

SPL-02 Project Report

Dhaka University Swimming Pool Management System

Submitted by

Ibne Bin Rafid (BSSE1330)

Eftekhar Mahmud Efty (BSSE 1309)

Supervised by

Abdus Satter

Assistant Professor



Institute of Information Technology

University of Dhaka

[13-06-2024]

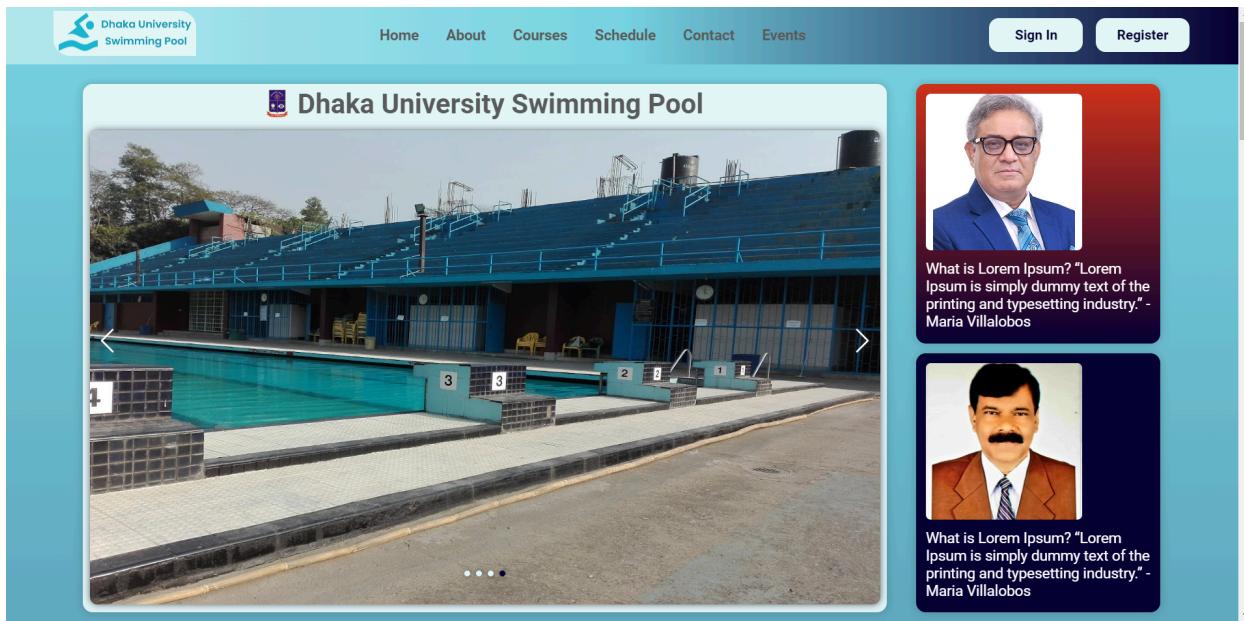
Signature of Supervisor

Table of Contents

- 1. Introduction
- 2. Background of the Project
 - 2.1. Swimming Pool Management System
 - 2.2. Web Application Development
 - 2.3. Database Management
- 3. Description of the Project
 - 3.1. User Management
 - 3.2. Schedule Management
 - 3.3. Financial Management
 - 3.4. Communication and Notification
- 4. Implementation
 - 4.1. Frontend
 - 4.1.1. Styling
 - 4.1.2. Components Structure
 - 4.1.3. User Interaction
 - 4.1.4. Testing
 - 4.2. Backend
 - 4.3. Database
 - 4.3.1. Database Design
 - 4.3.2. Data Management
- 5. User Manual
 - 5.1. Landing Page
 - 5.2. Sign In
 - 5.3. User Registration
 - 5.4. Schedule Management
 - 5.5. Financial Management
 - 5.6. Event Management
 - 5.6. Admin Panel
- 6. Challenges Faced
 - 6.1. Dependency Related Issues
 - 6.2. Integration of Components
 - 6.3. Maintainability in Backend Code
- 7. Conclusion and Future Work
- 8. References

1. Introduction

In response to the growing need for efficient management of Dhaka University's swimming pools, we propose the development of a comprehensive Swimming Pool Management System. This system aims to streamline various administrative tasks, enhance user experience, and improve overall operational efficiency.



2. Background of the Project

2.1. Swimming Pool Management System

The Swimming Pool Management System is designed to automate and optimize the management of swimming pool facilities at Dhaka University. This includes user registrations, scheduling, financial transactions, and communication with stakeholders.

2.2. Web Application Development

The project will be implemented as a web-based platform, utilizing modern web development technologies such as ReactJS for the frontend and NodeJS for the backend.

2.3. Database Management

A robust database management system, specifically MongoDB, will be employed

to securely store and manage user information, schedules, and financial data.

3. Description of the Project

3.1. User Management

The system will provide functionalities for user registration, login, and profile management. Each user will have a personalized account for accessing the swimming pool facilities.

3.2. Schedule Management

The system will allow administrators to create and manage schedules for various activities such as swimming lessons, training sessions, and maintenance periods. Users can view and register for these activities.

3.3. Financial Management

The system will facilitate online transactions, allowing users to make payments for various services. It will also provide financial insights for administrators, helping them track the pool's profit and loss.

3.4. Communication and Notification

Administrators will be able to send notifications and announcements to users. Users can also provide feedback and reviews, fostering engagement and transparency.

4. Implementation

4.1. Frontend

4.1.1. Styling

The frontend is styled using raw CSS to ensure a responsive and user-friendly interface. Consistent and responsive design principles are applied to ensure usability across various devices.

```

1  /* Header.css */
2
3  /* Base styles */
4  .header {
5    height: 60px;
6    background: linear-gradient(to right, #b4e8ef, #78d5e3, #080034);
7    padding: 0 50px;
8    display: flex;
9    justify-content: space-between;
10   align-items: center;
11   position: fixed;
12   top: 0;
13   left: 0;
14   right: 0;
15   z-index: 1000;
16 }
17
18 .logo img {
19   width: 146px;
20   height: 46px;
21   transition: transform 0.3s ease;
22 }
23
24 .logo img:hover {
25   transform: scale(1.1);
26 }
27
28 .navigation ul {
29   list-style-type: none;
30   margin: 0;
31   padding: 0;
32   display: flex;

```

Figure: Example styling of the Header component

4.1.2. Components Structure

The application is modularized into reusable components, making the development process more efficient and the codebase easier to maintain.

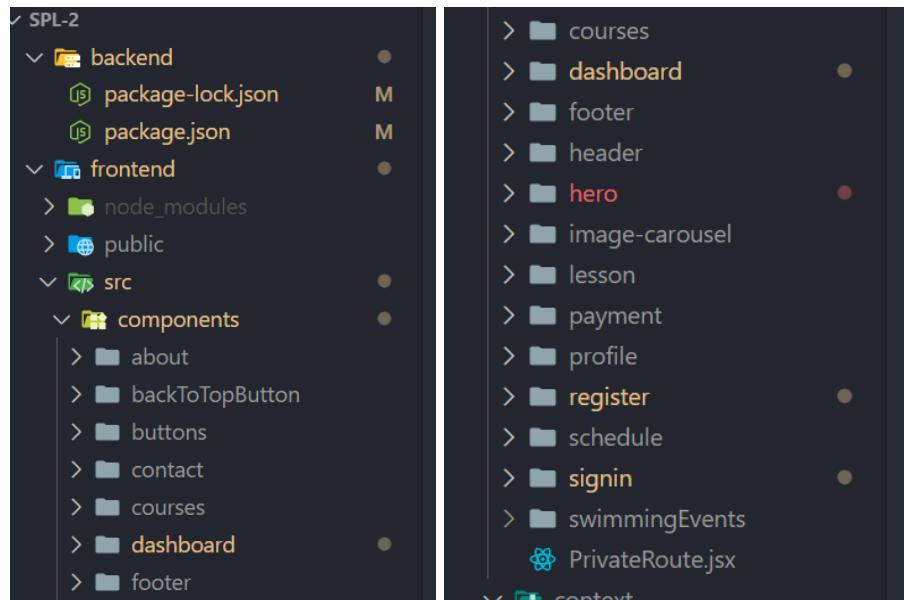


Figure: Components used in frontend

4.1.3. User Interaction

The system is providing interactive elements such as authentication forms, schedules, transaction, swimming events, swimming courses, swimming lessons interfaces to enhance user engagement and usability.

Authentication Forms: The authentication forms, including login and registration, are built using ReactJS and handle user input through controlled components. Form data is managed using state hooks and is submitted via secure API calls to the backend for authentication and user account creation.

Registration Code:

```
const Register = () => {
  const [swimmerData, setSwimmerData] = useState({
    name: "",
    email: "",
    number: "",
    category: "",
    gender: "",
    dob: "",
    registrationNumber: "",
    password: "",
    confirmPassword: "",
    courses: []
  });

  const navigate = useNavigate();

  const handleInputChange = (event) => {
    const { name, value } = event.target;
    setSwimmerData({ ...swimmerData, [name]: value });
  };
}
```

```
const handleCheckboxChange = (event) => {
  const { value, checked } = event.target;
  let updatedCourses = [...swimmerData.courses];
  if (checked) {
    updatedCourses.push(value);
  } else {
    updatedCourses = updatedCourses.filter((course) => course !== value);
  }
  setSwimmerData({ ...swimmerData, courses: updatedCourses });
};

const isValidEmail = (email) => {
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  return emailRegex.test(email);
};
```

```
const signup = async () => {
  const { password, confirmPassword, email } = swimmerData;

  if (password.length < 8) {
    alert("Password must be at least 8 characters long.");
    return;
  }

  if (password !== confirmPassword) {
    alert("Passwords do not match.");
    return;
  }

  if (!isValidEmail(email)) {
    alert("Please enter a valid email address.");
    return;
  }

  try {
    console.log('Sending swimmer data:', swimmerData); // Log data being sent

    const response = await fetch('http://localhost:4003/register', {
      method: 'POST',
      headers: {
        Accept: 'application/json',
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(swimmerData),
    });
  }
};
```

```

    return (
      <div className="signup">
        <div className="signup-container">
          <h1>Swimmer Registration Form</h1>
          <div className="signup-fields">
            <label htmlFor="swimmerName">Name</label>
            <input type="text" value={swimmerData.name} id="swimmerName" name="name" onChange={handleInputChange} required />
            <label htmlFor="swimmerEmail">Email</label>
            <input type="email" value={swimmerData.email} id="swimmerEmail" name="email" onChange={handleInputChange} required />
            <label htmlFor="swimmerNumber">Number</label>
            <input type="text" value={swimmerData.number} id="swimmerNumber" name="number" onChange={handleInputChange} required />
            <label htmlFor="swimmerCategory">Select a Swimmer Category:</label>
            <select id="swimmerCategory" name="category" value={swimmerData.category} onChange={handleInputChange} required>
              <option className='option' value="">Select Category</option>
              <option className='option' value="Dhaka University Student">Dhaka University Student</option>
              <option className='option' value="DU Teacher/Officer/Employee/Their Family Member">DU Teacher/Officer/Employee/Their Family Member</option>
              <option className='option' value="BUET/DMC Student">BUET/DMC Student</option>
              <option className='option' value="Associate Member">Associate Member</option>
              <option className='option' value="General Swimmer">General Swimmer</option>
            </select>
            <label htmlFor="swimmerDOB">Date of Birth</label>
            <input type="date" id="swimmerDOB" name="dob" value={swimmerData.dob} onChange={handleInputChange} required/>
            <label htmlFor="swimmerRegistrationNumber">Registration Number for DU Students</label>
            <input type="text" id="swimmerRegistrationNumber" name="registrationNumber" value={swimmerData

```

Figure: Registration Processes code

Sign In Code:

```

const SignIn = () => [
  const [formData, setFormData] = useState({
    email: '',
    password: '',
    showPassword: false,
  });

  const navigate = useNavigate();
  const { login } = useAuth();

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

  const togglePasswordVisibility = () => {
    setFormData({ ...formData, showPassword: !formData.showPassword });
  };
]

```

```

const handleLogin = async () => {
  console.log('Login function executed', formData);

  let responseData;
  await fetch('http://localhost:4003/login', {
    method: 'POST',
    headers: {
      Accept: 'application/form-data',
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({ email: formData.email, password: formData.password }),
  })
    .then((response) => response.json())
    .then((data) => (responseData = data));

  if (responseData.success) {
    localStorage.setItem('auth-token', responseData.token);
    login();
    navigate('/dashboard');
  } else {
    alert(responseData.errors);
  }
};

```

```

const SignIn = () => {
  return (
    <div className="login">
      <div className="login-container">
        <h1>Sign in to Dhaka University Swimming Pool</h1>
        <div className="login-fields">
          <label htmlFor="email">Email</label>
          <input
            type="email"
            value={formData.email}
            id="email"
            name="email"
            onChange={handleChange}
            required
          />
          <label htmlFor="password">
            Password
            <div className="password-input">
              <input
                type={formData.showPassword ? 'text' : 'password'}
                value={formData.password}
                id="password"
                name="password"
                onChange={handleChange}
                required
              />
              <button
                type="button"
                className="password-toggle"
                onClick={togglePasswordVisibility}
              >
                {formData.showPassword ? <FaEye /> : <FaEyeSlash />}
              </button>
            </div>
          </label>
        </div>
      </div>
    </div>
  );
};

```

Figure: SignIn Processes code

Schedules: Schedule display available time slots for swimming classes, events, and maintenance. Users can view and register for these slots and also can download as PDF.

Code and interface:

```
const Schedule = () => {
  const scheduleRef = useRef();

  const downloadAsPDF = () => {
    const input = scheduleRef.current;
    html2canvas(input, { scale: 2 }) // Increase scale for better quality
      .then(canvas => {
        const imgData = canvas.toDataURL('image/png');
        const pdf = new jsPDF('p', 'mm', 'a4'); // Create PDF in A4 size
        const imgProps = pdf.getImageProperties(imgData);
        const pdfWidth = pdf.internal.pageSize.getWidth();
        const pdfHeight = (imgProps.height * pdfWidth) / imgProps.width;
        pdf.addImage(imgData, 'PNG', 0, 0, pdfWidth, pdfHeight);
        pdf.save('DU-Swimming-Pool-Schedule.pdf');
      })
      .catch(err => console.error('Error generating PDF:', err));
  };

  const downloadAsPNG = () => {
    const input = scheduleRef.current;
    html2canvas(input, { scale: 2 }) // Increase scale for better quality
      .then(canvas => {
        const link = document.createElement('a');
        link.href = canvas.toDataURL('image/png');
        link.download = 'DU-Swimming-Pool-Schedule.png';
        link.click();
      })
      .catch(err => console.error('Error generating PNG:', err));
  };
}
```

Figure: Schedule code and Interface

Transaction Interfaces: Secure transaction interfaces are provided for users to make online payments for various courses. These interfaces are integrated with payment gateways, ensuring secure and seamless financial transactions. Users can view transaction history and download receipts.

Code and Interface:

```
app.post('/payment/success/:tran_id', async (req, res) => {
  try {
    const transactionId = req.params.tran_id;
    const paymentData = transactions[transactionId];

    if (!paymentData) {
      return res.status(400).json({ success: false, error: "Invalid transaction ID or payment data not found" });
    }

    const { userDetails, course } = paymentData;

    // Add the single course to the courses array if it's not already present
    let courses = userDetails.courses || [];
    if (course && !courses.includes(course)) {
      courses.push(course);
    }

    const registeredUser = new RegisteredUser({
      name: userDetails.name,
      email: userDetails.email,
      category: userDetails.category,
      courses: courses,
      tran_id: transactionId
    });

    await registeredUser.save();

    // Send email notification
    const transporter = nodemailer.createTransport({
      service: 'Gmail',
      auth: [
        ...
      ]
    });
  }
});
```

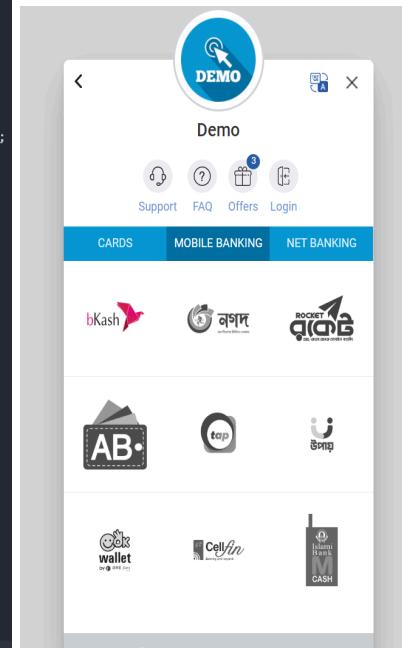


Figure: Transaction code and Interface

Courses: The courses module enables the creation and management of swimming courses offered at the pool. Users can browse available courses, view details such as schedules and fees, and register online.

Code and Interface:

```
import React from 'react';
import './CourseCard.css';
import Button1 from '../buttons/Button1';

const CourseCard = ({ className, imageSrc, title, description, buttonText, buttonLink }) => {

  return (
    <div className="courses">
      <div className={`section-1 ${className}`}>
        <div className="left-box-courses">
          <h1>{title}</h1>
          <p>{description}</p>
          <Button1 text={buttonText} to={buttonLink} className="course-btn" />
        </div>
        <div className="right-box-courses">
          <img
            src={imageSrc}
            alt="Course Image"
          />
        </div>
      </div>
    );
};

export default CourseCard;
```

The screenshot shows a website for the Dhaka University Swimming Pool. At the top, there's a navigation bar with links for Home, About, Courses, Schedule, Contact, and Events. On the right side of the nav bar are Sign In and Register buttons. Below the navigation, there are two main course modules.

Butterfly: This module features a circular illustration of a swimmer performing the butterfly stroke. The text describes the stroke as a dynamic movement resembling a butterfly's wings, requiring strength, core stability, and flexibility. A "Enroll Now" button is located below the text.

Freestyle: This module features a circular illustration of a swimmer performing the freestyle stroke. The text describes it as the most straightforward yet versatile stroke, offering freedom and fluidity. It requires endurance and focus on efficiency and speed over long distances.

Figure: Course modules code and Interface

Swimming Events: The swimming events module allows administrators to create and manage events. Users can view upcoming events and take event tickets. The interface is designed to handle event details, participant lists for admin UI, and ticket PDF download.

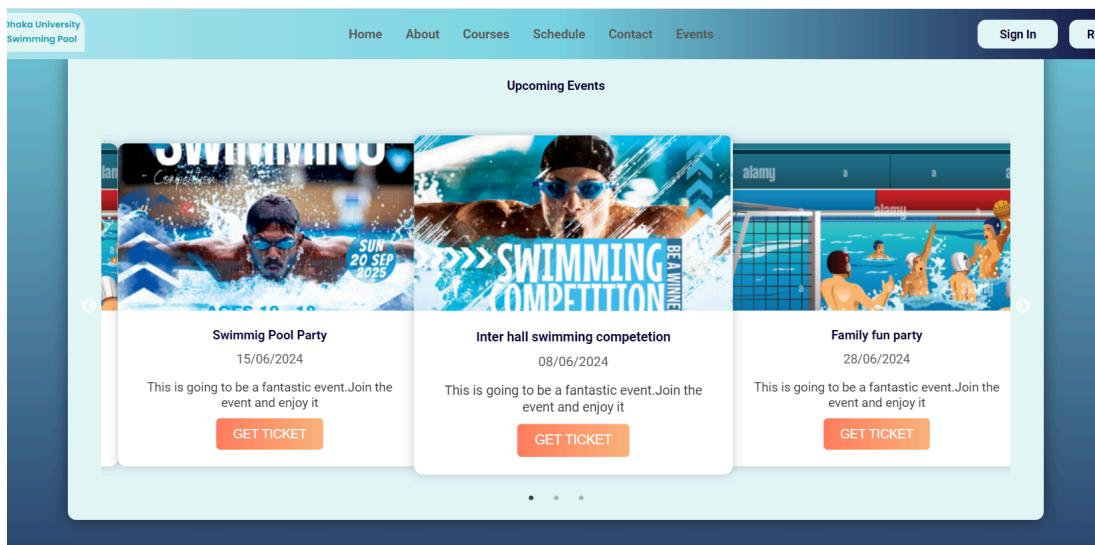


Figure: Event module UI

Swimming Lessons Interfaces: A comprehensive interface for swimming lessons includes video tutorials and guidelines to enhance the learning experience.

Code and Interface:

```
const SwimmingLessons = () => {
  return (
    <div className="swimming-lessons">
      <div className="swimming-lessons-container">
        <h1>Swimming Lessons Resources</h1>

        <div className="resources">
          <div className="resources-section">
            <h2>PDFs</h2>
            <ul>
              <li><a href="/path/to/pdf1.pdf" target="_blank" rel="noopener noreferrer">Swimming Basics</a></li>
              <li><a href="/path/to/pdf2.pdf" target="_blank" rel="noopener noreferrer">Advanced Techniques</a></li>
              <li><a href="/path/to/pdf3.pdf" target="_blank" rel="noopener noreferrer">Safety Guidelines</a></li>
            </ul>
          </div>

          <div className="resources-section">
            <h2>Photos</h2>
            <div className="photos">
              
              
              
            </div>
          </div>

          <div className="resources-section">
            <h2>Videos</h2>
            <div className="videos">
              <video controls>
                <source src="https://www.youtube.com/watch?v=p6R0h-M7S0k&ab_channel=GlobalTriathlonNetwork" type="video/mp4" />
              </video>
            </div>
          </div>
        </div>
      </div>
    </div>
  )
}
```

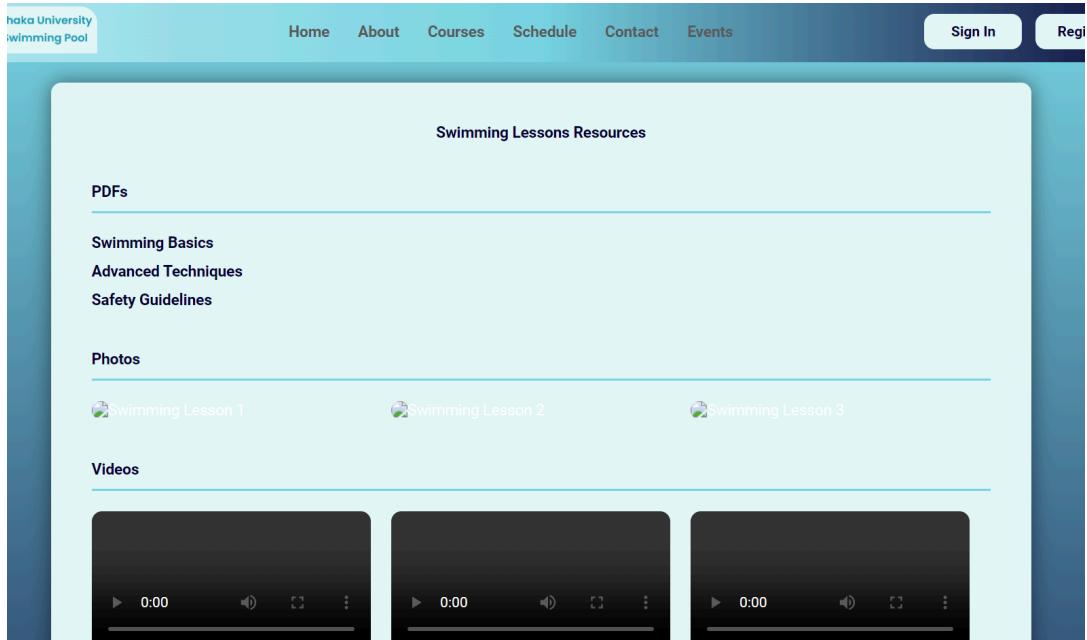


Figure: Lesson modules code and Interface

4.1.4. Testing

Extensive testing is and will be conducted to ensure the functionality, performance, and security of the application. This includes unit tests, integration tests, and user acceptance testing.

4.2. Backend

The backend is developed using NodeJS, with ExpressJS as the framework for building the API. It will handle user authentication, data management, and business logic.

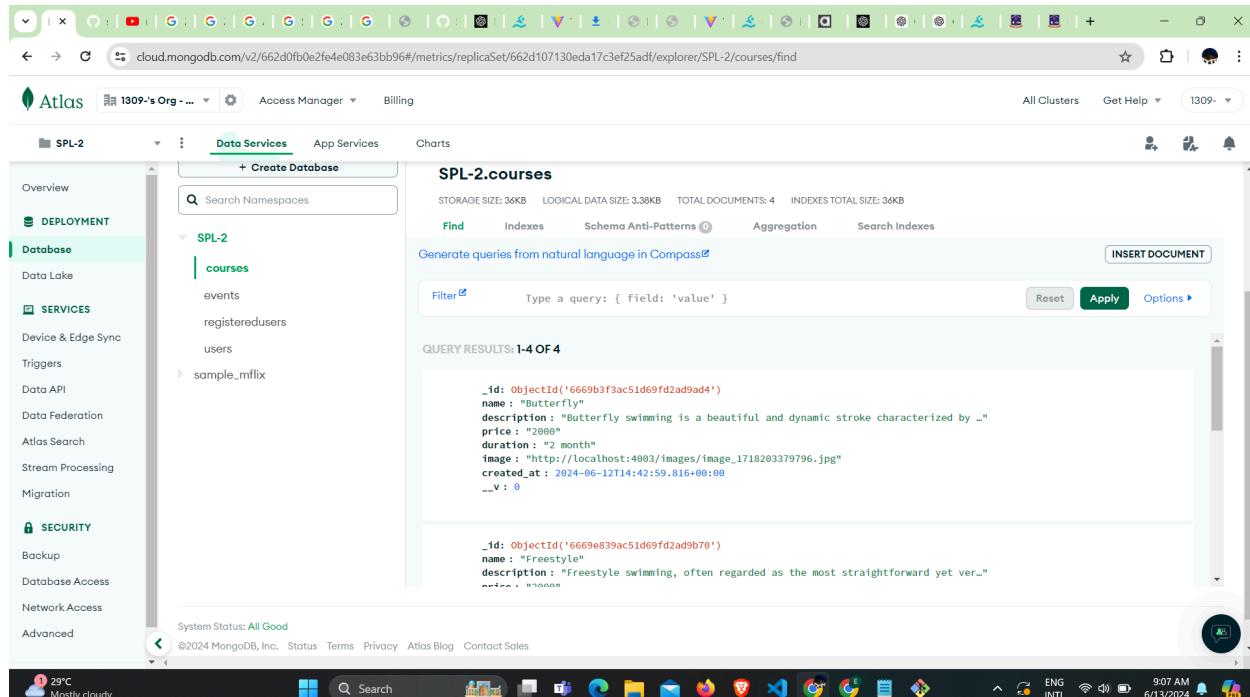
```
{
  "name": "backend",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": [
    "test": "echo \\\"Error: no test specified\\\" && exit 1",
    "start": "npm server.js",
    "dev": "nodemon server.js"
  ],
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "axios": "^1.7.2",
    "bcryptjs": "^2.4.3",
    "body-parser": "^1.20.2",
    "cors": "^2.8.5",
    "dotenv": "^16.4.5",
    "express": "^4.19.2",
    "jsonwebtoken": "^9.0.2",
    "mongoose": "^8.4.1",
    "nodemailer": "^6.9.13"
  },
  "devDependencies": {
    "nodemon": "^3.1.3"
  }
}
```

Figure: Backend dependencies

4.3. Database

4.3.1. Database Design

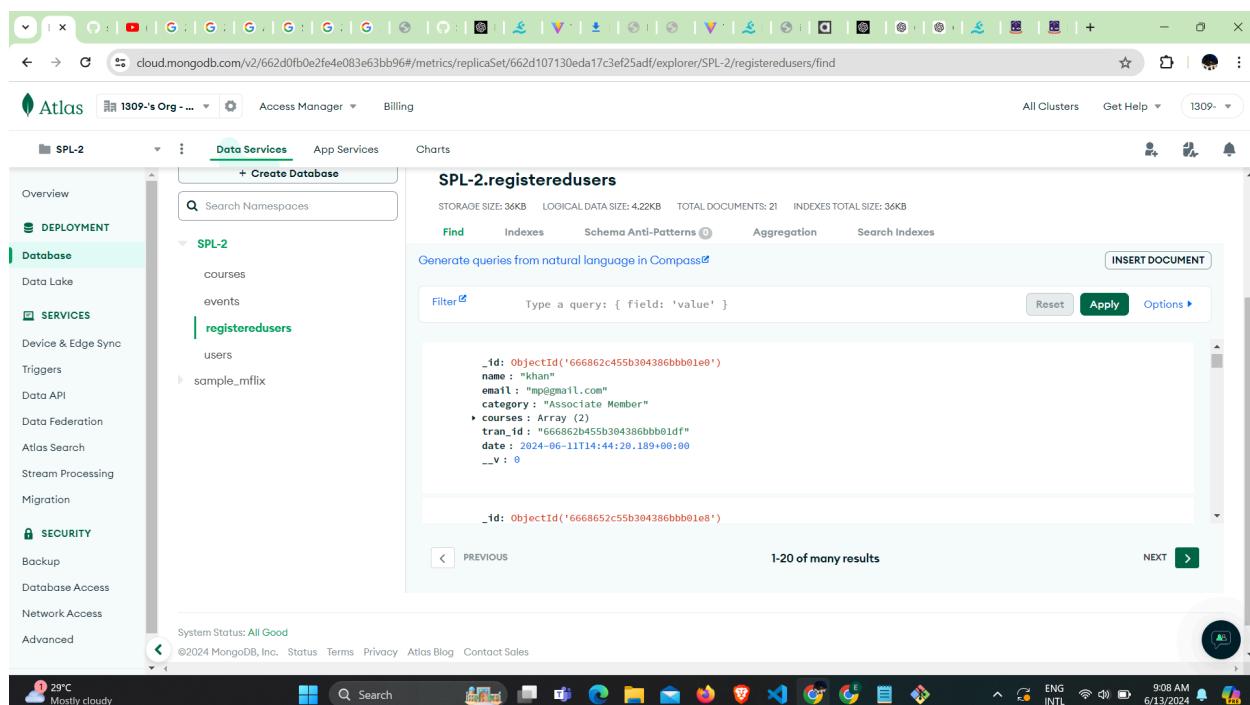
The database is designed to store user information, schedules, transactions, and other relevant data securely and efficiently.



The screenshot shows the MongoDB Atlas interface for the SPL-2 database. The left sidebar shows various services like Data Lake, Device & Edge Sync, and Stream Processing. Under the Database section, the SPL-2 database is selected, and the courses namespace is expanded, showing documents for 'butterfly' and 'Freestyle'. The right panel displays the document details for each.

```
_id: ObjectId('6669e839ac51d69fd2ad9ad4')
name : "Butterfly"
description: "Butterfly swimming is a beautiful and dynamic stroke characterized by _"
price : "2000"
duration : "2 month"
image : "http://localhost:4003/images/image_1718203379796.jpg"
created_at : 2024-06-12T14:42:59.816+00:00
__v : 0

_id: ObjectId('6669e839ac51d69fd2ad9b70')
name : "Freestyle"
description: "Freestyle swimming, often regarded as the most straightforward yet ver_"
price : "2000"
```



The screenshot shows the MongoDB Atlas interface for the SPL-2 database. The left sidebar shows various services like Data Lake, Device & Edge Sync, and Stream Processing. Under the Database section, the SPL-2 database is selected, and the registeredusers namespace is expanded, showing documents for 'khan'. The right panel displays the document details for each.

```
_id: ObjectId('666862c455b304386bbb01e0')
name : "khan"
email : "mp@gmail.com"
category : "Associate Member"
courses : Array (2)
  tran_id : "666862b455b304386bbb01df"
  date : 2024-06-11T14:44:28.189+00:00
  __v : 0

_id: ObjectId('6668652c55b304386bbb01e0')
```

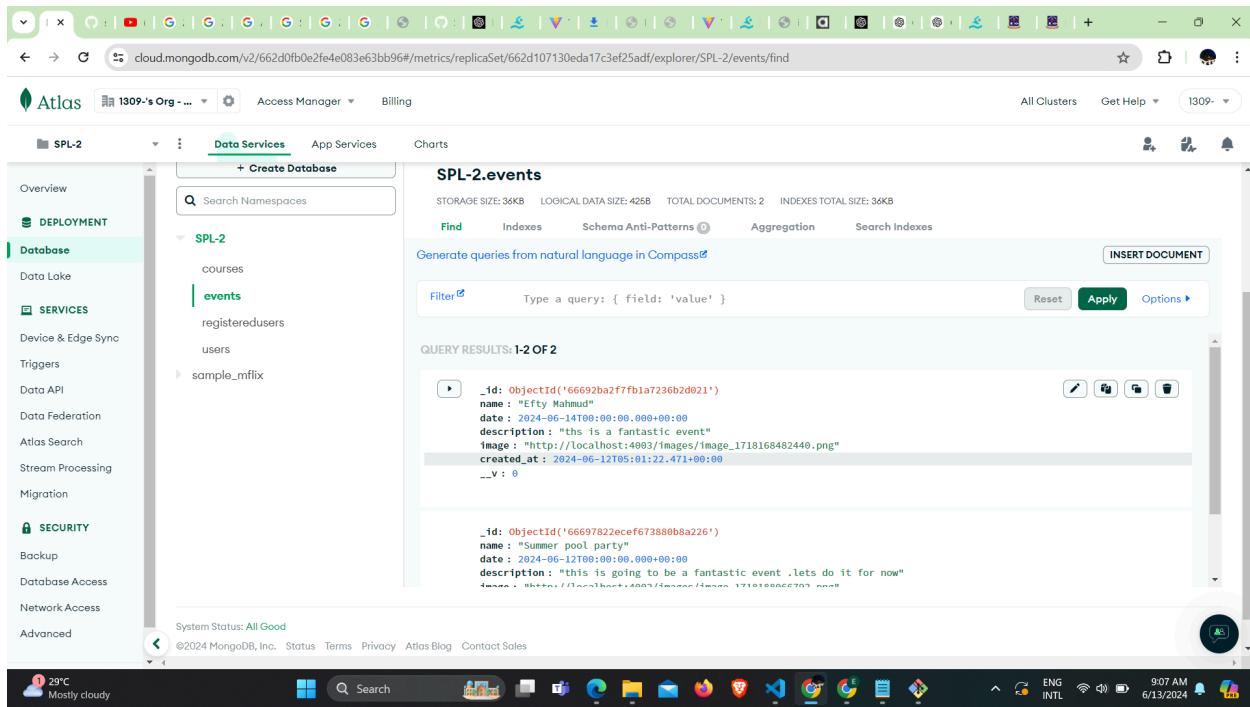


Figure: MongoDB Database implementation

4.3.2. Data Management

Data management practices are implemented to ensure data integrity, security, and availability. This includes regular backups and data validation procedures.

5. User Manual

5.1. Landing Page

The landing page will provide an overview of the system and guide users to the sign-in and registration pages.

Dhaka University Swimming Pool

About us

Mission

Our mission at Aqua Piscinas CR is to offer high-quality pool construction and remodeling solutions in La Fortuna de San Carlos, Costa Rica. We strive to provide our clients with exceptional services, using innovative techniques and materials to create durable aquatic environments. We work closely with our clients to ensure their satisfaction and aim to deliver outstanding results. Our goal is to use cutting-edge methods and materials to craft long-lasting, one-of-a-kind aquatic environments.

Vision

Our mission at Aqua Piscinas CR is to offer high-quality pool construction and remodeling solutions in La Fortuna de San Carlos, Costa Rica. We strive to provide our clients with exceptional services, using innovative techniques and materials to create durable aquatic spaces. We work closely with our clients to ensure their satisfaction and aim to deliver outstanding results. Our goal is to use cutting-edge methods and materials to craft long-lasting, one-of-a-kind aquatic environments.

Dive into our gallery and explore the splendor of aquatic moments captured in time.

Courses

Contact us

Contact Us

- ✉ Info@example.com
- 📞 +506 6233 5000
- 📍 Dhaka University Campus, Dhaka-1000.

Quick Links

- Home
- About
- Courses
- Schedule
- Contact

Follow Us

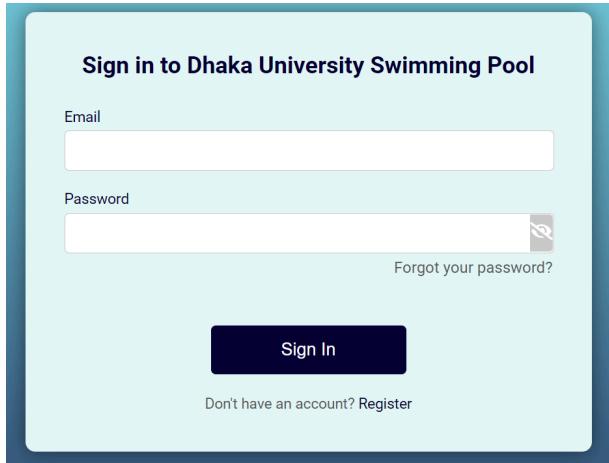
- Facebook
- Twitter
- Instagram
- LinkedIn

© 2024 Dhaka University Swimming Pool. All rights reserved

Figure: Landing page

5.2. Sign In

Users can sign in using their credentials to access their personalized dashboard.

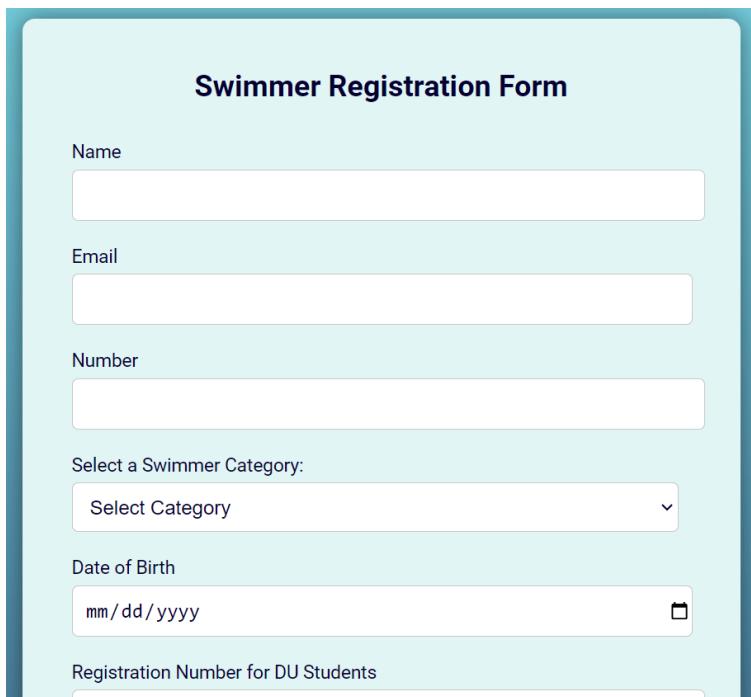


The image shows a login form titled "Sign in to Dhaka University Swimming Pool". It features two input fields: "Email" and "Password". Below the password field is a "Forgot your password?" link and a small icon. A "Sign In" button is centered at the bottom, and a "Don't have an account? Register" link is located just below it.

Figure: SignIn page

5.3. User Registration

New users can register for an account by providing necessary details and selecting a membership plan.



The image displays a registration form titled "Swimmer Registration Form". It includes fields for "Name", "Email", and "Number". There is a dropdown menu labeled "Select Category" under the heading "Select a Swimmer Category:". The "Date of Birth" field is formatted as "mm/dd/yyyy" with a calendar icon. At the bottom, there is a field for "Registration Number for DU Students".

Figure: Registration Page

5.4. Schedule Management

Users can view the and download schedule of activities and register for courses at their convenience.

Figure: SchedulePage

5.5. Financial Management

Users can make payments for services that is enrollment of courses and view their transaction history. Administrators can access financial reports through admin panel.

My Courses

Butterfly

Price: 2000 BDT

Duration: 3 months

Pay Now

Demo

Support FAQ Offers Login

CARDS MOBILE BANKING NET BANKING

Bkash BGMI

AB+ Oxy

Gok wallet Cellyn

M Cash

OTP Page

Do not press browser back or forward button while you are in payment page

Payment Summary	
Please review the following detail for this transaction:	
Amount:	2000.00
Invoice number:	24061394625f3YKK3kWFSAi3Do
Description:	Products

Enter Card Information	
OTP: <input type="text"/> <div style="display: flex; justify-content: space-around; width: 100%;"> Success Failed </div> <div style="display: flex; justify-content: space-around; width: 100%;"> Success with risk </div>	Your entered card information could not be corrupted or become known to the third party, as all transmitted data is encrypted by the SSL protocol. Note 1. For VISA and MC, look at the back side of your Card to find 3-digit CVV2/ CVC2. For AMEX, look at the upper right corner of the front side of your Card to find 4-digit CSC. 2. The cardholder's name should be entered just as it's written on the card.





SSLCOMMERZ TESTBOX GATEWAY (NO CARD INFORMATION WILL BE SAVED AND DUMMY CARD WILL BE DISPLAYED IN REPORTS)

Figure: Financial management through SSLCOMMERZ

5.6. Event Management

Users can participate in swimming events which are created by Admin. Users can download ticket as PDF.

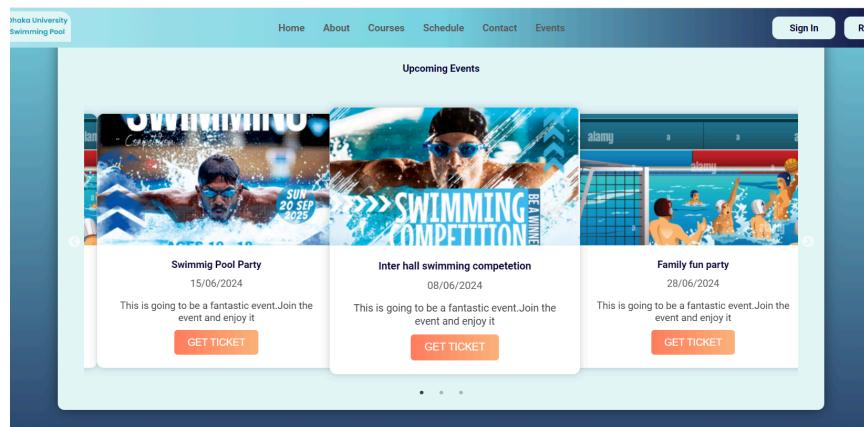
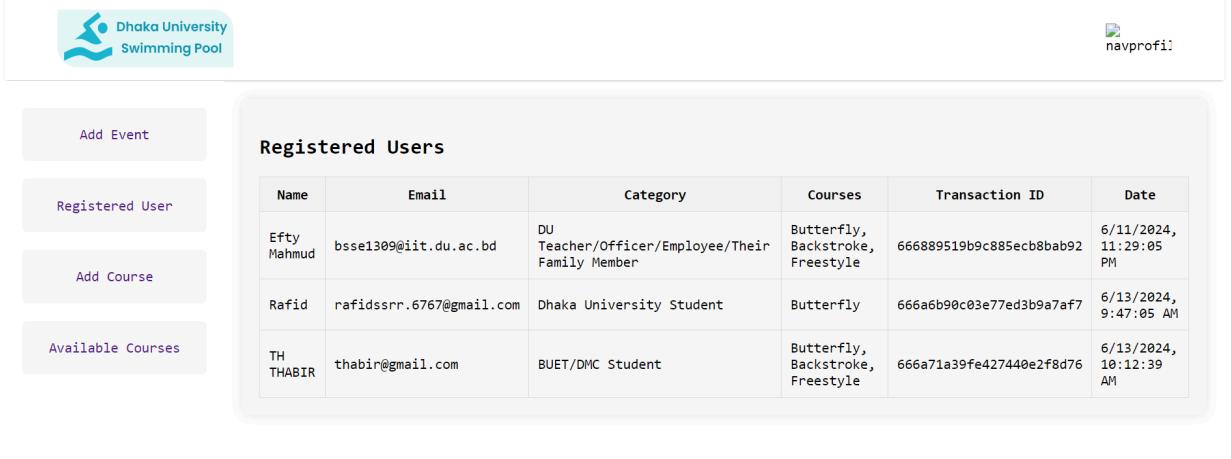


Figure: Event Module

5.6. Admin Panel

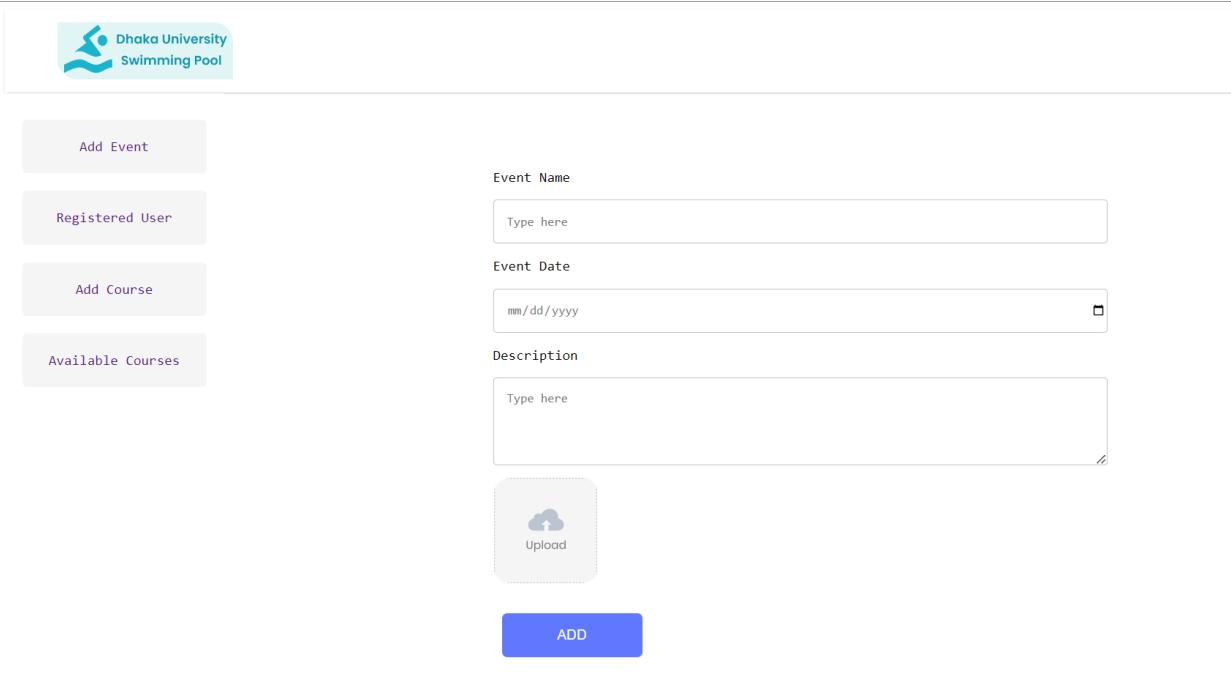
Admin panel is for to see registered users and their info, Admin can add different swimming events, can add swimming courses and can also check available courses. The UI's are given below:



The screenshot shows the 'Registered Users' section of the admin panel. On the left, there are four buttons: 'Add Event', 'Registered User' (highlighted in blue), 'Add Course', and 'Available Courses'. The main area displays a table titled 'Registered Users' with the following data:

Name	Email	Category	Courses	Transaction ID	Date
Efty Mahmud	bsse1309@iit.du.ac.bd	DU Teacher/Officer/Employee/Their Family Member	Butterfly, Backstroke, Freestyle	666889519b9c885ecb8bab92	6/11/2024, 11:29:05 PM
Rafid	rafidssrr.6767@gmail.com	Dhaka University Student	Butterfly	666a6b90c03e77ed3b9a7af7	6/13/2024, 9:47:05 AM
TH THABIR	thabir@gmail.com	BUET/DMC Student	Butterfly, Backstroke, Freestyle	666a71a39fe427440e2f8d76	6/13/2024, 10:12:39 AM

Figure: Admin can see registered users and their info



The screenshot shows the 'Add Event' form. On the left, there are four buttons: 'Add Event' (highlighted in blue), 'Registered User', 'Add Course', and 'Available Courses'. The right side contains fields for event details:

- 'Event Name' input field with placeholder 'Type here'.
- 'Event Date' input field with placeholder 'mm/dd/yyyy' and a calendar icon.
- 'Description' text area with placeholder 'Type here'.
- A dashed rectangular area labeled 'Upload' with a cloud icon.
- A blue 'ADD' button at the bottom.

Figure: Admin can upload event

[Add Event](#)

[Registered User](#)

[Add Course](#)

[Available Courses](#)

Course Name

Description

Price

Duration

Upload

ADD

Figure: Admin can add swimming courses

[Add Event](#)

[Registered User](#)

[Add Course](#)

[Available Courses](#)

Available Courses

Name	Description	Price	Duration	Image	Actions
Butterfly	Butterfly swimming is a beautiful and dynamic stroke characterized by... See More	2000	2 month		Remove
Freestyle	Freestyle swimming, often regarded as the most straightforward yet versatile... See More	2000	3 month		Remove
Breaststroke	Backstroke, known for its graceful and relaxed style, is one of the four primary strokes used in competitive swimming. Swimmers execute the backstroke while lying on their backs, facing upwards, and propelling themselves through the water with an alternating arm motion. Unlike other strokes, the backstroke allows swimmers to breathe freely as their face remains above the water surface throughout the stroke. The stroke's smooth and continuous motion requires coordination between the arms and legs, with swimmers performing a flutter kick to provide propulsion. Backstroke emphasizes maintaining a steady rhythm and balance to optimize speed and efficiency.. See Less	2000	2 month		Remove
Backstroke	Backstroke, known for its graceful and relaxed style, is one... See More	2500	2 month		Remove

Figure: Admin can see current available courses

6. Challenges Faced

6.1. Dependency Related Issues

Managing dependencies and ensuring compatibility between different libraries and frameworks posed a challenge during development.

6.2. Integration of Components

Integrating various components and ensuring smooth communication between the frontend and backend required careful planning and testing.

6.3. Maintainability in Backend Code

Ensuring the maintainability of the backend code involved implementing best practices for code organization, documentation, and testing.

7. Conclusion and Future Work

The Dhaka University Swimming Pool Management System represents a significant step towards modernizing and optimizing the management of swimming pool facilities at Dhaka University. By implementing this system, we aim to enhance user experience, improve operational efficiency, and contribute to a safer and more enjoyable swimming experience for all members of the university community.

8. References

1. *W3schools react tutorial.* Retrieved from
<https://www.w3schools.com/REACT/DEFAULT.ASP>
2. *SSLCOMMERZ intregation.* Retrieved from
<https://github.com/sslcommerz/SSLCommerz-NodeJS>