BÁO CÁO DẬN - TUẦN 8

Họ và Tên: Lý Chí Công

MSV: B22DCCN088

Bộ môn: Thực tập cơ sở

1. Thông tin dự án:

Tên dự án: Hệ thống quản lý sân bóng
Công nghệ sử dụng: ReactJS (frontend), Node.js + Express.js (backend), MongoDB (database)
Mục tiêu chung: Xây dựng hệ thống quản lý sân bóng, bao gồm đặt sân, thanh toán, quản lý lịch sử sử dụng.

---

2. Mục tiêu tuần 8

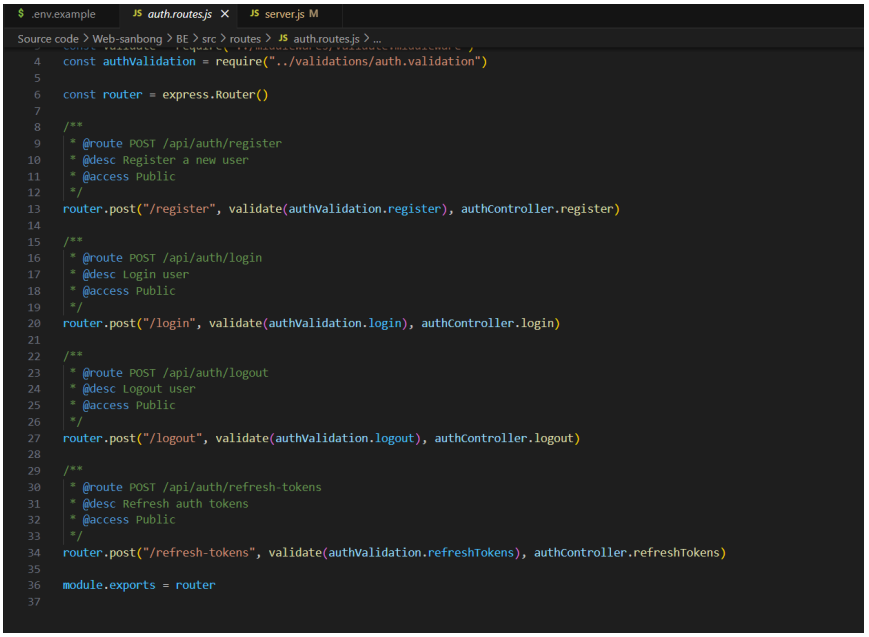
- Hoàn thiện các API cơ bản cho các đối tượng: users, bookings, fields, field_status.
- Tích hợp hệ thống xác thực và phân quyền người dùng bằng **JWT**.
- Cấu hình **CORS** cho phép frontend truy cập.
- Tìm hiểu cách triển khai xác thực qua email và tích hợp cổng thanh toán.

---

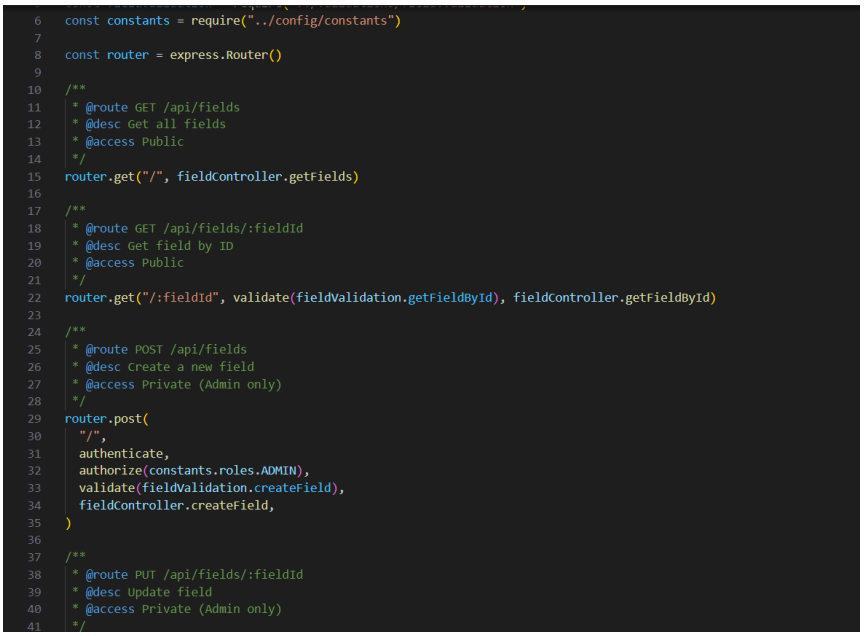
3. Kết quả đạt được

API cơ bản đã hoàn thiện

- **Users**: Đăng ký, đăng nhập, cập nhật thông tin, tìm kiếm, xóa.



Fields: Tạo, cập nhật, xóa, lấy danh sách sân.



Field status: Theo dõi trạng thái sân theo ngày/giờ.

```
/**
 * @route GET /api/field-status/date/:date
 * @desc Get field status by date
 * @access Public
 */
router.get(
  "/date/:date",
  validate(fieldStatusValidation.getFieldStatusByDate),
  fieldStatusController.getFieldStatusByDate,
)

/**
 * @route GET /api/field-status/:fieldId/date/:date
 * @desc Get field status by field ID and date
 * @access Public
 */
router.get(
  "/:fieldId/date/:date",
  validate(fieldStatusValidation.getFieldStatusByFieldAndDate),
  fieldStatusController.getFieldStatusByFieldAndDate,
)

/**
 * @route PUT /api/field-status/:fieldId/date/:date
 * @desc Create or update field status
 * @access Private (Admin/Manager)
 */
router.put(
  "/:fieldId/date/:date",
  authenticate,
  authorize(constants.roles.ADMIN, constants.roles.MANAGER),
  validate(fieldStatusValidation.createOrUpdateFieldStatus),
  fieldStatusController.createOrUpdateFieldStatus,
)
```

Bookings: Đặt sân, cập nhật lịch đặt, hủy đặt sân.

```
/**
 * @route GET /api/bookings
 * @desc Get all bookings
 * @access Private (Admin/Manager)
 */
router.get("/", authenticate, authorize(constants.roles.ADMIN, constants.roles.MANAGER), bookingController.getBookings)

/**
 * @route GET /api/bookings/:bookingId
 * @desc Get booking by ID
 * @access Private
 */
router.get("/:bookingId", authenticate, validate(bookingValidation.getBookingById), bookingController.getBookingById)

/**
 * @route GET /api/bookings/user/:userId
 * @desc Get bookings by user ID
 * @access Private
 */
router.get(
  "/user/:userId",
  authenticate,
  validate(bookingValidation.getBookingsByUser),
  bookingController.getBookingsByUser,
)

/**
 * @route POST /api/bookings
 * @desc Create a new booking
 * @access Private
 */
router.post("/", authenticate, validate(bookingValidation.createBooking), bookingController.createBooking)

/**
 * @route PUT /api/bookings/:bookingId
 * @desc Update booking
```

Phân quyền bằng JWT:

- Đã triển khai middleware xác thực và phân quyền người dùng (user, admin).

- Các route quan trọng chỉ cho phép truy cập khi có token hợp lệ.

```
const authenticate = async (req, res, next) => {
  try {
    const authHeader = req.headers.authorization

    if (!authHeader || !authHeader.startsWith("Bearer ")) {
      return apiResponse(res, 401, "Authentication required")
    }

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

    if (!token) {
      return apiResponse(res, 401, "Authentication required")
    }

    try {
      const decoded = jwt.verify(token, config.jwt.secret)

      const user = await userService.getUserById(decoded.sub)

      if (!user) {
        return apiResponse(res, 401, "Invalid token")
      }

      req.user = user
      next()
    } catch (error) {
      return apiResponse(res, 401, "Invalid token")
    }
  } catch (error) {
    next(error)
  }
}
```

Chưa hoàn thành

Xác thực người dùng qua Email (Mail Service): Chưa triển khai tính năng gửi email xác nhận hoặc đặt lại mật khẩu.

Tích hợp API thanh toán: Chưa viết được các API xử lý thanh toán qua ví điện tử hoặc ngân hàng.

---

4. Khó khăn và hướng giải quyết

Khó khăn	Hướng giải quyết
Thiết kế API thanh toán đòi hỏi tích hợp với bên thứ ba (e.g. Momo, ZaloPay)	Tạm hoãn, dự kiến sẽ xử lý sau khi hoàn thiện chức năng đặt sân
Chưa rõ cách gửi email xác nhận qua Node.js	Dự kiến dùng thư viện nodemailer và thiết lập tài khoản Gmail hoặc Mailtrap để kiểm thử
JWT cần xử lý nhiều trường hợp lỗi (hết hạn, token sai)	Đã tạo middleware xử lý token, sẽ tiếp tục tối ưu logic phản hồi lỗi

---

5. Mục tiêu tuần 9

- Triển khai xác thực qua email sử dụng nodemailer.
- Viết và hoàn thiện các API liên quan đến **thanh toán đơn giản** (giả lập).
- Kiểm thử toàn diện hệ thống booking: kiểm tra trùng lịch, trạng thái sân.
- Kết nối giao diện React với các API backend đã viết.
- Bổ sung phân trang và tìm kiếm nâng cao cho danh sách người dùng/sân.