Application Framework Assignment 1

Node(Express)

Arachchi D.S.U - IT21182914

University Timetable Management System Documentation

1. Introduction

The University Timetable Management System is a software solution designed to facilitate the efficient scheduling and management of academic activities within a university or educational institution. This documentation provides an overview of the system's architecture, functionality, and implementation details based on the provided codebase.

2. System Overview

The system comprises several modules responsible for managing different aspects of the university timetable:

- **Authentication and Authorization**: Handled by the `authRoutes` module, which manages user authentication and authorization using JSON Web Tokens (JWT).

- **Course Management**: Managed through the `courseRoutes` module, enabling CRUD operations for courses.

- **Class Scheduling**: Implemented in the `classSessionRoutes` module, facilitating the scheduling of class sessions.

- **Booking Management**: Managed via the `bookingRoutes` module, allowing users to book rooms for classes and other events.

- **Room Management**: Handled by the `roomRoutes` module, enabling the management of rooms and their attributes.

- **Resource Management**: Implemented in the `resourceRoutes` module, facilitating the management of resources required for classes.

- **Enrollment Management:** Managed through the `enrollmentRoutes` module, allowing students to enroll in courses.

- **Notification System**: Facilitated by the `notificationRoutes` module, enabling the system to send notifications to users regarding timetable updates and other relevant information.

3. User Roles and Permissions

The system supports different user roles, each with specific permissions:

- Authenticated Users: Have access to general functionalities such as course enrollment and timetable viewing.
- Administrators: Have elevated privileges to perform administrative tasks like managing users, courses, and rooms.
- Faculty Members: Can manage their teaching schedules and view course-related information.
- Students: Can enroll in courses, view their schedules, and receive notifications.

4. Features

The system boasts the following features:

- **Secure Authentication**: Users can securely authenticate using JWT tokens.
- **Flexible Course Management**: Enables admin to do the creation, updating, and deletion of courses.
- **Efficient Class Scheduling**: Facilitates the scheduling of class sessions while considering various constraints.
- **Room and Resource Management**: Allows the management of rooms and resources required for classes.
- **User-friendly Booking System**: Enables users to book rooms for classes and events seamlessly.
- **Streamlined Enrollment Process**: Provides a straightforward process for students to enroll in courses.
- **Notification System**: Keeps users informed about timetable updates and other relevant information.

5. Technology Stack

The system is built using the following technologies:

- **Backend:** Node.js with Express.js framework
- **Database:** MongoDB for data storage
- **Authentication:** JSON Web Tokens (JWT) for user authentication

6. Installation and Setup

To set up the system:

1. Clone the project repository.
2. Install dependencies using **npm install**.
3. Configure environment variables, including the MongoDB URI and JWT Secret Key.
4. Ensure MongoDB is installed and running.
5. Run the Node.js server using `npm start`

7. Usage

❖University Timetable Management System API

1.Authentication

- **Register a New User**

URL: **http://localhost:8080/auth/register**

Method: `POST`

Description: Register a new user.

Request Body:

```
{
  "username": "Dilan Shanuka",
  "email": "dilanshanuka99@gmail.com",
  "password": "123456"
}
```

Response:

```
{
    "message": "User registered successfully",
    "user": {
        "username": "Dilan Shanuka",
        "email": "dilanshanuka99@gmail.com",
        "password": "$2a$10$OP85AGZlbAHCSmoHxx6.pej9KXqE1DI7nAWZ4kUSFeH45IFZOz5VG",
        "role": "student",
        "_id": "65ffcdf1cd5f9971d844daa5",
        "__v": 0
    }
}
```

Validation:

Email should be unique.

```
{
    "message": "Email is already registered"
}
```

- Login

URL: **http://localhost:8080/auth/login**

Method: **POST**

Description: Login with existing credentials.

Request Body:

```
{
  "email": "dilanshanuka999@gmail.com",
  "password": "123456"
}
```

Response:

```
{
    "message": "Login successful",
    "token":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiI2NWY5OTI0MDFiZjkwMzA4OGE0YWNkMWIi
```

```
LCJyb2xlIjoiYWRtaW4iLCJpYXQiOjE3MTEyNjM0MjksImV4cCI6MTcxMTI2NzAyOX0.zGh3E17IxohTQkzKkj
avwNbGlKRHGEOkJp6rdhJ4DL4"
}
```


2.Courses

- Create a New Course

URL: **http://localhost:8080/courses**

Method: **POST**

Authorization :|

Bearer Token - token(admin)

Description: Create a new course.

Request Body:

```
{
  "courseName": "Computer Science",
  "courseCode": "CS101",
  "description": "Introduction to Computer Science",
  "credits": 3,
  "schedule": "Monday, Wednesday, Friday 10:00 AM - 11:30 AM"
}
```

Response:

```
{
    "message": "Course created successfully",
    "course": {
        "courseName": "Computer Science",
        "courseCode": "CS101",
        "description": "Introduction to Computer Science",
        "credits": 3,
        "classSessions": [],
        "schedule": "Monday, Wednesday, Friday 10:00 AM - 11:30 AM",
        "_id": "65ffd139cd5f9971d844dab1",
        "__v": 0
    }
}
```

- Get All Courses

URL: **http://localhost:8080/courses**

Method: **GET**

Authorization :|

Bearer Token - token(admin)

Description: Get all available courses.

Response: Array of course objects


```
[
    {
        "faculty": {
            "_id": "65f992401bf903088a4acd1b",
            "name": "Dilan Shanuka"
        },
        "classSessions": [],
        "_id": "65f9cbd8f518bf6dcac4274a",
        "courseName": "SE",
        "courseCode": "SE3020",
        "description": "SE",
        "credits": 4,
        "__v": 0
    }
```


Array of course objects

- Update a Course

URL: **http://localhost:8080/courses/:id**

Method: **PUT**

Authorization :|

Bearer Token - token(admin)

Description: Update an existing course.

Request Parameters **: (string): Course ID**

Request Body:


```
{
  "courseName": "Computer Science 1",
  "courseCode": "CS101",
  "description": "Introduction to Computer Science",
  "credits": 3,
  "schedule": "Monday, Wednesday, Friday 10:00 AM - 11:30 AM"
```

```
}
```

Response:

```
{
    "message": "Course updated successfully",
    "course": {
        "_id": "65ffd139cd5f9971d844dab1",
        "courseName": "Computer Science 1",
        "courseCode": "CS101",
        "description": "Introduction to Computer Science",
        "credits": 3,
        "classSessions": [],
        "schedule": "Monday, Wednesday, Friday 10:00 AM - 11:30 AM",
        "__v": 0
    }
}
```

- Delete a Course

URL: **http://localhost:8080/courses/:id**

Method: **`DELETE`**

Description: Delete an existing course.

Request Parameters:

`id` (string): Course ID

Response:

```
{
    "message": "Course deleted successfully"
}
```

3.Assign Faculty Member to Courses

- Assign Faculty members to course

URL: **http://localhost:8080/courses/course:id/assign-faculty**

Method: **`PUT`**

`Authorization :|`

`Bearer Token - token(admin)`

Description: Assign Faculty Member to Courses

Request Body:

```
{
  "facultyId": "65f992401bf903088a4acd1b",
  "facultyName": "Dilan Shanuka"
}
```

Response:

```
{
    "message": "Faculty assigned to course successfully",
    "course": {
        "faculty": {
            "_id": "65f992401bf903088a4acd1b",
            "name": "Dilan Shanuka"
        },
        "_id": "65fb2637a55e4fd2e7750136",
        "courseName": "Mathematics",
        "courseCode": "MATH101",
        "description": "Introduction to Mathematics",
        "credits": 3,
        "classSessions": [],
        "schedule": "Mon-Wed-Fri 10:00 AM - 11:30 AM",
        "__v": 0
    }
}
```

4.Class Sessions (Timetables)

- Create a New Class Session

URL: **http://localhost:8080/class-sessions**

Method: **POST**

Description: Create a new class session.

Request Body:

```
{
  "course": "65fb2637a55e4fd2e7750136",
  "dateTime": "2024-03-25T10:00:00",
  "faculty": "65f9b2e1bc2bcc76f1873ed4",
  "location": "Room 105",
```

```
  "students": ["65ffcdf1cd5f9971d844daa5"],
  "type": "ClassSession"
}
```


Response:


```
{
    "message": "Class session created successfully",
    "classSession": {
        "course": "65fb2637a55e4fd2e7750136",
        "dateTime": "2024-03-25T04:30:00.000Z",
        "faculty": "65f9b2e1bc2bcc76f1873ed4",
        "location": "Room 105",
        "students": [
            "65ffcdf1cd5f9971d844daa5"
        ],
        "_id": "6600191aba8d31b4038f5c9a",
        "__v": 0
    }
}
```


- Get All Class Sessions for a student

URL: **http://localhost:8080/class-sessions/timetable/student:id**

Method: **`GET`**

Description: Get all class sessions for a student.

Response: Array of class session objects


```
[
    {
        "_id": "65fa950303c72914f467bda1",
        "course": "65f9cbd8f518bf6dcac4274a",
        "dateTime": "2024-03-25T04:30:00.000Z",
        "faculty": "65f992401bf903088a4acd1b",
        "location": "Room 101",
        "students": [
            "65fa7414c16a47e88723d723"
        ],
        "__v": 0
    },
    {
        "_id": "65fa9981832942b97f4f88a6",
```

```
        "course": "65f9cbd8f518bf6dcac4274a",
        "dateTime": "2024-03-25T04:30:00.000Z",
        "faculty": "65f992401bf903088a4acd1b",
        "location": "Room 101",
        "students": [
            "65fa7414c16a47e88723d723",
            "65fa7414c16a47e88723d723"
        ],
        "__v": 0
    }
]
```


- Update a Class Session

URL: **http://localhost:8080/class-sessions/classsession:id**

Method: **PUT**

Description: Update an existing class session.

Request Parameters:

`id` (string): Class session ID

Request Body:


```
{
    "dateTime": "2024-03-26T11:00:00",
    "location": "Room 102"
}
```


Response:


```
{
    "message": "Class session updated successfully",
    "classSession": {
        "_id": "6600191aba8d31b4038f5c9a",
        "course": "65fb2637a55e4fd2e7750136",
        "dateTime": "2024-03-26T05:30:00.000Z",
        "faculty": "65f9b2e1bc2bcc76f1873ed4",
        "location": "Room 102",
        "students": [
```

```
            "65ffcdf1cd5f9971d844daa5"
        ],
        "__v": 0
    }
}
```

- Delete a Class Session

URL: **http://localhost:8080/class-sessions/classsession:id**

Method: **`DELETE`**

Description: Delete an existing class session.

Request Parameters:

`id` (string): Class session ID

Response:

```
{
    "message": "Class session deleted successfully"
}
```

5.Classrooms

- Create a New Room

URL: **http://localhost:8080/rooms**

Method: **`POST`**

Description: Create a new room.

Request Body:

```
{
    "name": "Room 101",
    "description": "Large classroom with projector",
    "capacity": 50
}
```

Response:


```
{
    "message": "Room created successfully",
    "room": {
        "name": "Room 101",
        "description": "Large classroom with projector",
        "capacity": 50,
        "isAvailable": true,
        "_id": "660029ad9758631ee526df69",
        "__v": 0
    }
}
```


- Get All Rooms

URL: **http://localhost:8080/rooms**

Method: **`GET`**

Description: Get all available rooms.

Response: Array of room objects


```
[
    {
        "_id": "65fa63f50551b6e9939ac08b",
        "name": "Updated Room A",
        "description": "Large classroom with projector",
        "capacity": 60,
        "isAvailable": true,
        "__v": 0
    }
]
```


- Create a booking

URL: **http://localhost:8080/bookings/create**
Method: **`POST`**
Description: Create a new room.
Request Body:

```
{
  "roomId": "660029ad9758631ee526df62",
  "startTime": "2024-03-27T10:00:00",
  "endTime": "2024-03-27T12:00:00"
}
```

Responses:

```
{
    "message": "Booking created successfully",
    "booking": {
        "roomId": "660029ad9758631ee526df62",
        "startTime": "2024-03-28T04:30:00.000Z",
        "endTime": "2024-03-28T06:30:00.000Z",
        "_id": "66003bf5237598cdcbe871da",
        "__v": 0
    }
}
```

```
{
    "message": "Booking overlaps with existing bookings"
}
```

6.Resources

- Create a New Resource

URL: **http://localhost:8080/resources**

Method: **POST**

Description: Create a new resource.

Request Body:

```
{
    "name": "Projector",
    "description": "High-quality projector for presentations",
    "quantity": 5,
    "isAvailable": true
}
```

Response:

```
{
    "message": "Resource created successfully",
    "resource": {
        "name": "Projector",
        "description": "High-quality projector for presentations",
        "quantity": 5,
        "isAvailable": true,
        "_id": "66003c53237598cdcbe871dd",
        "__v": 0
    }
}
```

- Get All Resources

URL: **http://localhost:8080/resources**

Method: **GET**

Description: Get all available resources.

Response: Array of resource objects

```
[
    {
        "_id": "65fa652b0551b6e9939ac090",
        "name": "Projector 2",
        "description": "High-quality projector for presentations",
        "quantity": 5,
        "isAvailable": true,
        "__v": 0
    }
]
```

7.Enrollments

- ## Enroll in a Course

URL: **http://localhost:8080/enrollments/enroll**

Method: **`POST`**

Description: Enroll a student into a course.

Request Body:

```
{
    "studentId": "65ffcdf1cd5f9971d844daa5",
    "studentName": "Dilan Shanuka",
    "courseId": "65f9cbd8f518bf6dcac4274a",
    "courseName": "SE"
}
```

Response:

```
{
    "message": "Enrollment successful",
    "enrollment": {
        "student": "65ffcdf1cd5f9971d844daa5",
        "course": "65f9cbd8f518bf6dcac4274a",
        "studentName": "Dilan Shanuka",
        "_id": "66003e88237598cdcbe871e9",
        "__v": 0
    }
}
```

- ## Get Student Enrolled Course

URL: **http://localhost:8080/enrollments/timetable/student:id**

Method: **`GET`**

Description: Get all available courses assigned to a student.

Response: Array of resource objects

```
[
    {
        "courseName": "Software Engineering",
        "schedule": "Monday, Wednesday, Friday 10:00 AM - 11:30 AM"
    }
]
```


- Get all the enrollments to a course

URL: **http://localhost:8080/enrollments/course/course:id/enrollments**

Method: **`GET`**

Description: Get all available courses assigned to a student.

Response: Array of resource objects


```
[
    {
        "_id": "6600409e237598cdcbe871ff",
        "student": {
            "_id": "65ffcdf1cd5f9971d844daa5",
            "username": "Dilan Shanuka",
            "email": "dilanshanuka99@gmail.com",
            "password":
"$2a$10$OP85AGZlbAHCSmoHxx6.pej9KXqE1DI7nAWZ4kUSFeH45IFZOz5VG",
            "role": "student",
            "__v": 0
        },
        "course": "66003ffd237598cdcbe871f1",
        "studentName": "Dilan Shanuka",
        "__v": 0
    }
]
```


- Remove a student from a course

URL: **http://localhost:8080/enrollments/course/course:id/enrollments**

Method: **`PUT`**

Description: Update an existing class session.

Request Parameters: `id` (string): Course ID

Request Body:

```
{
    "action": "remove",
    "studentId": "65fa7414c16a47e88723d723"
}
```

Response:

```
{
    "message": "Student removed from course"
}
```

8.Notifications

- Create a Notification

URL: **http://localhost:8080/notifications/create**

Method: **POST**

Authorization :|

Bearer Token - token(admin or faculty)

Description: Create a new notification.

Request Body:

```
{
    "type": "Timetable Updates",
    "message": "Timetable for Course SE is going to update in next week"
}
```

Response:


```
    "message": "Notifications created successfully",
    "notifications": [
        {
            "type": "Timetable Updates",
            "message": "Timetable for Course SE is going to update in next week",
            "read": false,
            "_id": "660017a2ba8d31b4038f5c84",
            "timestamp": "2024-03-24T12:08:02.163Z",
            "__v": 0
        }
]
```


- Get All Notifications

URL: **`http://localhost:8080/notifications/notifications`**

Method: **`GET`**

Description: Get all notifications.

Response: Array of notification objects


```
[
    {
        "_id": "65fb37699603835ef6668139",
        "type": "Timetable Update",
        "message": "Timetable for Course SE is going to update in next week",
        "read": false,
        "timestamp": "2024-03-20T19:22:17.946Z",
        "__v": 0
    }
]
```


9. Conclusion

The University Timetable Management System provides a robust solution for universities and educational institutions to effectively manage their academic schedules and

resources. By automating various processes, the system aims to streamline timetable management and enhance overall efficiency.