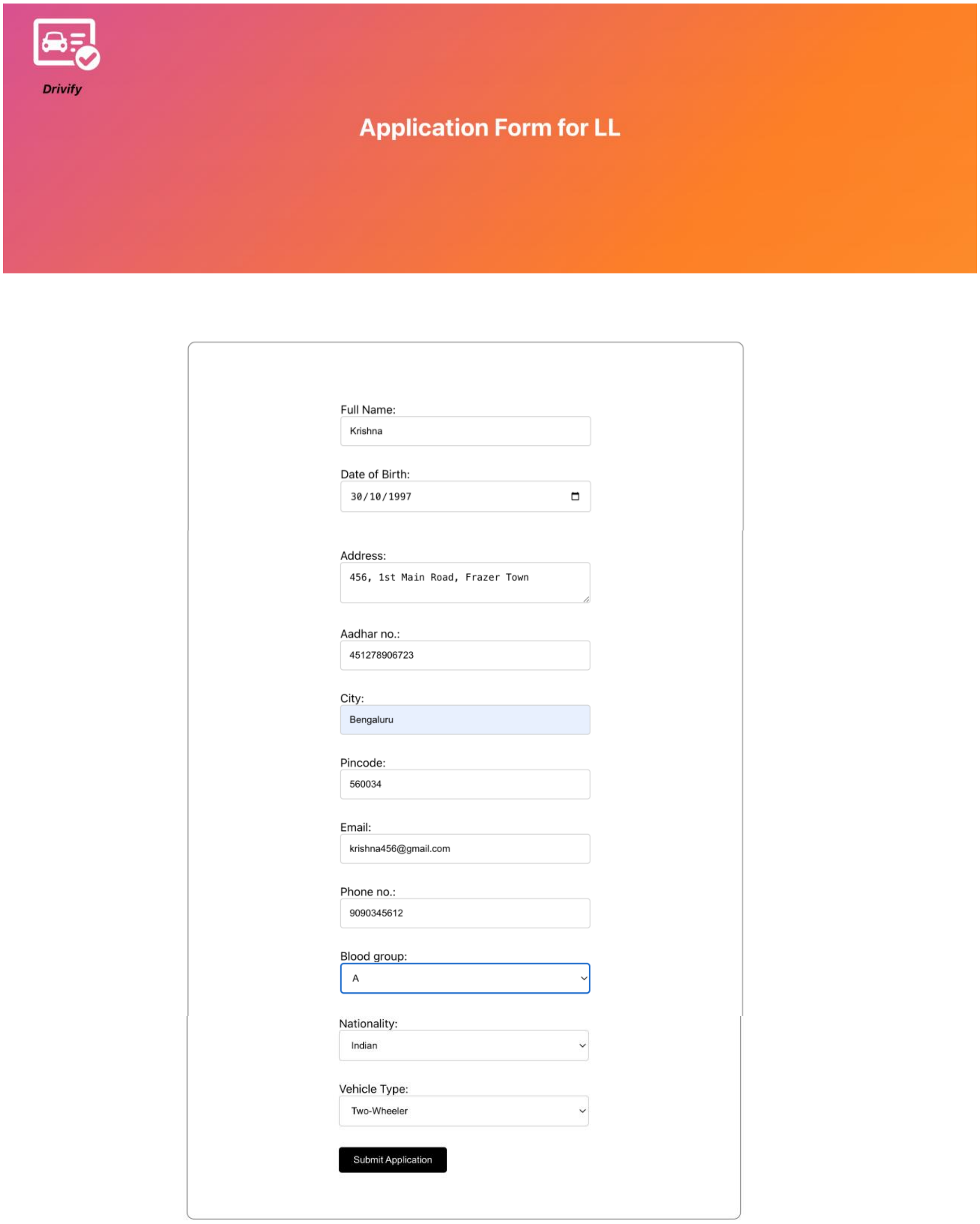
ently Asked Questions: What documents do I need to apply? User Testimonials: "Drivify made the application proces



Application for Learner's Licence

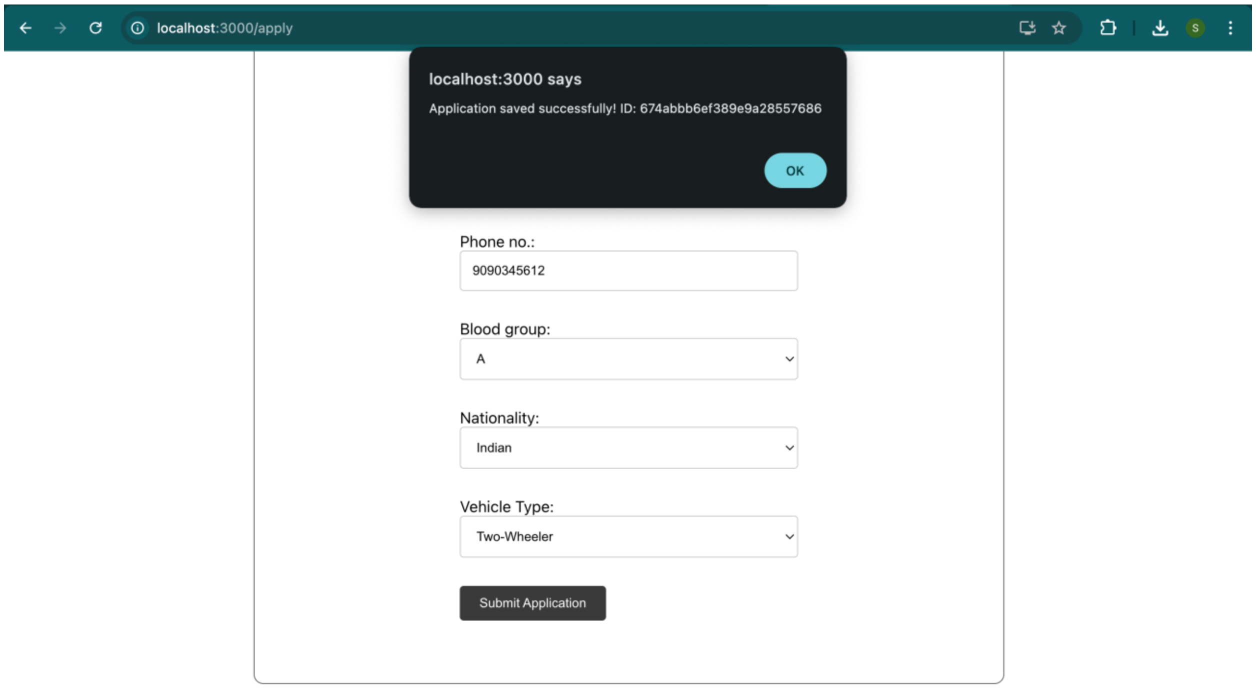
Instructions

1. Fill the application details
2. Fee payment
3. Verify the payment status
4. Print the application



Application Form for LL

Full Name:	Krishna
Date of Birth:	30/10/1997
Address:	456, 1st Main Road, Frazer Town
Aadhar no.:	451278906723
City:	Bengaluru
Pincode:	560034
Email:	krishna456@gmail.com
Phone no.:	9090345612
Blood group:	A
Nationality:	Indian
Vehicle Type:	Two-Wheeler



localhost:3000/apply

localhost:3000 says

Application saved successfully! ID: 674abbb6ef389e9a28557686

OK

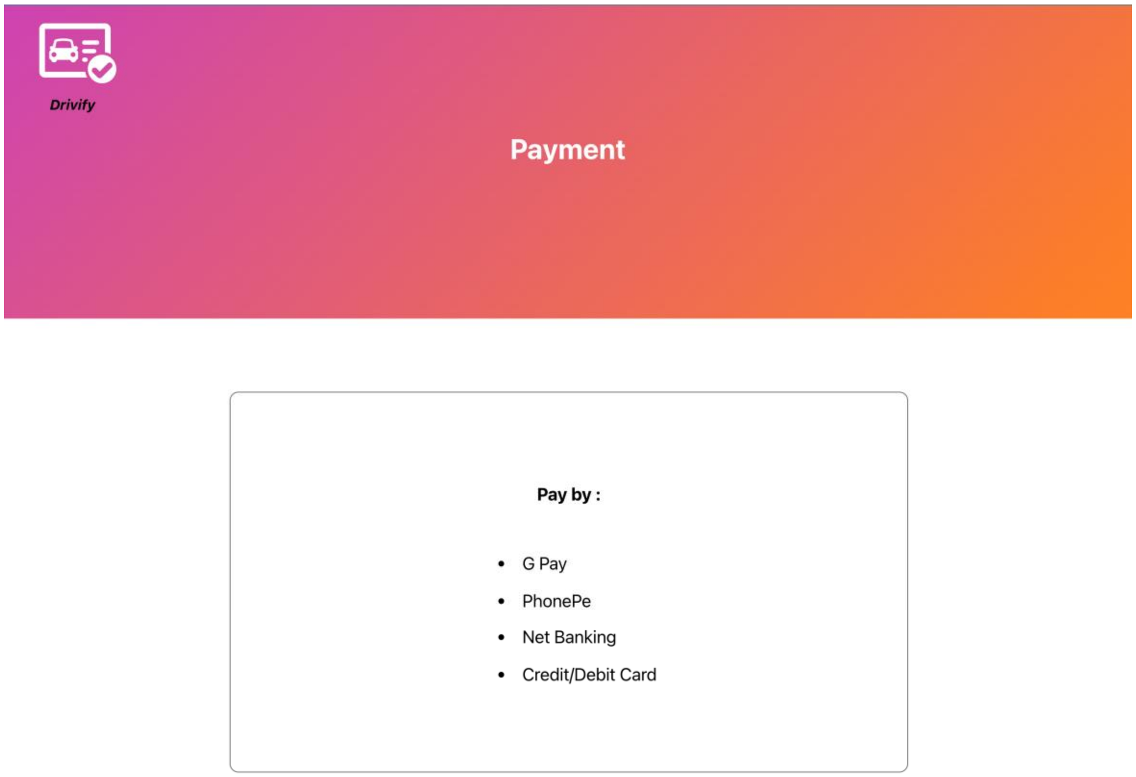
Phone no.: 9090345612

Blood group: A

Nationality: Indian

Vehicle Type: Two-Wheeler

Submit Application

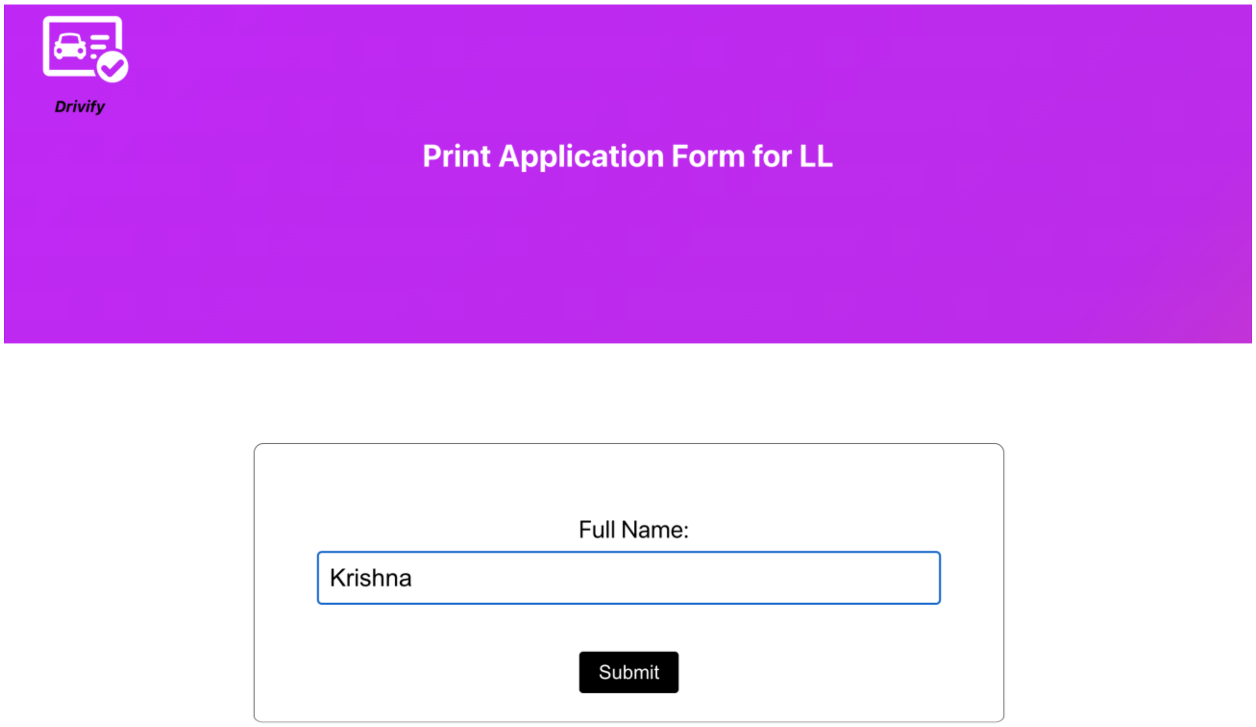


Drivify

Payment

Pay by :

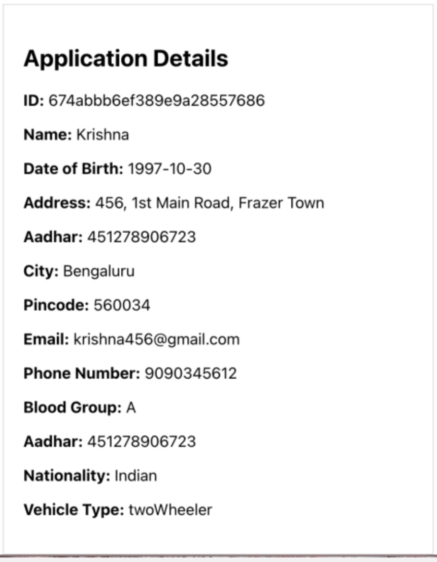
- G Pay
- PhonePe
- Net Banking
- Credit/Debit Card


Drivify

Print Application Form for LL

Full Name:

Krishna



Application Details

ID: 674abbb6ef389e9a28557686

Name: Krishna

Date of Birth: 1997-10-30

Address: 456, 1st Main Road, Frazer Town

Aadhar: 451278906723

City: Bengaluru

Pincode: 560034

Email: krishna456@gmail.com

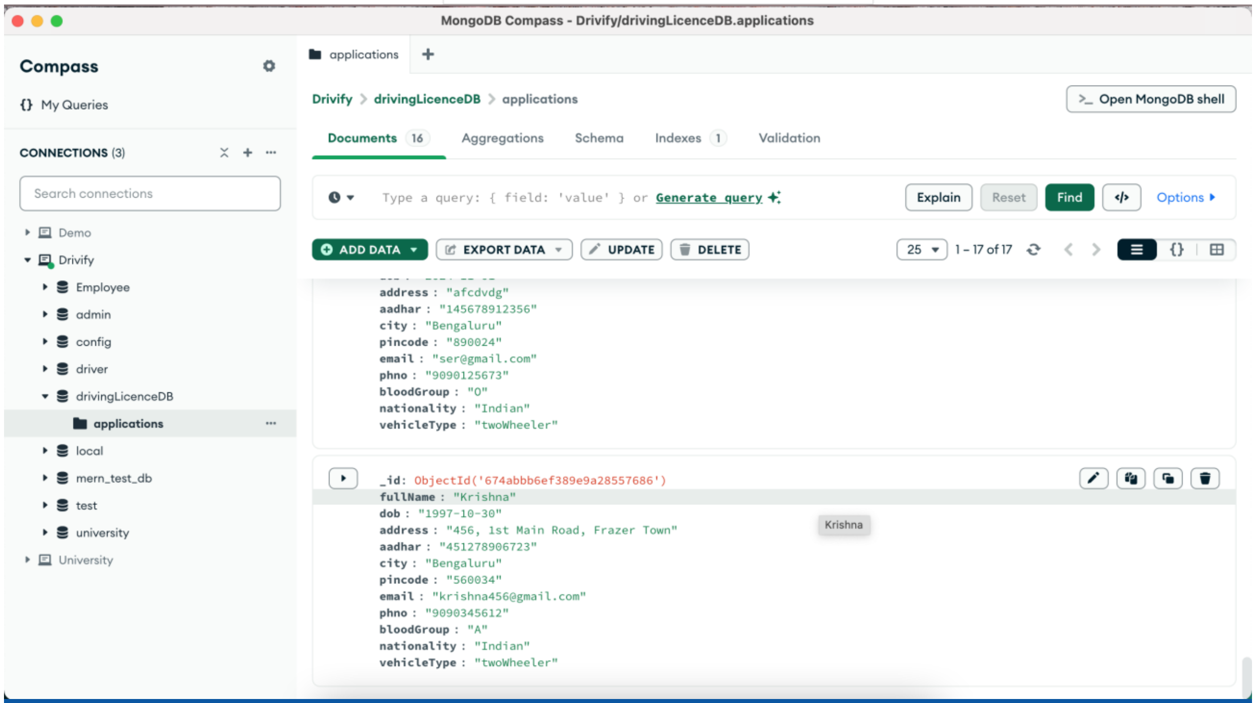
Phone Number: 9090345612

Blood Group: A

Aadhar: 451278906723

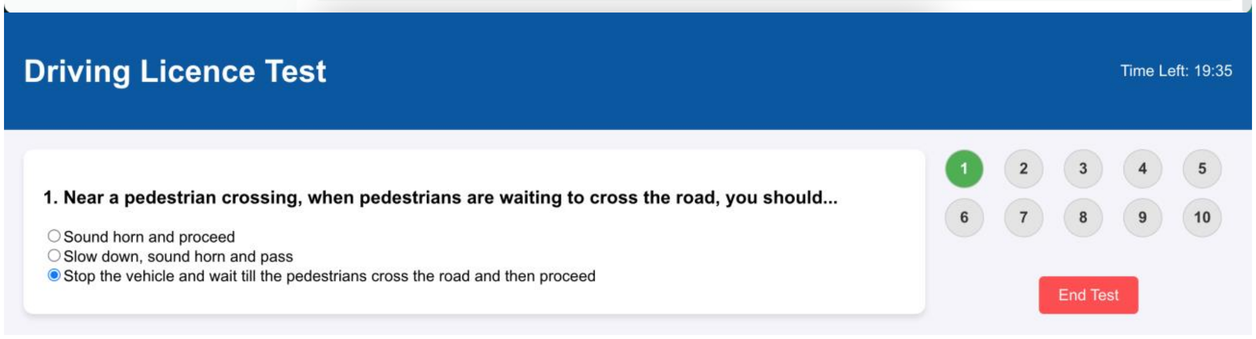
Nationality: Indian

Vehicle Type: twoWheeler



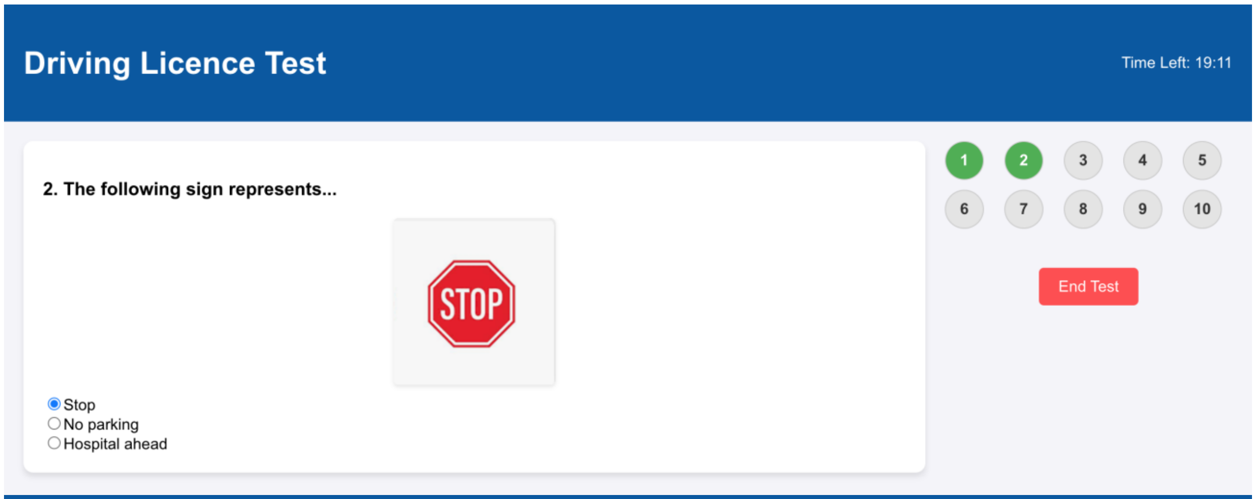
```
address : "afcdvdg"
aadhar : "145678912356"
city : "Bengaluru"
pincode : "890024"
email : "ser@gmail.com"
phno : "9090125673"
bloodGroup : "O"
nationality : "Indian"
vehicleType : "twoWheeler"
```

```
_id: ObjectId('674abbb6ef389e9a28557686')
fullName : "Krishna"
dob : "1997-10-30"
address : "456, 1st Main Road, Frazer Town"
aadhar : "451278906723"
city : "Bengaluru"
pincode : "560034"
email : "krishna456@gmail.com"
phno : "9090345612"
bloodGroup : "A"
nationality : "Indian"
vehicleType : "twoWheeler"
```



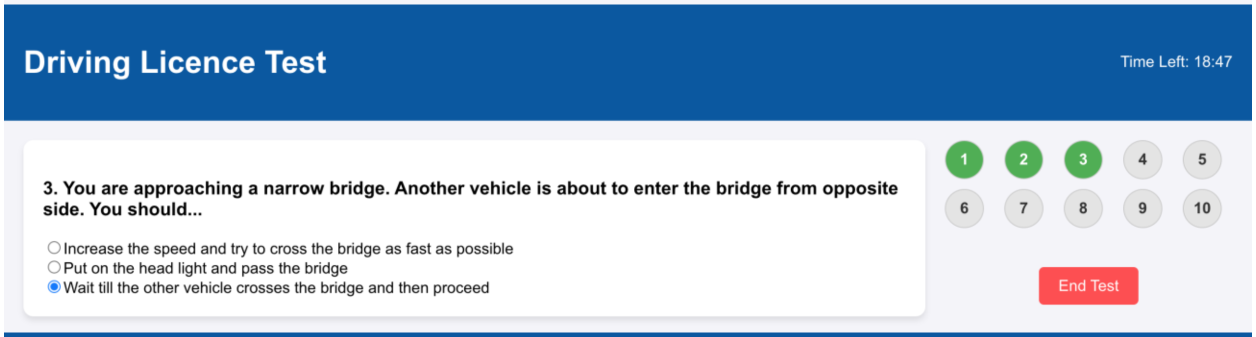
1. Near a pedestrian crossing, when pedestrians are waiting to cross the road, you should...

Sound horn and proceed
Slow down, sound horn and pass
[✓ Stop the vehicle and wait till the pedestrians cross the road and then proceed]



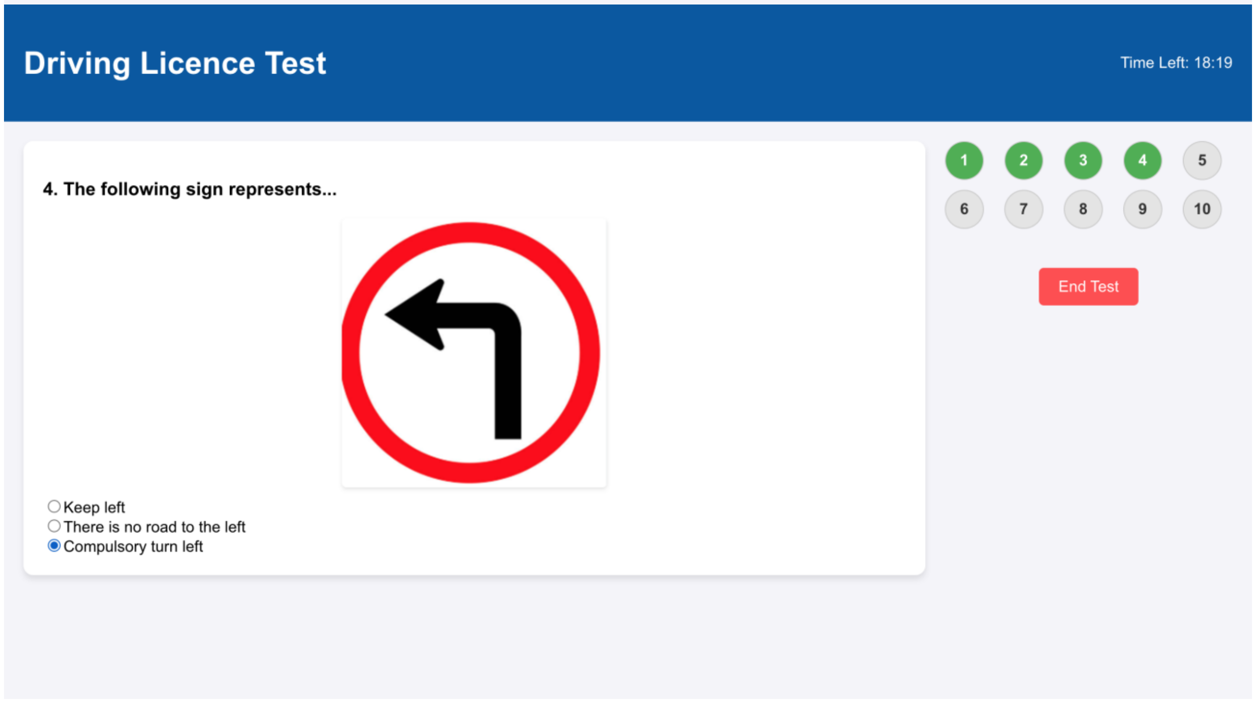
2. The following sign represents...

[✓ Stop]
No parking
Hospital ahead



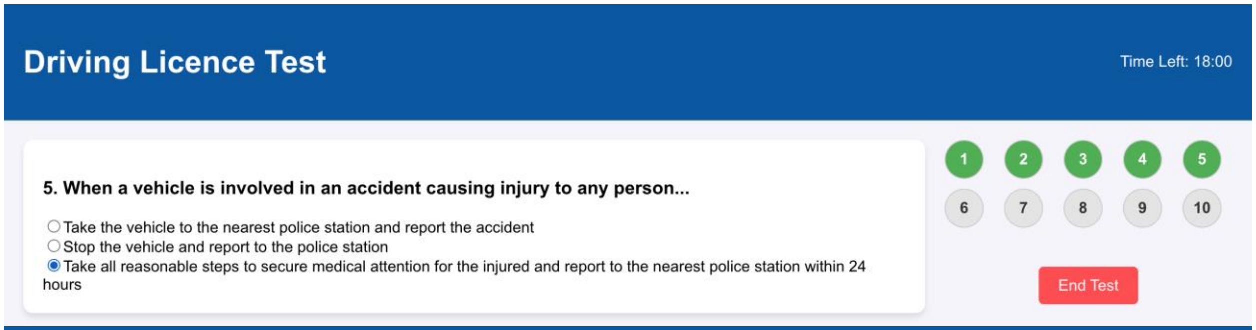
3. You are approaching a narrow bridge. Another vehicle is about to enter the bridge from opposite side. You should...

Increase the speed and try to cross the bridge as fast as possible
Put on the head light and pass the bridge
[✓ Wait till the other vehicle crosses the bridge and then proceed]



4. The following sign represents...

Keep left
There is no road to the left
[✓ Compulsory turn left]

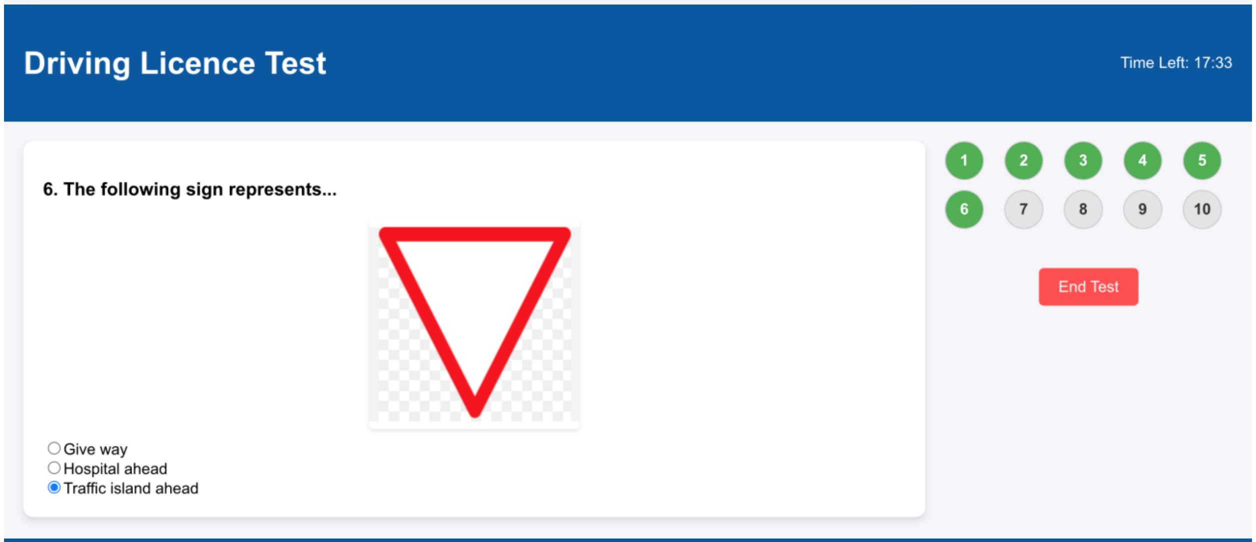


Driving Licence Test

Time Left: 18:00

5. When a vehicle is involved in an accident causing injury to any person...

- ○ Take the vehicle to the nearest police station and report the accident
- ○ Stop the vehicle and report to the police station
- [✓ Take all reasonable steps to secure medical attention for the injured and report to the nearest police station within 24 hours]

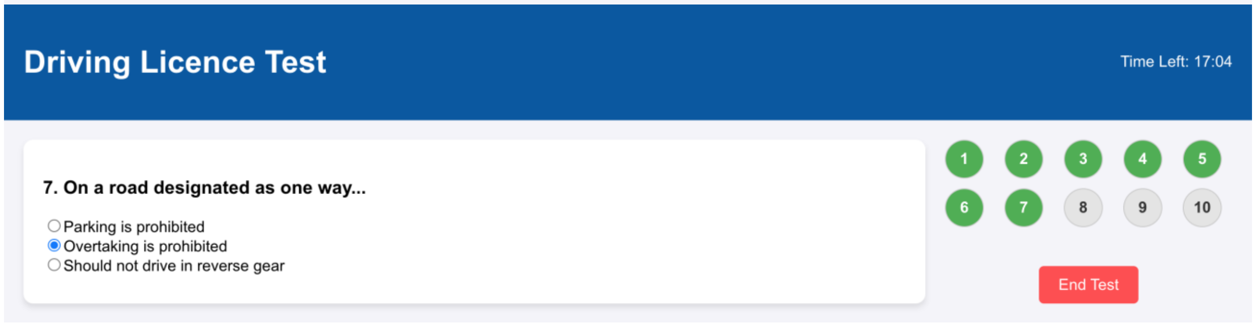


Driving Licence Test

Time Left: 17:33

6. The following sign represents...

- ○ Give way
- ○ Hospital ahead
- [✓ Traffic island ahead]

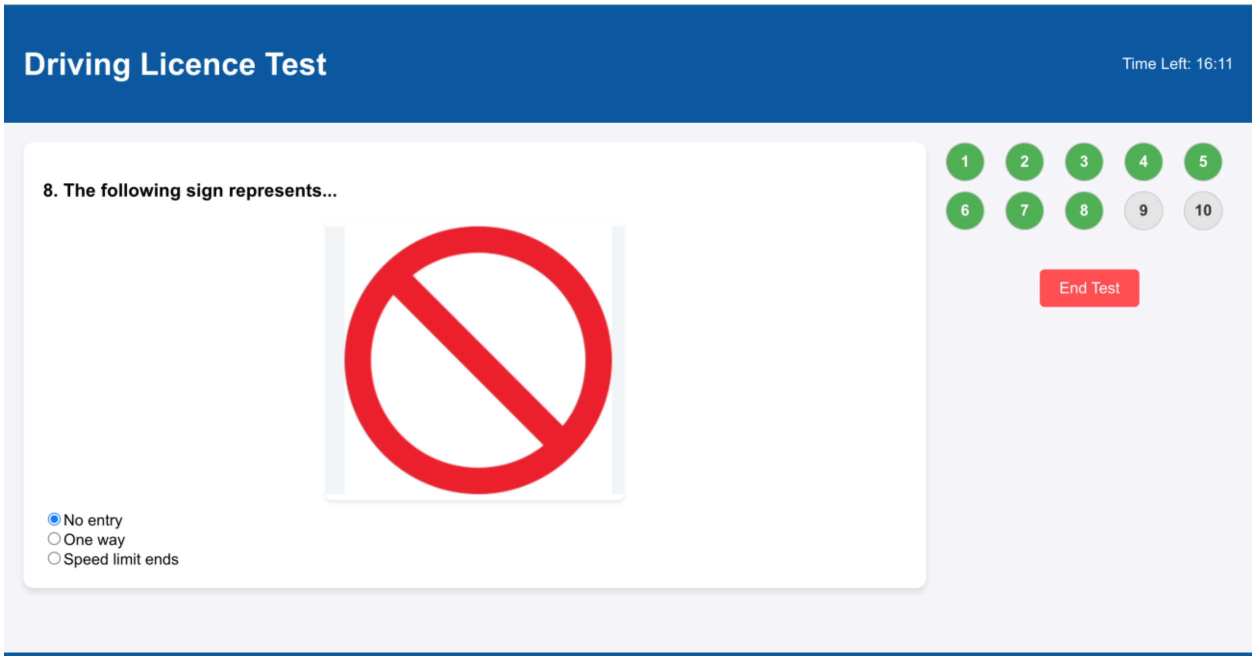


Driving Licence Test

Time Left: 17:04

7. On a road designated as one way...

- ○ Parking is prohibited
- [✓ Overtaking is prohibited]
- ○ Should not drive in reverse gear

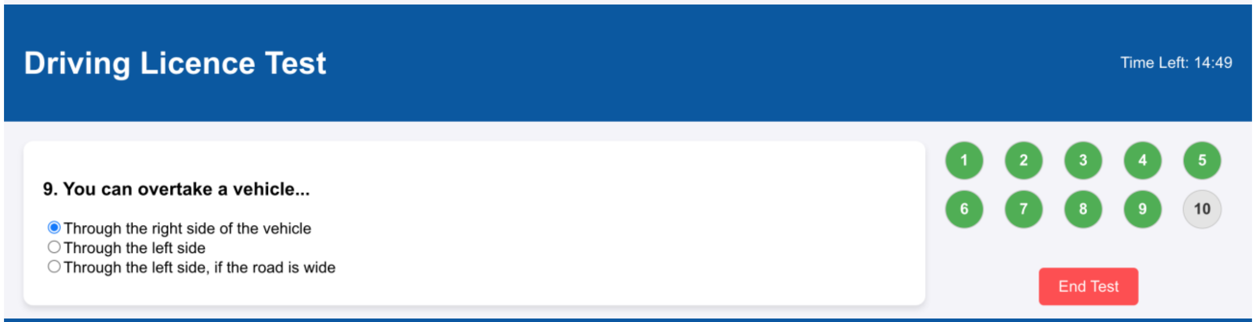


Driving Licence Test

Time Left: 16:11

8. The following sign represents...

[✓ No entry]
One way
Speed limit ends

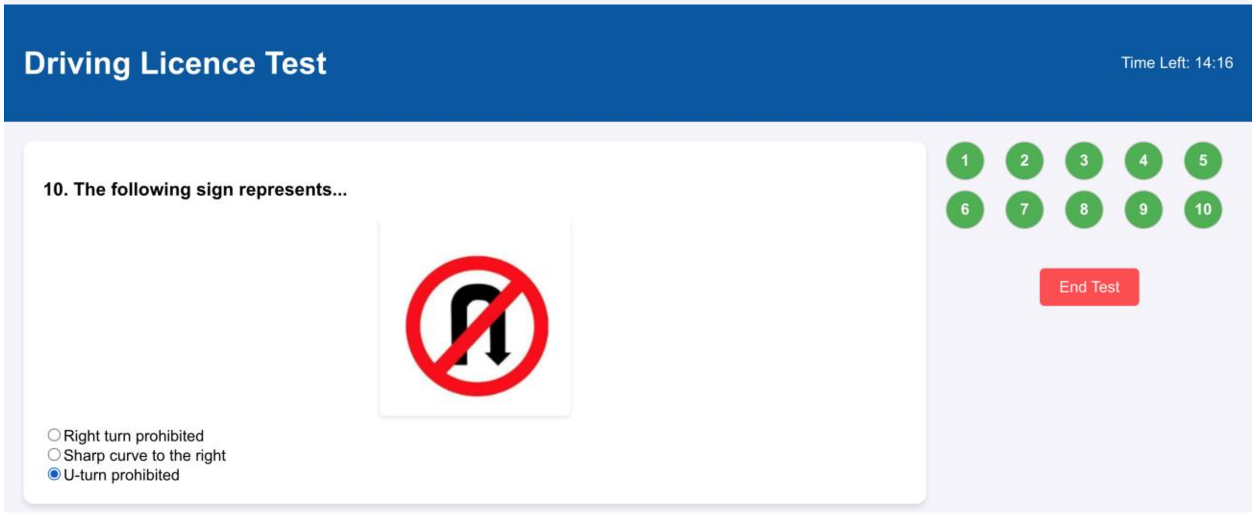


Driving Licence Test

Time Left: 14:49

9. You can overtake a vehicle...

[✓ Through the right side of the vehicle]
Through the left side
Through the left side, if the road is wide

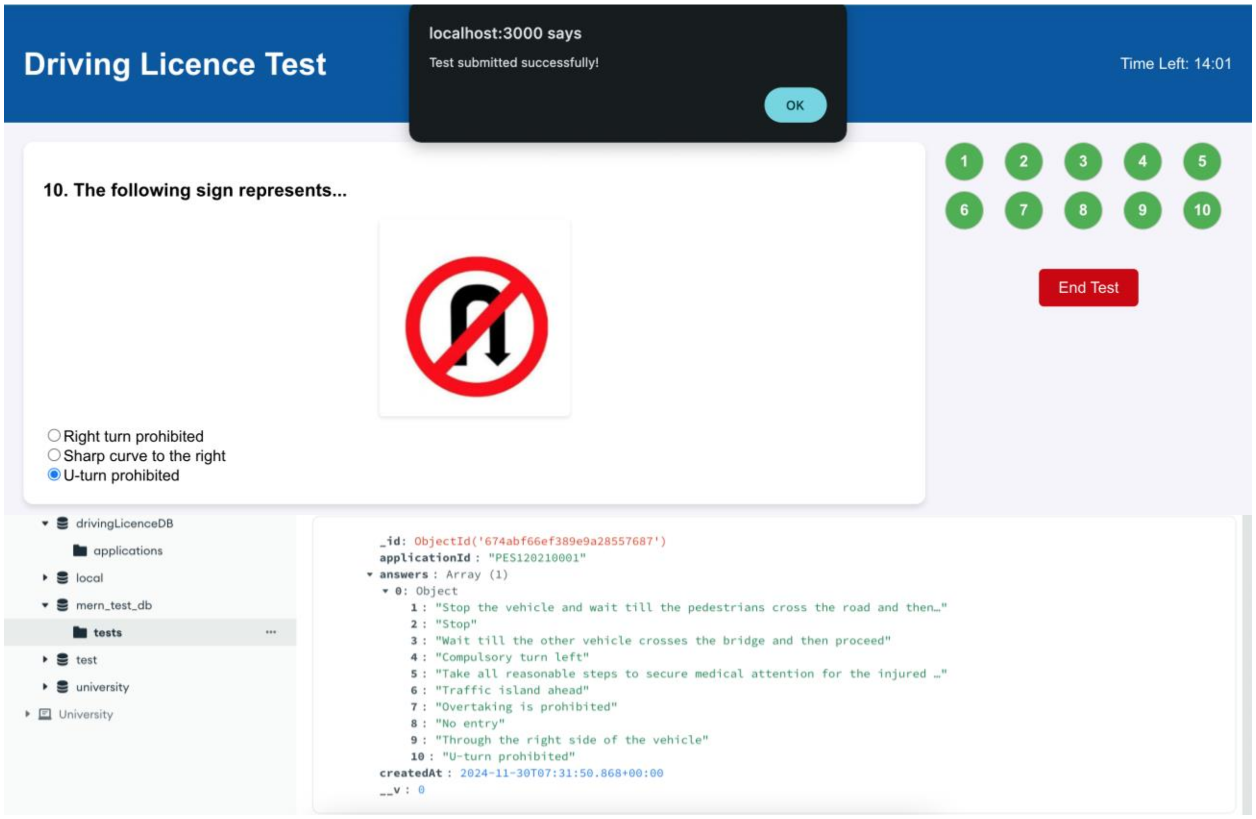


Driving Licence Test

Time Left: 14:16

10. The following sign represents...

Right turn prohibited
Sharp curve to the right
[✓ U-turn prohibited]



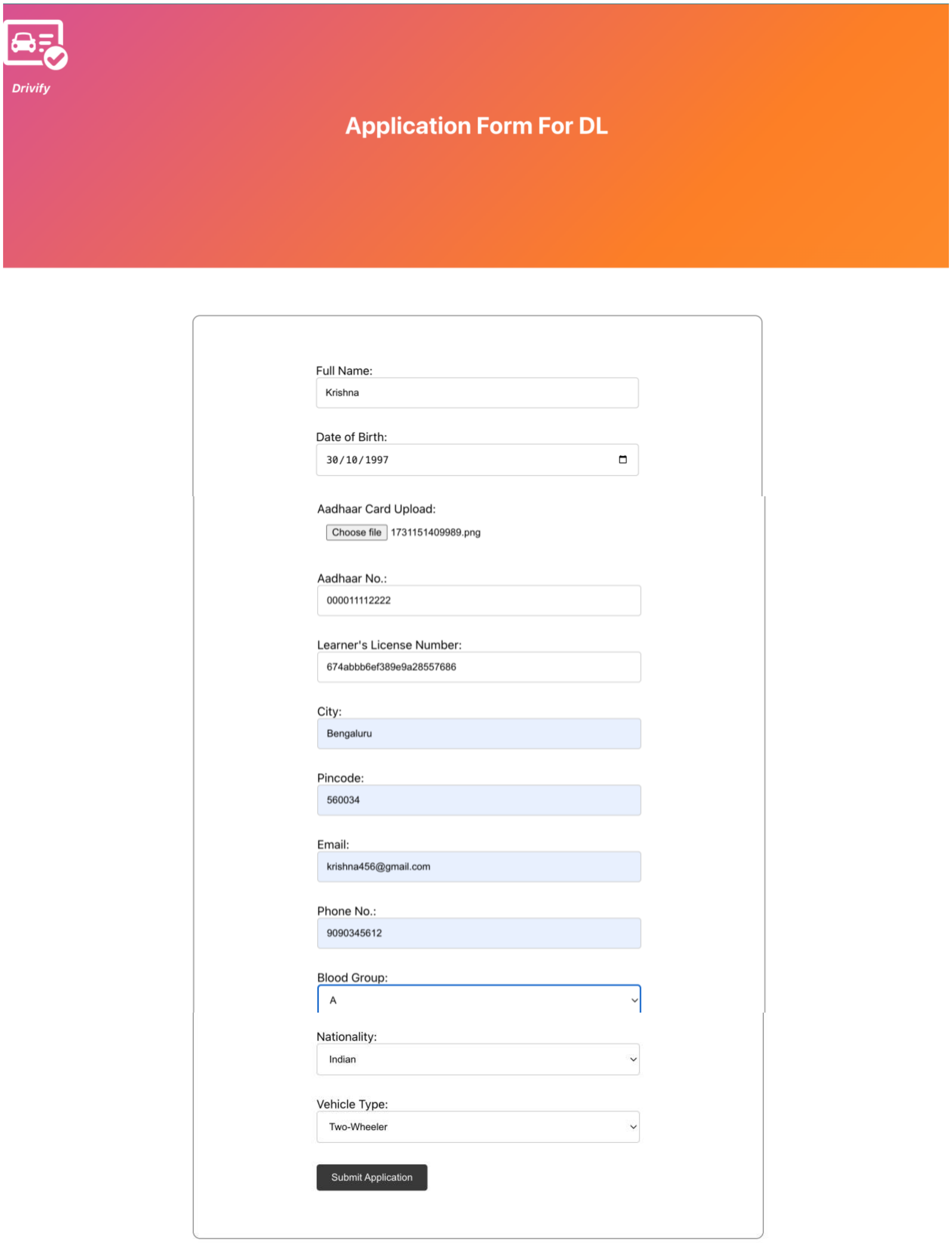
localhost:3000 says

Test submitted successfully!

10. The following sign represents...

- [ ] Right turn prohibited
- [ ] Sharp curve to the right
- [✓] U-turn prohibited

```
    _id: ObjectId('674abf66ef389e9a28557687')
    applicationId : "PES120210001"
  ▾ answers : Array (1)
    ▾ 0: Object
        1 : "Stop the vehicle and wait till the pedestrians cross the road and then…"
        2 : "Stop"
        3 : "Wait till the other vehicle crosses the bridge and then proceed"
        4 : "Compulsory turn left"
        5 : "Take all reasonable steps to secure medical attention for the injured …"
        6 : "Traffic island ahead"
        7 : "Overtaking is prohibited"
        8 : "No entry"
        9 : "Through the right side of the vehicle"
        10 : "U-turn prohibited"
    createdAt : 2024-11-30T07:31:50.868+00:00
    __v : 0
```



Drivify

Application Form For DL

Full Name: Krishna

Date of Birth: 30/10/1997

Aadhaar Card Upload: Choose file 1731151409989.png

Aadhaar No.: 000011112222

Learner's License Number: 674abbb6ef389e9a28557686

City: Bengaluru

Pincode: 560034

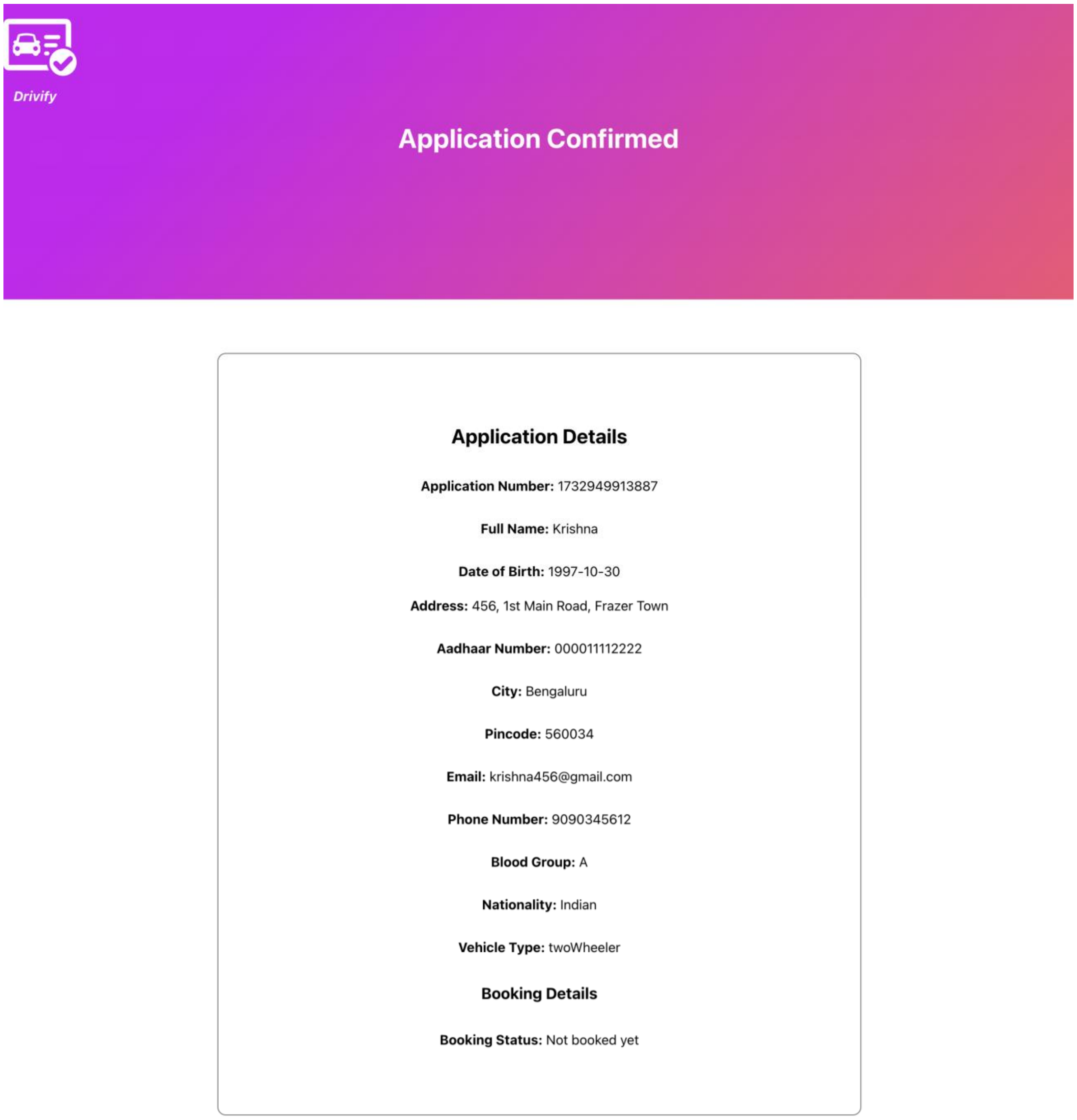
Email: krishna456@gmail.com

Phone No.: 9090345612

Blood Group: A

Nationality: Indian

Vehicle Type: Two-Wheeler



Application Confirmed

Application Details

Application Number: 1732949913887

Full Name: Krishna

Date of Birth: 1997-10-30

Address: 456, 1st Main Road, Frazer Town

Aadhaar Number: 000011112222

City: Bengaluru

Pincode: 560034

Email: krishna456@gmail.com

Phone Number: 9090345612

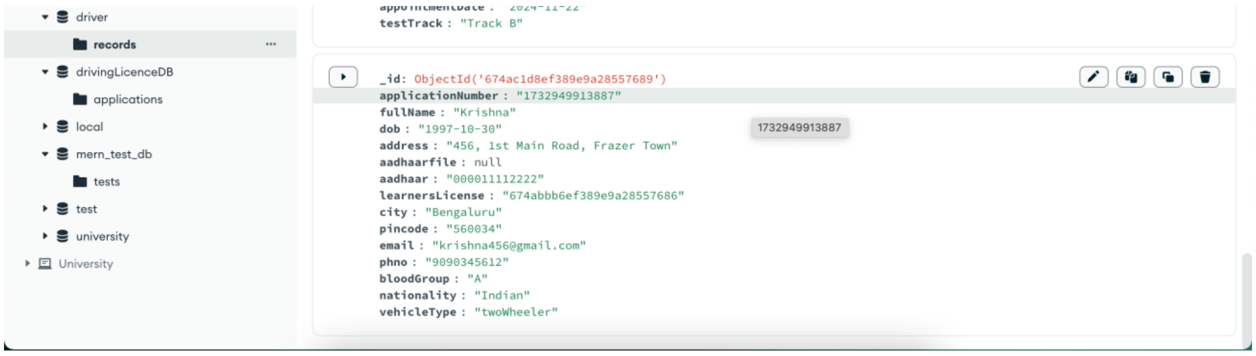
Blood Group: A

Nationality: Indian

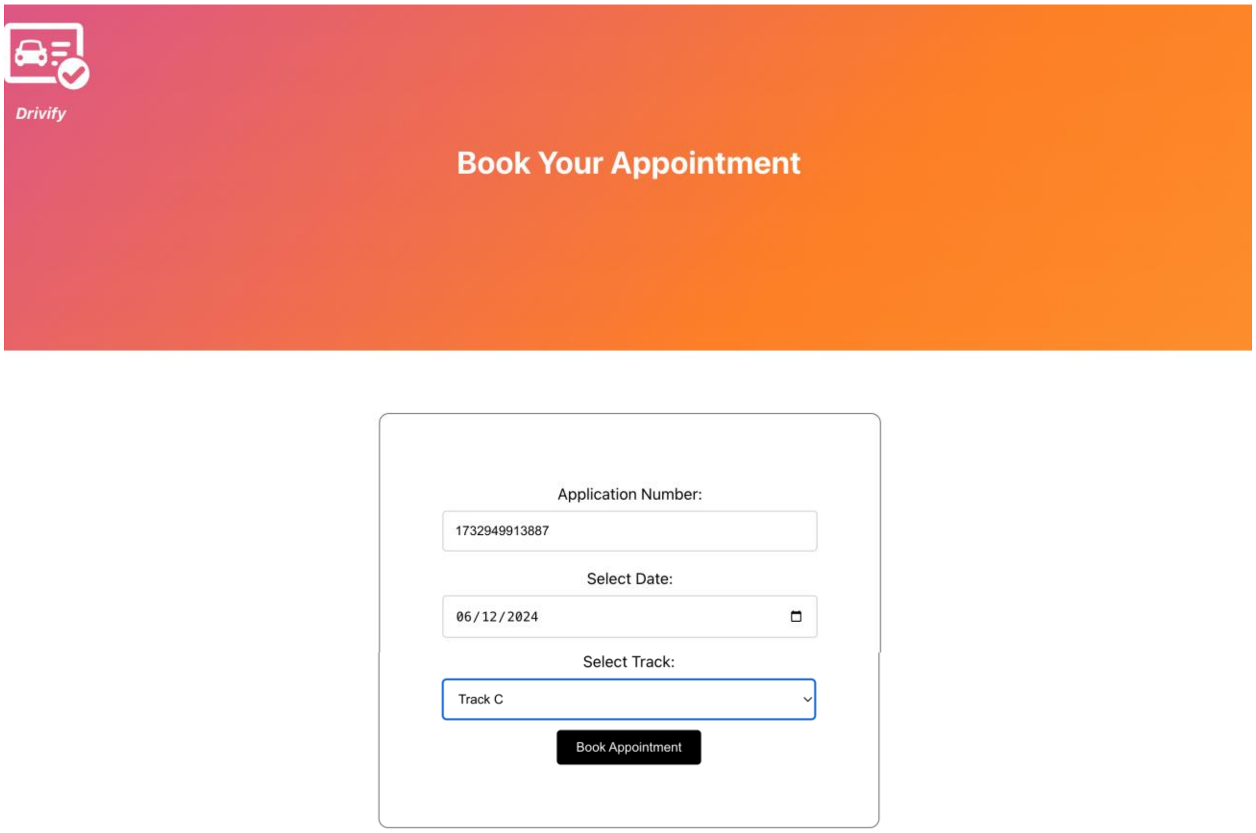
Vehicle Type: twoWheeler

Booking Details

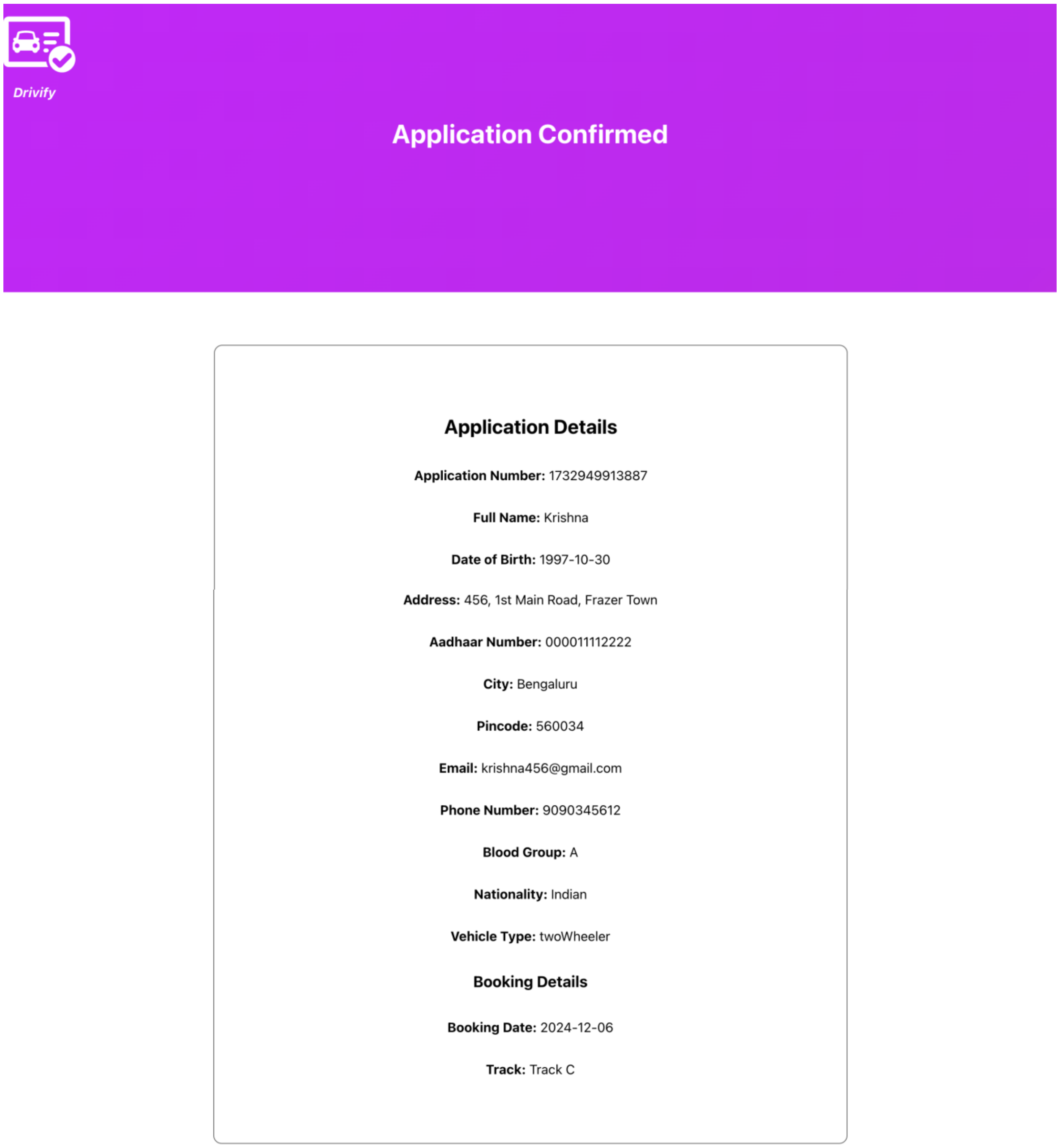
Booking Status: Not booked yet



```
appointmentDate : "2024-11-22"
testTrack : "Track B"

_id: ObjectId('674ac1d8ef389e9a28557689')
applicationNumber : "1732949913887"
fullName : "Krishna"
dob : "1997-10-30"
address : "456, 1st Main Road, Frazer Town"
aadhaarfile : null
aadhaar : "000011112222"
learnersLicense : "674abbb6ef389e9a28557686"
city : "Bengaluru"
pincode : "560034"
email : "krishna456@gmail.com"
phno : "9090345612"
bloodGroup : "A"
nationality : "Indian"
vehicleType : "twoWheeler"
```



Application Number:	1732949913887
Select Date:	06/12/2024
Select Track:	Track C



Application Confirmed

Application Details

Application Number: 1732949913887

Full Name: Krishna

Date of Birth: 1997-10-30

Address: 456, 1st Main Road, Frazer Town

Aadhaar Number: 000011112222

City: Bengaluru

Pincode: 560034

Email: krishna456@gmail.com

Phone Number: 9090345612

Blood Group: A

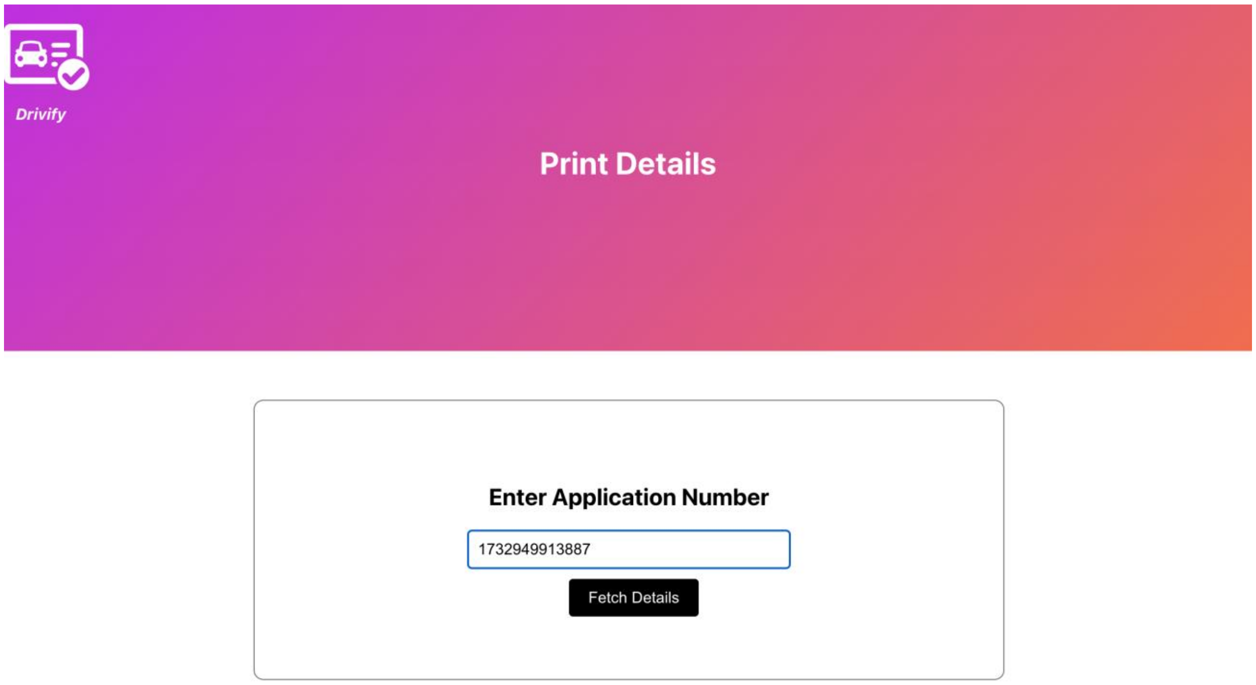
Nationality: Indian

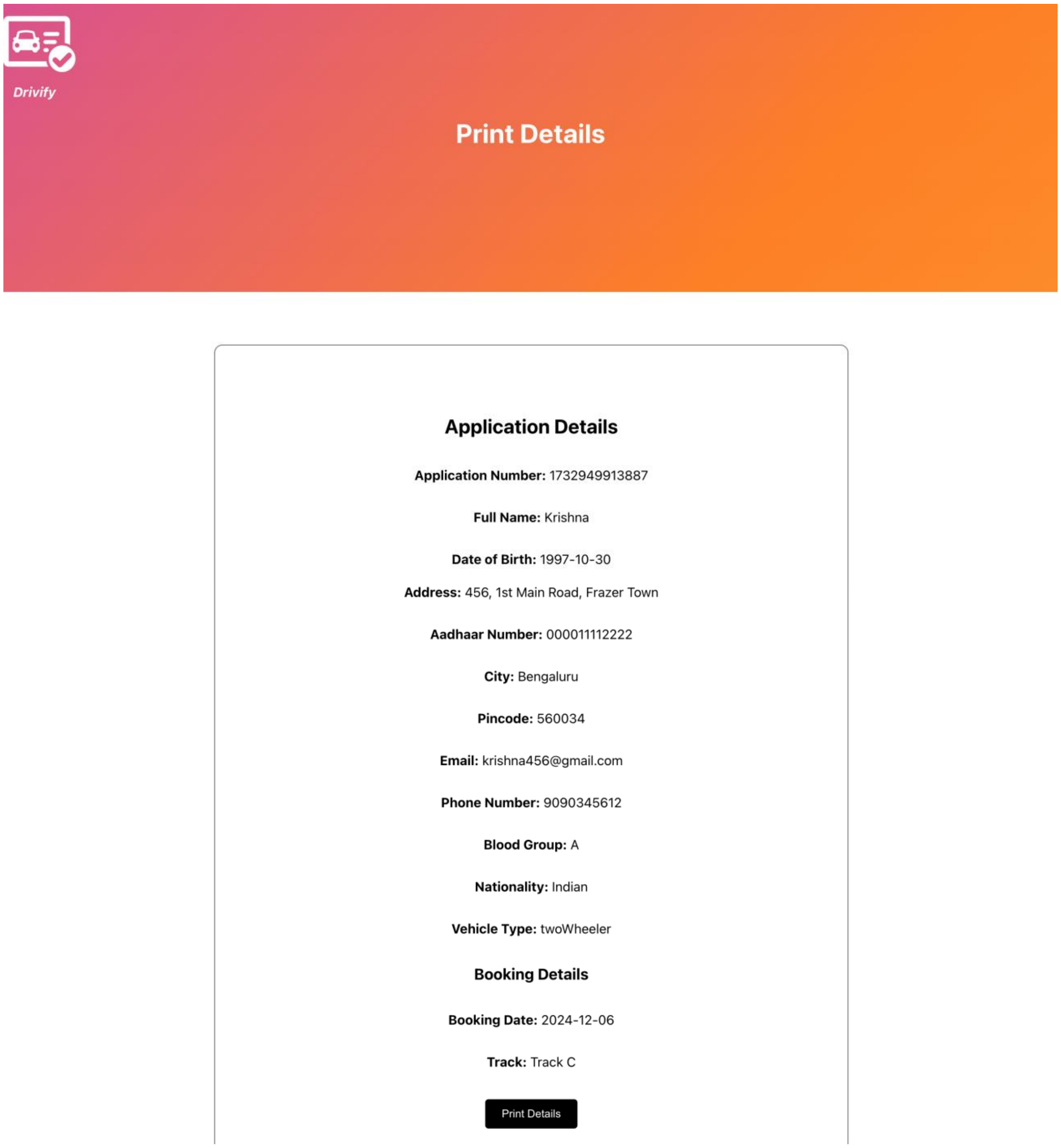
Vehicle Type: twoWheeler

Booking Details

Booking Date: 2024-12-06

Track: Track C


Drivify
Print Details
Enter Application Number
1732949913887
Fetch Details



Application Details

Application Number: 1732949913887

Full Name: Krishna

Date of Birth: 1997-10-30

Address: 456, 1st Main Road, Frazer Town

Aadhaar Number: 000011112222

City: Bengaluru

Pincode: 560034

Email: krishna456@gmail.com

Phone Number: 9090345612

Blood Group: A

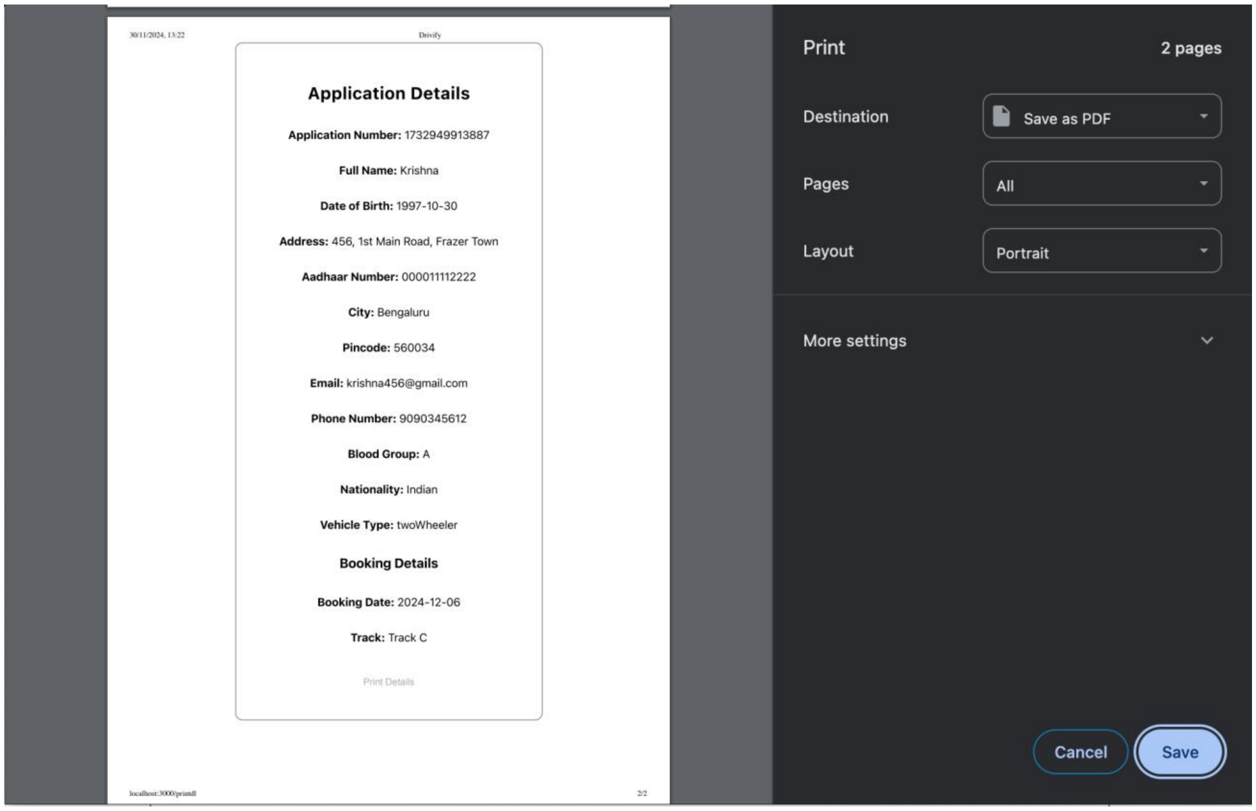
Nationality: Indian

Vehicle Type: twoWheeler

Booking Details

Booking Date: 2024-12-06

Track: Track C



30/11/2024, 13:22 Drivify

Application Details

Application Number: 1732949913887

Full Name: Krishna

Date of Birth: 1997-10-30

Address: 456, 1st Main Road, Frazer Town

Aadhaar Number: 000011112222

City: Bengaluru

Pincode: 560034

Email: krishna456@gmail.com

Phone Number: 9090345612

Blood Group: A

Nationality: Indian

Vehicle Type: twoWheeler

Booking Details

Booking Date: 2024-12-06

Track: Track C

Print Details

localhost:3000/printdl 2/2

Print 2 pages

Destination: Save as PDF

Pages: All

Layout: Portrait