Sri Lanka Institute of Information Technology

Submitted by

NAME	STUDENT REGISTRATION NUMBER
SARANGA R.G.V	IT20096298
JAYASINGHE A.H.N	IT20010058
DIVYANTHA U.A.S	IT20000462
WIJERATHNE D.M.R	IT19208022

Introduction

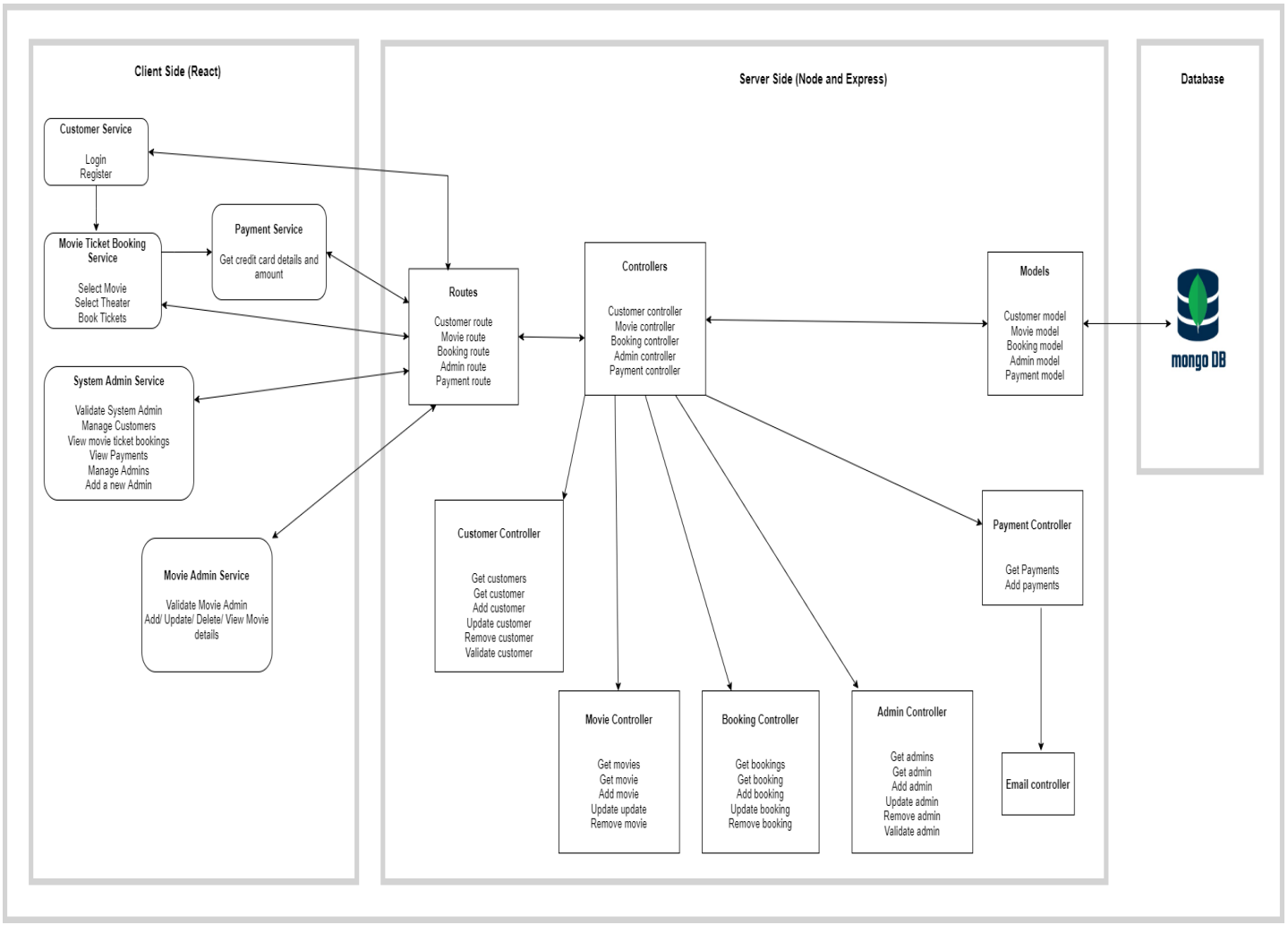
People are always seeking the easiest way to do something. Because of that nowadays all the things are available on online platforms. In our project also we got a requirement, to develop an online movie reservation platform. As the solution, we created a web application called “Movie App” using react js, node js, and mongo DB to fulfill the client requirement.

In this “Movie App” web application, first users have to register as a customer by providing the required details. After that, they can log in to the system and search for their favourite movies easily. Also, customers can book multiple tickets for one movie at a time. After their booking, those details are shown in the cart. In that interface, they can delete the booking if they need, or they can easily calculate the total price and make a credit card payment. Finally, they will get their confirmation email. That is the very easy process a customer has to face in this system.

The movies which are shown in the application to customers are added by the movie admin. That is the person who can add, update, and delete the movie details when it is necessary. Also, the system can have one or more movie admin.

The system admin is the one who can add a movie admin to the system, and he can update or delete details of a movie admin. Also, the system admin can delete a customer’s details if it is necessary. He has access to see all the payment details and booking details(cart) of the application

High level architecture diagram



Service interfaces

Movie Admin

- Firstly, the movie admin needs to log in to the system. If the user enters the wrong email and password, then we get an error message. If the user enters a valid username and password, then we get the successful message, and the user navigates to the movie management page.

- Movie admin login frontend code - we are giving the username and password from the user and pass to the relevant route in the backend through http POST request to validate. (Appendix [1]).

- Movie admin login backend code (authentication code)
- Server.js file in the backend. (appendix [2]).

- Imported relevant admin controllers. if get POST request to the "/validate" in then call validateAdmin() method in the admin controller file. (Appendix [3]).

- Get the relevant user details from the database which belong to the email input by the admin in the log in interface. Then decrypted the encrypted password. Then compare the database password with the password that is input from the frontend and if those passwords are matching then user validation is complete. (Appendix [4]).

- To get all movies we send a GET http request to the backend. To delete a specific movie record we send a DELETE http request to the backend with the relevant movie id. (Appendix [5]).

- Backend code for reading all movies and deleting a movie. In the server.js file, we imported relevant route files for movies. (Appendix [6])

- Import the relevant methods for admin in the admin controller file. (Appendix [7]).

- The method for getting all movies or a movie from the database. (Appendix [8]).

- Get relevant movie id from frontend through http DELETE request and find that movie by id from the database and delete it. (Appendix [9]).

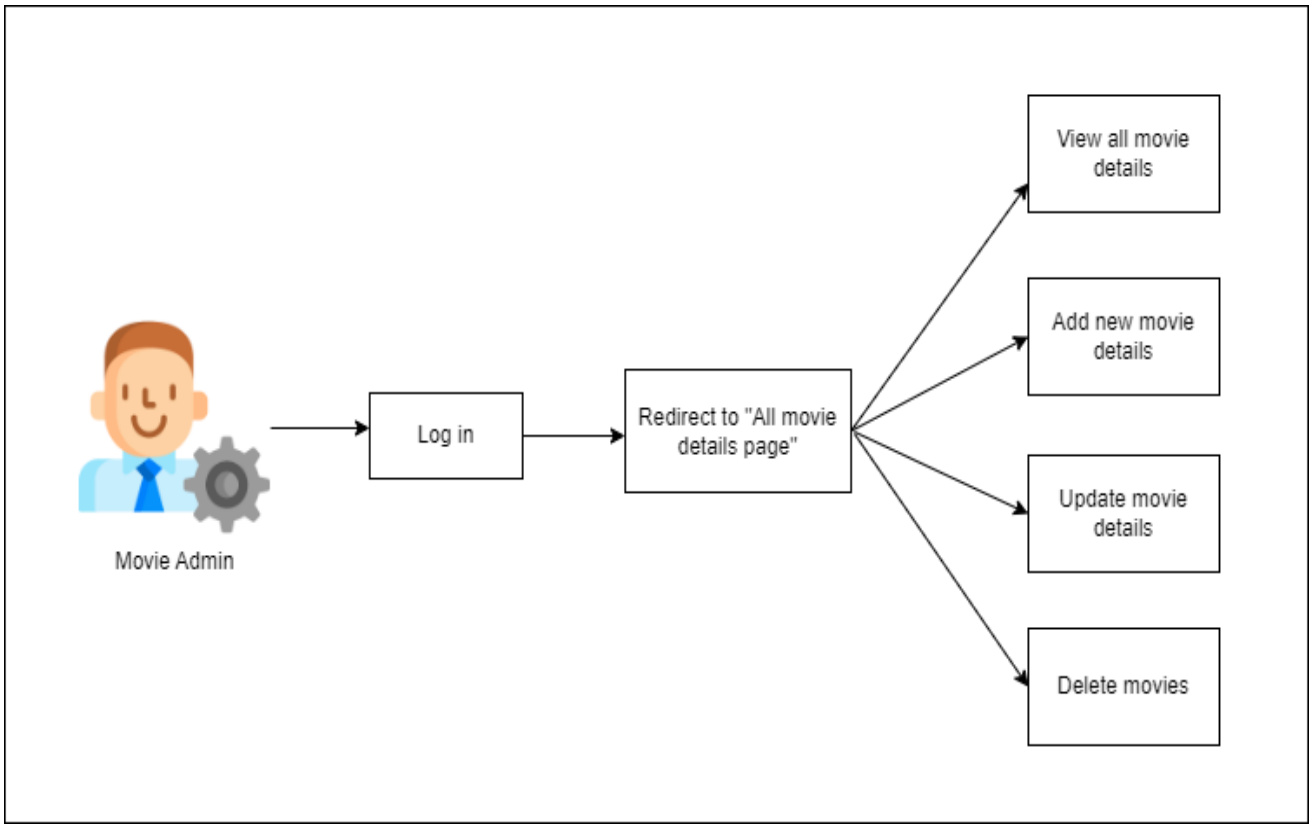
- This is the frontend code for adding a new movie. In this code pass new movie data to the relevant backend route through http POST request. (Appendix [10]).

- This is the Backend code for adding a new movie to the database. In this code, Get relevant new movie details and add those data to the database. (Appendix [11]).

- If needs to update a movie can select the relevant movie and update relevant data and send PUT request to the database. This is the frontend code for updating movie data. In here pass
  relevant movie id and updated data to the backend through http PUT request. (Appendix [12]

- This is the movie update code in the backend. In this code, get the updated data relevant to the specific movie and the id of that movie from frontend through http PUT request. After get the movie id and updated data find the relevant movie by using the movie id from the database and pass the updated data to that movie and save it. (Appendix [13])

- Bellow diagram is shown movie admin procedure.



Customer

- Register as a customer by providing their details. (Appendix [14] [15]) .

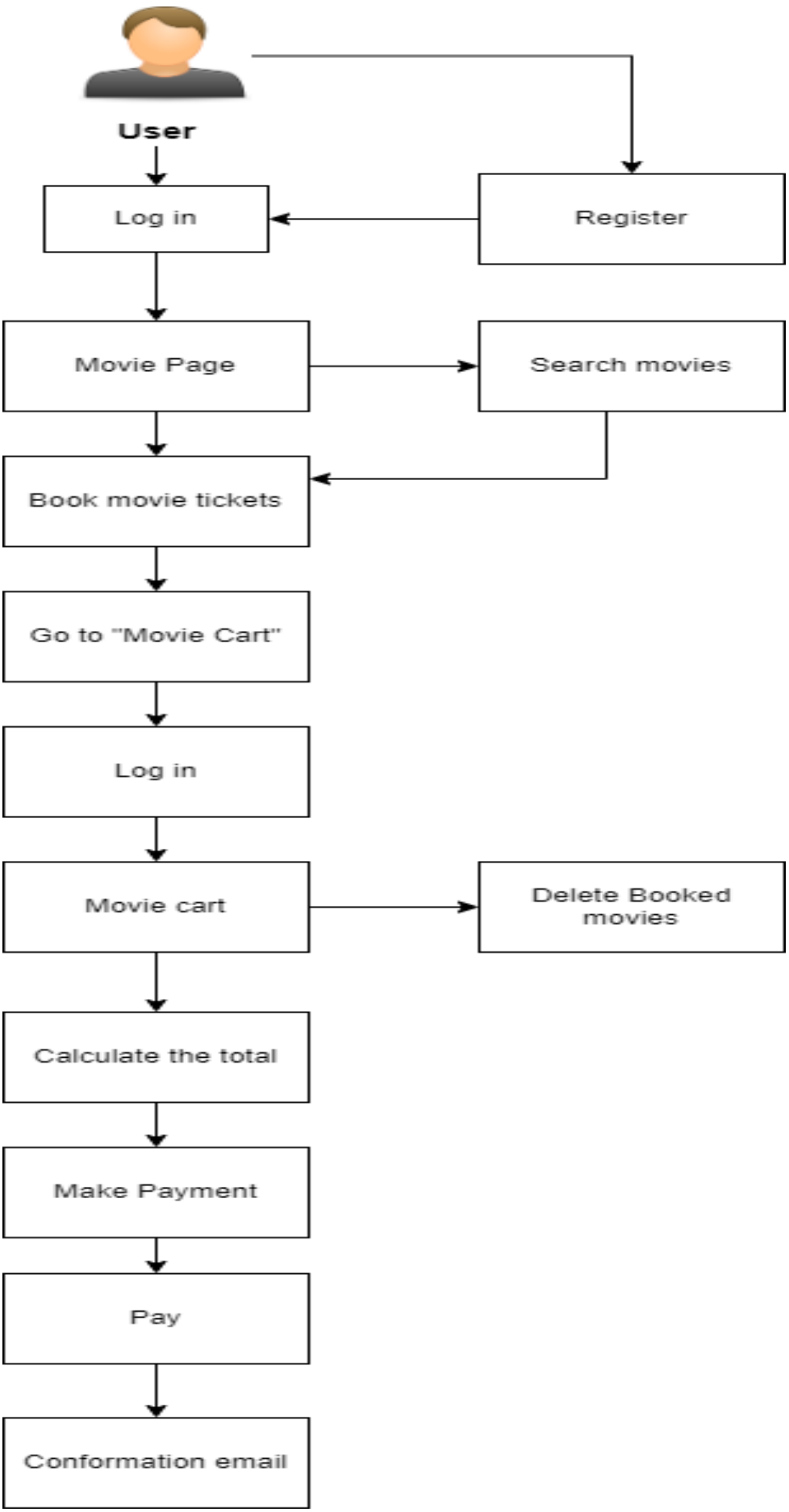
- Firstly, the Customer needs to log in to the system. If the user enters the wrong email and password, then we get an error message. If the user

enters a valid username and password, then we get the successful message, and the user navigates to the movie home page. (Appendix [16] [17]).

- Then customer can see movie home page. Then the customer can book movie tickets or customer can search for movies. (Appendix [18] [19] [20])

- After their booking, those details are shown in the cart. In that interface, they can delete the booking if they need, or they can easily calculate the total price and make a credit card payment. Finally, they will get their confirmation email. That is the very easy process a customer has to face in this system. (Appendix [21] [22] [23] [25]).

- The diagram is attached given below.


User
Log in
Register
Movie Page
Search movies
Book movie tickets
Go to "Movie Cart"
Log in
Movie cart
Delete Booked movies
Calculate the total
Make Payment
Pay
Conformation email

System Admin

- Firstly, the System admin needs to log in to the system. If the user enters the wrong email and password, then we get an error message. If the user enters a valid username and password, then we get the successful message, and the user navigates to the System admin page. (Appendix [26] [27] [28])

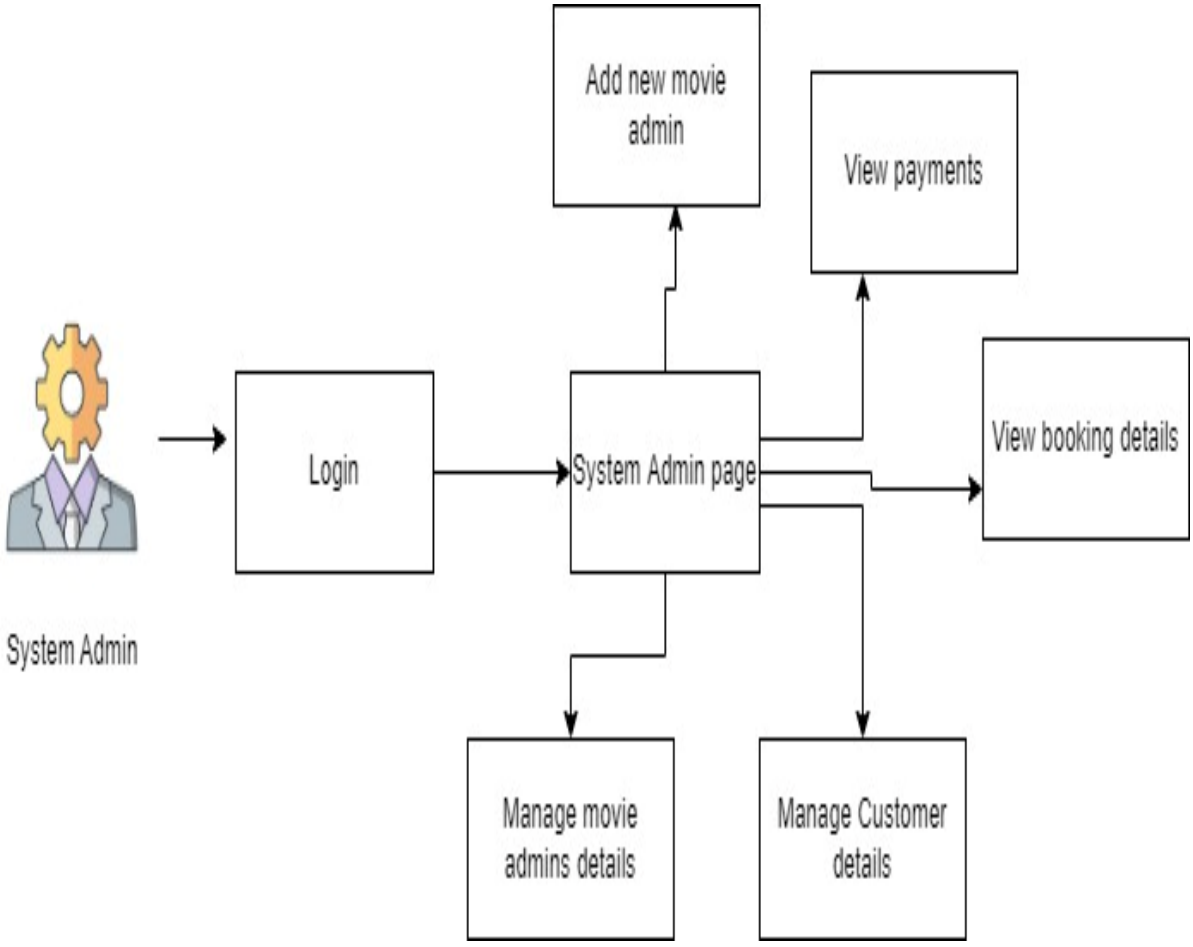
- After that system admin can manage customers details by clicking the 'manage customer' button.(Appendix [29] [30][31])

- As well as system admin can view movie tickets booking details. (Appendix [32])

- As well as System admin can view payment details by clicking the 'View payments' button.(Appendix [34] [35])

- Click the System admin home page has the 'Manage Admins' button system admin can easily manage their details. (Appendix [36] [37][38][39]).

- Clicking the 'Add New Admin' button System admin can add a new admin for the system.(Appendix [40] [41])


Add new movie admin
View payments
View booking details
Login
System Admin page
System Admin
Manage movie admins details
Manage Customer details

Benefits of the system

- Users can easily get updates.
- The system provides a great deal of flexibility.
- Well-organized the customers.
- Provide an easy payment platform.
- Faster communication.

Appendix [1]

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import axios from "axios";
import "./App.css";

function AdminLogin() {
  const [valid, setValid] = useState(false);
  const [admin, setAdmin] = useState({
    email: "",
    password: "",
  });

  let navigate = useNavigate();

  function sendData(e) {
    e.preventDefault();

    axios
      .post("http://localhost:5000/api/admin/validate", admin)
      .then((res) => {
        if (res.status === 200) {
          alert("admin validated");
          setValid(res.data);

          navigate("/manageMovies");
        }
      })
      .catch((err) => {
        alert("Login details are invalid, Please try again!!!");

        setValid(false);
      });
  }

  function handleChange(event) {
    const { name, value } = event.target;

    setAdmin((preValue) => {
      return {
        ...preValue,
        [name]: value,
      };
    });
  }

  return (
```

```
    <div>
      <div className="container mt-5">
        <div className="loginForm">
          <h1>Movie Admin Login</h1>

          <div className="row">
            <div className="col-sm-8">
              <div className="card">
                <div className="card-body">
                  <form onSubmit={sendData}>
                    <div className="form-group">
                      <label for="id">Email</label>
                      <input
                        type="email"
                        className="form-control"
                        name="email"
                        value={admin.email}
                        onChange={handleChange}
                        required
                      />
                    </div>
                    <div className="form-group">
                      <label for="password">Password</label>
                      <input
                        type="password"
                        className="form-control"
                        name="password"
                        value={admin.password}
                        onChange={handleChange}
                        required
                      />
                    </div>
                    <button type="submit" className="btn btn-dark">
                      Login
                    </button>
                  </form>
                  <br />
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    );
}

export default AdminLogin;
```

Appendix [2]

```
const express = require("express");
const mongoose = require("mongoose");
const cors = require("cors");
const path = require("path");

require("dotenv").config();

const app = express();
app.use(cors());
app.use(express.json());

const port = process.env.PORT || 5000;
const uri = process.env.MONGO_URI;

mongoose.connect(uri, { useNewUrlParser: true, useUnifiedTopology: true });
mongoose.connection.once("open", () => {
  console.log("MongoDB Connected");
});

app.use("/api/movies", require("./routes/Movie.route"));
app.use("/api/admin", require("./routes/Admin.route"));
app.use("/api/customers", require("./routes/Customer.route"));
app.use("/api/bookings", require("./routes/Book.route"));
app.use("/api/payments", require("./routes/Payment.route"));

//deploy code

//serve static assets if in production
if (process.env.NODE_ENV === "production") {
  //set static folder
  app.use(express.static(path.join(__dirname, "/client/build")));

  app.get("*", (req, res) => {
    res.sendFile(path.join(__dirname, "client", "build", "index.html"));
  });
} else {
  app.get("/", (req, res) => {
    res.send("Api running");
  });
}

app.listen(port, () => {
  console.log("Server is starting on port " + port);
```

Appendix [3]

```
const express = require("express");
const router = express.Router();
const {
  addAdmin,
  getAdmins,
  getAdmin,
  updateAdmin,
  removeAdmin,
  validateAdmin,
} = require("../controllers/Admin.controller");

router.get("/", getAdmins);

router.get("/:id", getAdmin);

router.post("/", addAdmin);

router.put("/:id", updateAdmin);

router.delete("/:id", removeAdmin);

router.post("/validate", validateAdmin);

module.exports = router;
```

Appendix [4]

```
const validateAdmin = async (req, res) => {
  const mail = req.body.email;
  const pass = md5(req.body.password);

  try {
    const foundUser = await Admin.findOne({ email: mail });

    if (!foundUser) {
      return res.status(404).json("invalid user");
    } else if (foundUser.password === pass) {
      return res.status(200).json(true);
    } else {
```

```
      return res.status(404).json("incorrect password");
    }
  } catch (error) {
    res.status(400).json(error);
  }
};
```

Appendix [5]

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import "./App.css";

import DeleteForeverIcon from "@mui/icons-material/DeleteForever";
import EditIcon from "@mui/icons-material/Edit";

export default function ManageMovies() {
  const [movies, setMovies] = useState([]);

  useEffect(() => {
    function getMovies() {
      axios
        .get("http://localhost:5000/api/movies")
        .then((res) => {
          setMovies(res.data);
        })
        .catch((err) => {
          alert(err.message);
        });
    }
    getMovies();
  }, []);

  function deleteMovie(_id) {
    axios
      .delete("http://localhost:5000/api/movies/" + _id)
      .then((res) => {
        console.log(res.data);

        alert("movie deleted");
      })
      .catch((err) => {
```

```
      alert(err);
    });

  setMovies(movies.filter((movie) => movie._id !== _id));
}

return (
  <div className="all">
    <div>
      <a href="/addMovie">
        <button type="button" className="btn btn-secondary">
          <b style={{ fontSize: "130%" }}>Add new movie</b>
        </button>
      </a>
    </div>
    <h2 className="heading">All Movies Details</h2>
    <table className="table table-bordered">
      <thead className="table-dark">
        <tr>
          <th scope="col">No</th>
          <th scope="col">Movie Title</th>
          <th scope="col">Theater Name</th>
          <th scope="col">Ticket Price .Rs.</th>
          <th scope="col">Show Date</th>
          <th scope="col">Show Time</th>
          <th></th>
        </tr>
      </thead>
      <tbody className="table-light">
        {movies.map((movie, index) => {
          return (
            <tr key={movie._id}>
              <td>{index + 1}</td>
              <td>{movie.MovieTitle}</td>
              <td>{movie.theaterName}</td>
              <td>{movie.price}.00</td>
              <td>{movie.date}</td>
              <td>{movie.time}</td>
              <td>
                <a
                  className="btn btn-warning"
                  href={`/updateMovie/${movie._id}`}
                >
                  <EditIcon />
                     Update
                </a>
                    
                <a
```

```
                    className="btn btn-danger"
                    href="#"
                    onClick={() => {
                      if (
                        window.confirm(
                          "Are you sure you wish to delete this record?"
                        )
                      )
                        deleteMovie(movie._id);
                    }}
                  >
                    <DeleteForeverIcon />
                       Delete
                  </a>
                </td>
              </tr>
            );
          })}
        </tbody>
      </table>
    </div>
  );
}
```

Appendix [6]

```
const express = require("express");
const mongoose = require("mongoose");
const cors = require("cors");
const path = require("path");

require("dotenv").config();

const app = express();
app.use(cors());
app.use(express.json());

const port = process.env.PORT || 5000;
const uri = process.env.MONGO_URI;

mongoose.connect(uri, { useNewUrlParser: true, useUnifiedTopology: true });
mongoose.connection.once("open", () => {
  console.log("MongoDB Connected");
});

app.use("/api/movies", require("./routes/Movie.route"));
```

```
app.use("/api/admin", require("./routes/Admin.route"));
app.use("/api/customers", require("./routes/Customer.route"));
app.use("/api/bookings", require("./routes/Book.route"));
app.use("/api/payments", require("./routes/Payment.route"));

//deploy code

//serve static assets if in production
if (process.env.NODE_ENV === "production") {
  //set static folder
  app.use(express.static(path.join(__dirname, "/client/build")));

  app.get("*", (req, res) => {
    res.sendFile(path.join(__dirname, "client", "build", "index.html"));
  });
} else {
  app.get("/", (req, res) => {
    res.send("Api running");
  });
}

app.listen(port, () => {
  console.log("Server is starting on port " + port);
});
```

Appendix [7]

```
const express = require("express");
const router = express.Router();
const {
  addMovie,
  getMovies,
  getMovie,
  updateMovie,
  removeMovie,
} = require("../controllers/Movie.controller");

router.get("/", getMovies);

router.get("/:id", getMovie);

router.post("/", addMovie);

router.put("/:id", updateMovie);
```

```
router.delete("/:id", removeMovie);
module.exports = router;
```

Appendix [8]

```
const getMovies = async (req, res) => {
  try {
    const movies = await Movie.find();
    res.json(movies);
  } catch (error) {
    res.status(400).json(error);
  }
};

const getMovie = async (req, res) => {
  const movieId = req.params.id;

  try {
    const movie = await Movie.findById(movieId);
    res.json(movie);
  } catch (error) {
    res.status(400).json(error);
  }
};
```

Appendix [9]

```
const removeMovie = async (req, res) => {
  const movieId = req.params.id;

  try {
    const movie = await Movie.findById(movieId);

    if (!movie) {
      return res.status(404).json("There is no movie to remove");
    }

    const removedMovie = await Movie.findByIdAndDelete(movieId);
    res.status(200).json(removedMovie);
  } catch (error) {
    res.status(400).json(error.message);
  }
};
```

Appendix [10]

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import axios from "axios";
import "./App.css";

function AddMovie() {
  const [movie, setMovie] = useState({
    MovieTitle: "",
    CoverPhoto: "",
    desc: "",
    cast: "",
    theaterName: "",
    date: "",
    time: "",
    price: "",
  });

  let navigate = useNavigate();

  function sendData(e) {
    e.preventDefault();

    axios
      .post("http://localhost:5000/api/movies", movie)
      .then(() => {
        alert("movie added");
        navigate("/manageMovies");
      })
      .catch((err) => {
        alert(err);
      });

    setMovie({
      MovieTitle: "",
      CoverPhoto: "",
      desc: "",
      cast: "",
      theaterName: "",
      date: "",
      time: "",
      price: "",
    });
  }

  function handleChange(event) {
    const { name, value } = event.target;
```

```
  setMovie((preValue) => {
    return {
      ...preValue,
      [name]: value,
    };
  });
}

return (
  <div>
    <div className="container">
      <div className="formStyle">
        <h2 className="heading">Add New Movie</h2>
        <form onSubmit={sendData}>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Movie Title
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
                name="MovieTitle"
                placeholder="enter movie title"
                onChange={handleChange}
                value={movie.MovieTitle}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Cover Photo
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
                name="CoverPhoto"
                placeholder="paste cover photo url"
                onChange={handleChange}
                value={movie.CoverPhoto}
                required
              />
            </div>
```

```
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Cast
            </label>
            <div className="col-sm-10">
              <input
                maxlength="50"
                type="text"
                className="form-control"
                id=""
                name="cast"
                placeholder="enter cast details"
                onChange={handleChange}
                value={movie.cast}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Theater Name
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
                name="theaterName"
                placeholder="enter theater name"
                onChange={handleChange}
                value={movie.theaterName}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Ticket Price
            </label>
            <div className="col-sm-10">
              <input
                type="number"
                className="form-control"
                id=""
                name="price"
                placeholder="enter ticket price for one person"
                onChange={handleChange}
```

```
            value={movie.price}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Date
        </label>
        <div className="col-sm-10">
          <input
            type="date"
            className="form-control"
            id=""
            name="date"
            onChange={handleChange}
            value={movie.date}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Time
        </label>
        <div className="col-sm-10">
          <input
            type="time"
            className="form-control"
            id=""
            name="time"
            onChange={handleChange}
            value={movie.time}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Description
        </label>
        <div className="col-sm-10">
          <textarea
            maxlength="200"
            type="text"
            className="form-control"
            id=""
            name="desc"
```

```
                placeholder="enter movie description"
                onChange={handleChange}
                value={movie.desc}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <div className="col-sm-10">
              <button type="submit" className="btn btn-secondary">
                <b style={{ fontSize: "130%" }}>Submit</b>
              </button>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
  );
}

export default AddMovie;
```

Appendix [11]

```
app.use("/api/movies", require("./routes/Movie.route"));
```

```
router.post("/", addMovie);
```

```
const addMovie = (req, res) => {
  const { MovieTitle, CoverPhoto, desc, cast, theaterName, date, time, price }
=
    req.body;

  const movie = new Movie({
    MovieTitle,
    CoverPhoto,
    desc,
```

```
    cast,
    theaterName,
    date,
    time,
    price,
  });

  movie
    .save()
    .then((createdMovie) => {
      res.json(createdMovie);
    })
    .catch((error) => {
      res.status(400).json(error);
    });
};
```

Appendix [12]

```
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import axios from "axios";
import { useParams } from "react-router";
import "./App.css";

function UpdateMovie() {

  const [movie, setMovie] = useState({
    MovieTitle: "",
    CoverPhoto: "",
    desc: "",
    cast: "",
    theaterName: "",
    date: "",
    time: "",
    price: "",
  });

  let navigate = useNavigate();

  const { id } = useParams();

  useEffect(() => {
    function getMovie() {
      axios
        .get(`http://localhost:5000/api/movies/${id}`)
        .then((res) => {
          setMovie(res.data);
```

```
      })
      .catch((err) => {
        alert(err.message);
      });
  }
  getMovie();
}, []);

function sendData(e) {
  e.preventDefault();

  axios
    .put(`http://localhost:5000/api/movies/${id}`, movie)
    .then(() => {
      alert("movie updated");
      navigate("/manageMovies");
    })
    .catch((err) => {
      alert(err);
    });
}

function handleChange(event) {
  const { name, value } = event.target;

  setMovie((preValue) => {
    return {
      ...preValue,
      [name]: value,
    };
  });
}

return (
  <div>
    <div className="container">
      <div className="formStyle">
        <h2 className="heading">Update Movie</h2>
        <form onSubmit={sendData}>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Movie Title
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
```

```
            name="MovieTitle"
            onChange={handleChange}
            value={movie.MovieTitle}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Cover Photo
        </label>
        <div className="col-sm-10">
          <input
            type="text"
            className="form-control"
            id=""
            name="CoverPhoto"
            onChange={handleChange}
            value={movie.CoverPhoto}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Cast
        </label>
        <div className="col-sm-10">
          <input
            type="text"
            className="form-control"
            id=""
            name="cast"
            onChange={handleChange}
            value={movie.cast}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Theater Name
        </label>
        <div className="col-sm-10">
          <input
            type="text"
            className="form-control"
            id=""
```

```
            name="theaterName"
            onChange={handleChange}
            value={movie.theaterName}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Ticket Price
        </label>
        <div className="col-sm-10">
          <input
            type="number"
            className="form-control"
            id=""
            name="price"
            onChange={handleChange}
            value={movie.price}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Date
        </label>
        <div className="col-sm-10">
          <input
            type="date"
            className="form-control"
            id=""
            name="date"
            onChange={handleChange}
            value={movie.date}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Time
        </label>
        <div className="col-sm-10">
          <input
            type="time"
            className="form-control"
            id=""
```

```
                name="time"
                onChange={handleChange}
                value={movie.time}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Description
            </label>
            <div className="col-sm-10">
              <textarea
                type="text"
                maxlength="200"
                className="form-control"
                id=""
                name="desc"
                onChange={handleChange}
                value={movie.desc}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <div className="col-sm-10">
              <button type="submit" className="btn btn-secondary">
                <b style={{ fontSize: "130%" }}>Update</b>
              </button>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
  );
}

export default UpdateMovie;
```

Appendix [13]

```
app.use("/api/movies", require("./routes/Movie.route"));
```

```
router.put("/:id", updateMovie);
```



```
const updateMovie = async (req, res) => {
  const movieId = req.params.id;

  try {
    const movie = await Movie.findById(movieId);

    if (!movie) {
      return res.status(404).json("There is no movie to update");
    }

    const {
      MovieTitle,
      CoverPhoto,
      desc,
      cast,
      theaterName,
      date,
      time,
      price,
    } = req.body;

    const updatedMovie = await Movie.findByIdAndUpdate(movieId, {
      MovieTitle,
      CoverPhoto,
      desc,
      cast,
      theaterName,
      date,
      time,
      price,
    });

    res.status(200).json(updatedMovie);
  } catch (error) {
    res.status(400).json(error.message);
  }
};
```

Appendix [14] -Front-end

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import axios from "axios";
import "./App.css";

function CustomerRegister() {
  const [customer, setCustomer] = useState({
    fName: "",
    lName: "",
    phone: "",
    email: "",
    password: "",
  });

  let navigate = useNavigate();

  function sendData(e) {
    e.preventDefault();

    axios
      .post("http://localhost:5000/api/customers", customer)
      .then(() => {
        alert("successfully registerd");
        navigate("/");
      })
      .catch((err) => {
        alert(err);
      });

    setCustomer({
      fName: "",
      lName: "",
      phone: "",
      email: "",
      password: "",
    });
  }

  function handleChange(event) {
    const { name, value } = event.target;
```

```
    setCustomer((preValue) => {
      return {
        ...preValue,
        [name]: value,
      };
    });
  }

  return (
    <div>
      <div className="container">
        <div className="formStyle">
          <h2 className="heading">Customer Registration</h2>
          <form onSubmit={sendData}>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                First Name
              </label>
              <div className="col-sm-10">
                <input
                  type="text"
                  className="form-control"
                  id=""
                  name="fName"
                  placeholder="enter first name"
                  onChange={handleChange}
                  value={customer.fName}
                  required
                />
              </div>
            </div>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                Last Name
              </label>
              <div className="col-sm-10">
                <input
                  type="text"
                  className="form-control"
                  id=""
                  name="lName"
                  placeholder="enter last name"
                  onChange={handleChange}
                  value={customer.lName}
                  required
                />
              </div>
            </div>
```

```
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                Phone Number
              </label>
              <div className="col-sm-10">
                <input
                  type="number"
                  className="form-control"
                  id=""
                  name="phone"
                  placeholder="enter phone number"
                  onChange={handleChange}
                  value={customer.phone}
                  required
                />
              </div>
            </div>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                Email
              </label>
              <div className="col-sm-10">
                <input
                  type="email"
                  className="form-control"
                  id=""
                  name="email"
                  placeholder="enter email"
                  onChange={handleChange}
                  value={customer.email}
                  required
                />
              </div>
            </div>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                Password
              </label>
              <div className="col-sm-10">
                <input
                  type="password"
                  className="form-control"
                  id=""
                  name="password"
                  placeholder="enter password"
                  onChange={handleChange}
                  value={customer.password}
                  required
```

```
                />
              </div>
            </div>

            <div className="form-group row">
              <div className="col-sm-10">
                <button type="submit" className="btn btn-secondary">
                  <b style={{ fontSize: "130%" }}>Submit</b>
                </button>
              </div>
            </div>
          </form>
        </div>
      </div>
    </div>
  );
}

export default CustomerRegister;
```

Appendix [15] – Backend

```
const addCustomer = (req, res) => {
  const { fName, lName, phone, email } = req.body;
  const password = md5(req.body.password);

  const customer = new Customer({
    fName,
    lName,
    phone,
    email,
    password,
  });

  customer
    .save()
    .then((createdCustomer) => {
      res.json(createdCustomer);
    })
    .catch((error) => {
      res.status(400).json(error);
    });
};
```

Appendix [16] -Front-end

```
import React, { useState } from "react";
import axios from "axios";
import "./App.css";
import Movies from "./Movies";

function CustomerLogin() {
  const [valid, setValid] = useState(false);
  const [customer, setCustomer] = useState({
    email: "",
    password: "",
  });

  function sendData(e) {
    e.preventDefault();

    axios
      .post("http://localhost:5000/api/customers/validate", customer)
      .then((res) => {
        if (res.status === 200) {
          alert("customer validated");
          setValid(res.data);
        }
      })
      .catch((err) => {
        alert("Login details are invalid, Please try again!!!");

        setValid(false);
      });
  }

  function handleChange(event) {
    const { name, value } = event.target;

    setCustomer((preValue) => {
      return {
        ...preValue,
        [name]: value,
      };
    });
  }

  return (
    <div>
      {valid === false ? (
        <div className="container mt-5">
          <div className="loginForm">
```

```
          <h1>Customer Login</h1>

          <div className="row">
            <div className="col-sm-8">
              <div className="card">
                <div className="card-body">
                  <form onSubmit={sendData}>
                    <div className="form-group">
                      <label for="id">Email</label>
                      <input
                        type="email"
                        className="form-control"
                        name="email"
                        value={customer.email}
                        onChange={handleChange}
                        required
                      />
                    </div>
                    <div className="form-group">
                      <label for="password">Password</label>
                      <input
                        type="password"
                        className="form-control"
                        name="password"
                        value={customer.password}
                        onChange={handleChange}
                        required
                      />
                    </div>
                    <button type="submit" className="btn btn-dark">
                      Login
                    </button>
                  </form>
                  <br />
                  <br />

                  <div>
                    <label for="password">
                      If you don't have an account click to
                    </label>
                      
                    <a href="/addCustomer">Register</a>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
```

```
        </div>
      ) : (
        <Movies customerEmail={customer.email} />
      )}
    </div>
  );
}

export default CustomerLogin;
```

Appendix [17] -Back-end

```
const validateCustomer = async (req, res) => {
  const mail = req.body.email;
  const pass = md5(req.body.password);

  try {
    const foundUser = await Customer.findOne({ email: mail });

    if (!foundUser) {
      return res.status(404).json("invalid customer");
    } else if (foundUser.password === pass) {
      return res.status(200).json(true);
    } else {
      return res.status(404).json("incorrect password");
    }
  } catch (error) {
    res.status(400).json(error);
  }
```

Appendix [18]

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import Movie from "./Movie";
import "./App.css";

export default function Movies(props) {
  const [movies, setMovies] = useState([]);
```

```
useEffect(() => {
  function getMovies() {
    axios
      .get("http://localhost:5000/api/movies")
      .then((res) => {
        setMovies(res.data);
      })
      .catch((err) => {
        alert(err.message);
      });
  }
  getMovies();
}, []);

function filterData(movies, searchKey) {
  const result = movies.filter((movie) => {
    return movie.MovieTitle.toLowerCase().includes(searchKey);
  });

  setMovies(result);
}

function handleSearchArea(e) {
  const searchKey = e.target.value;

  axios.get("http://localhost:5000/api/movies").then((res) => {
    filterData(res.data, searchKey);
  });
}

return (
  <div className="movies">
    <h1 className="moviesHeading">Latest Movies</h1>

    <div
      class="input-group"
      style={{
        width: "17%",
        height: "30px",
        margin: "0px 40px 20px",
      }}
    >
      <input
        type="text"
        class="form-control rounded"
        placeholder="Search"
        aria-label="Search"
        aria-describedby="search-addon"
```

```
          onChange={handleSearchArea}
        />
      </div>

      {movies.map((movie, index) => {
        return (
          <Movie
            key={index}
            title={movie.MovieTitle}
            cover={movie.CoverPhoto}
            desc={movie.desc}
            cast={movie.cast}
            theaterName={movie.theaterName}
            date={movie.date}
            time={movie.time}
            price={movie.price}
            cMail={props.customerEmail}
          />
        );
      })}
    </div>
  );
}
```

Appendix [19]- Back-end

```
const getMovies = async (req, res) => {
  try {
    const movies = await Movie.find();
    res.json(movies);
  } catch (error) {
    res.status(400).json(error);
  }
};
```

Appendix [20]

```
const addBooking = (req, res) => {
  const { email, title, theater, date, time, price, tickets } = req.body;

  const book = new Book({
    email,
    title,
    theater,
    date,
    time,
    price,
    tickets,
  });

  book
    .save()
    .then((createdBooking) => {
      res.json(createdBooking);
    })
    .catch((error) => {
      res.status(400).json(error);
    });
};
```

Appendix [21]

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import "./App.css";

import DeleteForeverIcon from "@mui/icons-material/DeleteForever";

export default function Cart(props) {
  const [bookings, setBookings] = useState([]);
  const [valid, setValid] = useState(false);
  const [pay, setPay] = useState(false);

  useEffect(() => {
    function getBookings() {
      axios
        .get("http://localhost:5000/api/bookings")
        .then((res) => {
          const results = res.data;
          setBookings(
            results.filter((result) => result.email === props.customerEmail)
          );
```

```
      })
      .catch((err) => {
        alert(err.message);
      });
  }
  getBookings();
}, []);

function deleteBooking(_id) {
  axios
    .delete("http://localhost:5000/api/bookings/" + _id)
    .then((res) => {
      console.log(res.data);

      alert("booking deleted");
    })
    .catch((err) => {
      alert(err);
    });

  setBookings(bookings.filter((booking) => booking._id !== _id));
}

let sum = 0;

bookings.forEach((booking) => {
  sum += Number(booking.price) * Number(booking.tickets);
});

function calculate() {
  setValid(true);
}

const [payment, setPayment] = useState({
  cardNumber: "",
  cvcNumber: "",
  amount: "",
  name: "",
  email: props.customerEmail,
});

function handlePayment() {
  setPay(true);
  setPayment((preValue) => {
    return {
      ...preValue,
      amount: sum,
    };
```

```
  });
}

function sendData(e) {
  e.preventDefault();

  axios
    .post("http://localhost:5000/api/payments", payment)
    .then(() => {
      alert("payment successful");
    })
    .catch((err) => {
      alert(err);
    });

  setPayment({
    cardNumber: "",
    cvcNumber: "",
    amount: "",
    name: "",
    email: props.customerEmail,
  });
}

function handleChange(event) {
  const { name, value } = event.target;

  setPayment((preValue) => {
    return {
      ...preValue,
      [name]: value,
    };
  });
}

return (
  <div>
    {pay === false ? (
      <div className="all">
        <h2 className="heading">Booked Movies</h2>
        <p style={{ fontSize: "140%", fontWeight: "bold" }}>
          Your Email --- {props.customerEmail}
        </p>
        <table className="table table-bordered">
          <thead className="table-dark">
            <tr>
              <th scope="col">Movie Title</th>
              <th scope="col">Theater Name</th>
```

```
            <th scope="col">Show Date</th>
            <th scope="col">Show Time</th>
            <th scope="col">Ticket Price .Rs.</th>
            <th scope="col">No of Tickets</th>
            <th scope="col"></th>
          </tr>
        </thead>
        <tbody className="table-light">
          {bookings.map((booking, index) => {
            return (
              <tr key={booking._id}>
                <td>{booking.title}</td>
                <td>{booking.theater}</td>
                <td>{booking.date}</td>
                <td>{booking.time}</td>
                <td>{booking.price}.00</td>
                <td>{booking.tickets}</td>
                <td>
                  <a
                    className="btn btn-danger"
                    href="#"
                    onClick={() => {
                      if (
                        window.confirm(
                          "Are you sure you wish to delete this record?"
                        )
                      )
                        deleteBooking(booking._id);
                    }}
                  >
                    <DeleteForeverIcon />
                       Delete
                  </a>
                </td>
              </tr>
            );
          })}
        </tbody>
      </table>

      <br />
      <button onClick={calculate}>
        <h5>
          <b>Calculate Total</b>
        </h5>
      </button>
      <br />
      <br />
```

```
        {valid && (
          <div>
            <h3>Total Price is : {sum}.00</h3>
            <br />
            <button onClick={handlePayment}>
              <h5>
                <b>Make Payment</b>
              </h5>
            </button>
          </div>
        )}
      </div>
    ) : (
      <div className="container">
        <div className="formStylePayment">
          <h2 className="heading" style={{ margin: "30px" }}>
            Enter payment details{" "}
          </h2>
          <p style={{ fontSize: "140%", fontWeight: "bold" }}>
            Your Email --- {props.customerEmail}
          </p>{" "}
          <p style={{ fontSize: "140%", fontWeight: "bold" }}>
            Amount --- Rs.{sum}.00
          </p>
          <form onSubmit={sendData}>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                Credit Card Number
              </label>

              <div className="col-sm-7">
                <input
                  type="tel"
                  maxlength="19"
                  className="form-control"
                  id=""
                  name="cardNumber"
                  placeholder="enter your credit card number"
                  onChange={handleChange}
                  value={payment.cardNumber}
                  required
                />
              </div>
            </div>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                CVC Number
              </label>
```

```
              <div className="col-sm-7">
                <input
                  type="tel"
                  maxlength="3"
                  className="form-control"
                  id=""
                  name="cvcNumber"
                  placeholder="enter CVC number"
                  onChange={handleChange}
                  value={payment.cvcNumber}
                  required
                />
              </div>
            </div>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                Full Name
              </label>
              <div className="col-sm-7">
                <input
                  type="text"
                  className="form-control"
                  id=""
                  name="name"
                  placeholder="enter your full name"
                  onChange={handleChange}
                  value={payment.name}
                  required
                />
              </div>
            </div>
            <div className="form-group row">
              <div className="col-sm-10">
                <button type="submit" className="btn btn-secondary">
                  <b style={{ fontSize: "130%" }}>Pay</b>
                </button>
              </div>
            </div>
          </form>
        </div>
      </div>
    )}
  </div>
);
}
```

Appendix [22] – Back-end get bookings

```
const getBookings = async (req, res) => {
  try {
    const bookings = await Book.find();
    res.json(bookings);
  } catch (error) {
    res.status(400).json(error);
  }
};

const getBooking = async (req, res) => {
  const bookId = req.params.id;

  try {
    const booking = await Book.findById(bookId);
    res.json(booking);
  } catch (error) {
    res.status(400).json(error);
  }
};
```

Appendix [23] – Back-end remove bookings

```
const removeBooking = async (req, res) => {
  const bookId = req.params.id;

  try {
    const booking = await Book.findById(bookId);

    if (!booking) {
      return res.status(404).json("There is no booking to remove");
    }

    const removedBooking = await Book.findByIdAndDelete(bookId);
    res.status(200).json(removedBooking);
  } catch (error) {
    res.status(400).json(error.message);
  }
};
```

Appendix [24] -Front-end payment

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import "./App.css";

export default function Payments() {
  const [payments, setPayments] = useState([]);

  useEffect(() => {
    function getPayments() {
      axios
        .get("http://localhost:5000/api/payments")
        .then((res) => {
          setPayments(res.data);
        })
        .catch((err) => {
          alert(err.message);
        });
    }
    getPayments();
  }, []);

  return (
    <div className="all">
      <h2 className="heading">Customers Payment Details</h2>
      <table className="table table-bordered">
        <thead className="table-dark">
          <tr>
            <th scope="col">No</th>
            <th scope="col">Customer Name</th>
            <th scope="col">Email</th>
            <th scope="col">Credit card Number</th>
            <th scope="col">CVC Number</th>
            <th scope="col">Amount .Rs.</th>
          </tr>
        </thead>
        <tbody className="table-light">
          {payments.map((payment, index) => {
            return (
              <tr key={payment._id}>
                <td>{index + 1}</td>
                <td>{payment.name}</td>
                <td>{payment.email}</td>
                <td>{payment.cardNumber}</td>
                <td>{payment.cvcNumber}</td>
                <td>{payment.amount}.00</td>
              </tr>
```

```
            );
          })}
        </tbody>
      </table>
    </div>
  );
}
```

Appendix [25] – Back-end payment

```
const addPayment = (req, res) => {
  const { cardNumber, amount, cvcNumber, name, email } = req.body;

  const payment = new Payment({
    cardNumber,
    amount,
    cvcNumber,
    name,
    email,
  });

  payment
    .save()
    .then((createdPayment) => {
      res.json(createdPayment);
    })
    .catch((error) => {
      res.status(400).json(error);
    });

  //send email to relevant customer
  var transporter = nodemailer.createTransport({
    service: "gmail",
    auth: {
      user: "dmrwijerathne@gmail.com",
      pass: "dmrw98629",
    },
  });

  var mailOptions = {
    from: "dmrwijerathne@gmail.com",
    to: email,
    subject: "Payment Successfull",
    text: `Hi ${name} you have succesfully payed. Your amount is
Rs.${amount}.00`,
  };
```

```
  transporter.sendMail(mailOptions, function (error, info) {
    if (error) {
      console.log(error);
    } else {
      console.log("Email sent: " + info.response);
    }
  });
};
```

Appendix [26]

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import axios from "axios";
import "./App.css";

function SystemAdminLogin() {
  const [valid, setValid] = useState(false);
  const [admin, setAdmin] = useState({
    email: "",
    password: "",
  });

  let navigate = useNavigate();

  function sendData(e) {
    e.preventDefault();

    axios
      .post("http://localhost:5000/api/admin/validate", admin)
      .then((res) => {
        if (res.status === 200) {
          alert("admin validated");
          setValid(res.data);

          navigate("/handleSystem");
        }
      })
      .catch((err) => {
        alert("Login details are invalid, Please try again!!!");

        setValid(false);
      });
  }

  function handleChange(event) {
```

```
    const { name, value } = event.target;

    setAdmin((preValue) => {
      return {
        ...preValue,
        [name]: value,
      };
    });
  }

  return (
    <div>
      <div className="container mt-5">
        <div className="loginForm">
          <h1>System Admin Login</h1>

          <div className="row">
            <div className="col-sm-8">
              <div className="card">
                <div className="card-body">
                  <form onSubmit={sendData}>
                    <div className="form-group">
                      <label for="id">Email</label>
                      <input
                        type="email"
                        className="form-control"
                        name="email"
                        value={admin.email}
                        onChange={handleChange}
                        required
                      />
                    </div>
                    <div className="form-group">
                      <label for="password">Password</label>
                      <input
                        type="password"
                        className="form-control"
                        name="password"
                        value={admin.password}
                        onChange={handleChange}
                        required
                      />
                    </div>
                    <button type="submit" className="btn btn-dark">
                      Login
                    </button>
                  </form>
                  <br />
```

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

export default SystemAdminLogin;
```

Appendix [27]

```
const validateAdmin = async (req, res) => {
  const mail = req.body.email;
  const pass = md5(req.body.password);

  try {
    const foundUser = await Admin.findOne({ email: mail });

    if (!foundUser) {
      return res.status(404).json("invalid user");
    } else if (foundUser.password === pass) {
      return res.status(200).json(true);
    } else {
      return res.status(404).json("incorrect password");
    }
  } catch (error) {
    res.status(400).json(error);
  }
};
```

Appendix [28]

```
import React from "react";
import "./App.css";

export default function HandleSystem() {
  return (
    <div className="sysMenu">
```

```
      <div className="navItem">
        <a href="/customers">
          <button type="button" className="btn btn-secondary menuBtn">
            Manage customers
          </button>
        </a>
      </div>
      <div className="navItem">
        <a href="/bookings">
          <button type="button" className="btn btn-secondary menuBtn">
            View movie ticket bookings
          </button>
        </a>
      </div>
      <div className="navItem">
        <a href="/payments">
          <button type="button" className="btn btn-secondary menuBtn">
            View payments
          </button>
        </a>
      </div>
      <div className="navItem">
        <a href="/admins">
          <button type="button" className="btn btn-secondary menuBtn">
            Manage admins
          </button>
        </a>
      </div>
      <div className="navItem">
        <a href="/addAdmin">
          <button type="button" className="btn btn-secondary menuBtn">
            Add a new admin
          </button>
        </a>
      </div>
    </div>
  );
}
```

Appendix [29]

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import "./App.css";

import DeleteForeverIcon from "@mui/icons-material/DeleteForever";

export default function Customers() {
  const [customers, setCustomers] = useState([]);

  useEffect(() => {
    function getCustomers() {
      axios
        .get("http://localhost:5000/api/customers")
        .then((res) => {
          setCustomers(res.data);
        })
        .catch((err) => {
          alert(err.message);
        });
    }
    getCustomers();
  }, []);

  function deleteCustomer(_id) {
    axios
      .delete("http://localhost:5000/api/customers/" + _id)
      .then((res) => {
        alert("customer deleted");
      })
      .catch((err) => {
        alert(err);
      });

    setCustomers(customers.filter((customer) => customer._id !== _id));
  }

  return (
    <div className="all">
      <h2 className="heading">All Customers Details</h2>
      <table className="table table-bordered">
        <thead className="table-dark">
          <tr>
            <th scope="col">No</th>
            <th scope="col">First Name </th>
            <th scope="col">Last Name</th>
            <th scope="col">Phone Number</th>
```

```
            <th scope="col">Email</th>
            <th></th>
          </tr>
        </thead>
        <tbody className="table-light">
          {customers.map((customer, index) => {
            return (
              <tr key={customer._id}>
                <td>{index + 1}</td>
                <td>{customer.fName}</td>
                <td>{customer.lName}</td>
                <td>{customer.phone}</td>
                <td>{customer.email}</td>
                <td>
                  <a
                    className="btn btn-danger"
                    href="#"
                    onClick={() => {
                      if (
                        window.confirm(
                          "Are you sure you wish to delete this record?"
                        )
                      )
                        deleteCustomer(customer._id);
                    }}
                  >
                    <DeleteForeverIcon />
                       Delete
                  </a>
                </td>
              </tr>
            );
          })}
        </tbody>
      </table>
    </div>
  );
}
```

Appendix [30] – Back-end get customer

```
const getCustomers = async (req, res) => {
  try {
    const customers = await Customer.find();
    res.json(customers);
  } catch (error) {
```

```
    res.status(400).json(error);
  }
};
```

Appendix [31] -Back-end delete method

```
const removeCustomer = async (req, res) => {
  const cusId = req.params.id;

  try {
    const customer = await Customer.findById(cusId);

    if (!customer) {
      return res.status(404).json("There is no customer to remove");
    }

    const removedCustomer = await Customer.findByIdAndDelete(cusId);
    res.status(200).json(removedCustomer);
  } catch (error) {
    res.status(400).json(error.message);
  }
};
```

Appendix [32]

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import "./App.css";

export default function Bookings() {
  const [bookings, setBookings] = useState([]);

  useEffect(() => {
    function getBookings() {
      axios
        .get("http://localhost:5000/api/bookings")
        .then((res) => {
          setBookings(res.data);
        })
        .catch((err) => {
          alert(err.message);
        });
    }
    getBookings();
```

```
  }, []);

  return (
    <div className="all">
      <h2 className="heading">All movie tickets bookings details</h2>
      <table className="table table-bordered">
        <thead className="table-dark">
          <tr>
            <th scope="col">No</th>
            <th scope="col">Customer Email</th>
            <th scope="col">Movie Title</th>
            <th scope="col">Movie Theater</th>
            <th scope="col">Show Date</th>
            <th scope="col">Show Time</th>
            <th scope="col">Ticket Price .Rs.</th>
            <th scope="col">No of Tickets</th>
          </tr>
        </thead>
        <tbody className="table-light">
          {bookings.map((booking, index) => {
            return (
              <tr key={booking._id}>
                <td>{index + 1}</td>
                <td>{booking.email}</td>
                <td>{booking.title}</td>
                <td>{booking.theater}</td>
                <td>{booking.date}</td>
                <td>{booking.time}</td>
                <td>{booking.price}.00</td>
                <td>{booking.tickets}</td>
              </tr>
            );
          })}
        </tbody>
      </table>
    </div>
  );
}
```

Appendix [33]- Back-end get bookings

```
const getBookings = async (req, res) => {
  try {
    const bookings = await Book.find();
    res.json(bookings);
  } catch (error) {
    res.status(400).json(error);
  }
};
```

Appendix [34]

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import "./App.css";

export default function Payments() {
  const [payments, setPayments] = useState([]);

  useEffect(() => {
    function getPayments() {
      axios
        .get("http://localhost:5000/api/payments")
        .then((res) => {
          setPayments(res.data);
        })
        .catch((err) => {
          alert(err.message);
        });
    }
    getPayments();
  }, []);

  return (
    <div className="all">
      <h2 className="heading">Customers Payment Details</h2>
      <table className="table table-bordered">
        <thead className="table-dark">
          <tr>
            <th scope="col">No</th>
            <th scope="col">Customer Name</th>
            <th scope="col">Email</th>
            <th scope="col">Credit card Number</th>
            <th scope="col">CVC Number</th>
            <th scope="col">Amount .Rs.</th>
```

```
          </tr>
        </thead>
        <tbody className="table-light">
          {payments.map((payment, index) => {
            return (
              <tr key={payment._id}>
                <td>{index + 1}</td>
                <td>{payment.name}</td>
                <td>{payment.email}</td>
                <td>{payment.cardNumber}</td>
                <td>{payment.cvcNumber}</td>
                <td>{payment.amount}.00</td>
              </tr>
            );
          })}
        </tbody>
      </table>
    </div>
  );
}
```

Appendix [35]

```
const getPayments = async (req, res) => {
  try {
    const payments = await Payment.find();
    res.json(payments);
  } catch (error) {
    res.status(400).json(error);
  }
};
```

Appendix [36] -front-end View admin and delete admin page

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import "./App.css";

import DeleteForeverIcon from "@mui/icons-material/DeleteForever";
import EditIcon from "@mui/icons-material/Edit";

export default function Admins() {
  const [admins, setAdmins] = useState([]);

  useEffect(() => {
    function getAdmins() {
      axios
        .get("http://localhost:5000/api/admin")
        .then((res) => {
          setAdmins(res.data);
        })
        .catch((err) => {
          alert(err.message);
        });
    }
    getAdmins();
  }, []);

  function deleteAdmin(_id) {
    axios
      .delete("http://localhost:5000/api/admin/" + _id)
      .then((res) => {
        alert("admin deleted");
      })
      .catch((err) => {
        alert(err);
      });

    setAdmins(admins.filter((admin) => admin._id !== _id));
  }

  return (
    <div className="all">
      <h2 className="heading">All Admins Details</h2>
      <table className="table table-bordered">
        <thead className="table-dark">
          <tr>
            <th scope="col">No</th>
            <th scope="col">First Name </th>
            <th scope="col">Last Name</th>
```

```
            <th scope="col">Company Id</th>
            <th scope="col">NIC</th>
            <th scope="col">Email</th>
            <th></th>
          </tr>
        </thead>
        <tbody className="table-light">
          {admins.map((admin, index) => {
            return (
              <tr key={admin._id}>
                <td>{index + 1}</td>
                <td>{admin.fName}</td>
                <td>{admin.lName}</td>
                <td>{admin.companyId}</td>
                <td>{admin.nic}</td>
                <td>{admin.email}</td>
                <td>
                  <a
                    className="btn btn-warning"
                    href={`/updateAdmin/${admin._id}`}
                  >
                    <EditIcon />
                       Update
                  </a>
                      
                  <a
                    className="btn btn-danger"
                    href="#"
                    onClick={() => {
                      if (
                        window.confirm(
                          "Are you sure you wish to delete this record?"
                        )
                      )
                        deleteAdmin(admin._id);
                    }}
                  >
                    <DeleteForeverIcon />
                       Delete
                  </a>
                </td>
              </tr>
            );
          })}
        </tbody>
      </table>
    </div>
  );
```

```
}
```

Appendix [37] – front-end Update admin page

```
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import axios from "axios";
import { useParams } from "react-router";
import "./App.css";

export default function UpdateAdmin() {
  const [admin, setAdmin] = useState({
    fName: "",
    lName: "",
    companyId: "",
    nic: "",
    email: "",
    password: "",
  });

  let navigate = useNavigate();

  const { id } = useParams();

  useEffect(() => {
    function getAdmin() {
      axios
        .get(`http://localhost:5000/api/admin/${id}`)
        .then((res) => {
          setAdmin(res.data);
        })
        .catch((err) => {
          alert(err.message);
        });
    }
    getAdmin();
  }, []);

  function sendData(e) {
    e.preventDefault();

    axios
      .put(`http://localhost:5000/api/admin/${id}`, admin)
      .then(() => {
        alert("admin updated");
        navigate("/admins");
```

```
    })
    .catch((err) => {
      alert(err);
    });
}

function handleChange(event) {
  const { name, value } = event.target;

  setAdmin((preValue) => {
    return {
      ...preValue,
      [name]: value,
    };
  });
}

return (
  <div>
    <div className="container">
      <div className="formStyle">
        <h2 className="heading">Update Admin</h2>
        <form onSubmit={sendData}>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              First Name
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
                name="fName"
                onChange={handleChange}
                value={admin.fName}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Last Name
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
```

```
            name="lName"
            onChange={handleChange}
            value={admin.lName}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Company Id
        </label>
        <div className="col-sm-10">
          <input
            type="text"
            className="form-control"
            id=""
            name="companyId"
            onChange={handleChange}
            value={admin.companyId}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          NIC
        </label>
        <div className="col-sm-10">
          <input
            type="text"
            className="form-control"
            id=""
            name="nic"
            onChange={handleChange}
            value={admin.nic}
            required
          />
        </div>
      </div>
      <div className="form-group row">
        <label for="" className="col-sm-2 col-form-label">
          Email
        </label>
        <div className="col-sm-10">
          <input
            type="text"
            className="form-control"
            id=""
```

```
                name="email"
                onChange={handleChange}
                value={admin.email}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Password
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
                name="password"
                onChange={handleChange}
                value={admin.password}
                required
              />
            </div>
          </div>

          <div className="form-group row">
            <div className="col-sm-10">
              <button type="submit" className="btn btn-secondary">
                <b style={{ fontSize: "130%" }}>Update</b>
              </button>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
  );
}
```

Appendix [38]-Back-end get admins

```
const getAdmins = async (req, res) => {
  try {
    const admins = await Admin.find();
    res.json(admins);
  } catch (error) {
    res.status(400).json(error);
  }
};
```

Appendix [39] – Back-end update and delete admins

```
const updateAdmin = async (req, res) => {
  const adminId = req.params.id;

  try {
    const admin = await Admin.findById(adminId);

    if (!admin) {
      return res.status(404).json("There is no admin to update");
    }

    const { fName, lName, companyId, nic, email, password } = req.body;

    const updatedAdmin = await Admin.findByIdAndUpdate(adminId, {
      fName,
      lName,
      companyId,
      nic,
      email,
      password,
    });

    res.status(200).json(updatedAdmin);
  } catch (error) {
    res.status(400).json(error.message);
  }
};

const removeAdmin = async (req, res) => {
  const adminId = req.params.id;

  try {
    const admin = await Admin.findById(adminId);
```

```
    if (!admin) {
      return res.status(404).json("There is no admin to remove");
    }

    const removedAdmin = await Admin.findByIdAndDelete(adminId);
    res.status(200).json(removedAdmin);
  } catch (error) {
    res.status(400).json(error.message);
  }
};
```

Appendix [40]- Front-end add admin page

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import axios from "axios";
import "./App.css";

function AddAdmin() {
  const [admin, setAdmin] = useState({
    fName: "",
    lName: "",
    companyId: "",
    nic: "",
    email: "",
    password: "",
  });

  let navigate = useNavigate();

  function sendData(e) {
    e.preventDefault();

    axios
      .post("http://localhost:5000/api/admin", admin)
      .then(() => {
        alert("successfully registerd");
        navigate("/admins");
      })
      .catch((err) => {
        alert(err);
      });
```

```
    setAdmin({
      fName: "",
      lName: "",
      companyId: "",
      nic: "",
      email: "",
      password: "",
    });
  }

  function handleChange(event) {
    const { name, value } = event.target;

    setAdmin((preValue) => {
      return {
        ...preValue,
        [name]: value,
      };
    });
  }

  return (
    <div>
      <div className="container">
        <div className="formStyle">
          <h2 className="heading">Admin Registration</h2>
          <form onSubmit={sendData}>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                First Name
              </label>
              <div className="col-sm-10">
                <input
                  type="text"
                  className="form-control"
                  id=""
                  name="fName"
                  placeholder="enter first name"
                  onChange={handleChange}
                  value={admin.fName}
                  required
                />
              </div>
            </div>
            <div className="form-group row">
              <label for="" className="col-sm-2 col-form-label">
                Last Name
              </label>
```

```
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
                name="lName"
                placeholder="enter last name"
                onChange={handleChange}
                value={admin.lName}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Company Id
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
                name="companyId"
                placeholder="enter company id"
                onChange={handleChange}
                value={admin.companyId}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              NIC
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-control"
                id=""
                name="nic"
                placeholder="enter nic"
                onChange={handleChange}
                value={admin.nic}
                required
              />
            </div>
          </div>
          <div className="form-group row">
```

```
            <label for="" className="col-sm-2 col-form-label">
              Email
            </label>
            <div className="col-sm-10">
              <input
                type="email"
                className="form-control"
                id=""
                name="email"
                placeholder="enter email"
                onChange={handleChange}
                value={admin.email}
                required
              />
            </div>
          </div>
          <div className="form-group row">
            <label for="" className="col-sm-2 col-form-label">
              Password
            </label>
            <div className="col-sm-10">
              <input
                type="password"
                className="form-control"
                id=""
                name="password"
                placeholder="enter password"
                onChange={handleChange}
                value={admin.password}
                required
              />
            </div>
          </div>

          <div className="form-group row">
            <div className="col-sm-10">
              <button type="submit" className="btn btn-secondary">
                <b style={{ fontSize: "130%" }}>Submit</b>
              </button>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
  );
}
```

```
export default AddAdmin;
```

Appendix [41] – Back-end add admin

```
const addAdmin = (req, res) => {
  const { fName, lName, companyId, nic, email } = req.body;
  const password = md5(req.body.password);

  const admin = new Admin({
    fName,
    lName,
    companyId,
    nic,
    email,
    password,
  });

  admin
    .save()
    .then((createdAdmin) => {
      res.json(createdAdmin);
    })
    .catch((error) => {
      res.status(400).json(error);
    });
};
```

*****