Update authenticate.js as follows:

…

exports.verifyAdmin = (req, res, next) => {

    passport.authenticate("jwt", { session: false }, (err, user) => {

        if (err) {

            return next(err);

        }

        if (!user || !user.admin) {

            const err = new Error(

                "You are not authorized to perform this operation!"

            );

            err.status = 403; // Forbidden status code

            return next(err);

        }

        next();

    })(req, res, next);

};

Open dishRouter.js and updated the code

dishRouter.route('/')

…

.post(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {

…

 .put(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {

…

.delete(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {

…

dishRouter.route('/:dishId')

.post(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {

…

.put(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {

…

.delete(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {

dishRouter.route('/:dishId/comments')

….

.delete(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {

dishRouter.route('/:dishId/comments/:commentId')

…

.put(authenticate.verifyUser, (req, res, next) => {

        Dishes.findById(req.params.dishId)

            .then((dish) => {

                if (dish != null && dish.comments.id(req.params.commentId) != null

                    && dish.comments.id(req.params.commentId).author.equals(req.user.\_id)) { // kiểm tra user chỉ được sửa comment của chính họ

                    if (req.body.rating) {

                        dish.comments.id(req.params.commentId).rating = req.body.rating;

                    }

                    if (req.body.comment) {

                        dish.comments.id(req.params.commentId).comment = req.body.comment;

                    }

                    dish.save()

                        .then((dish) => {

                            Dishes.findById(dish.\_id)

                                .populate('comments.author')

                                .then((dish) => {

                                    res.statusCode = 200;

                                    res.setHeader('Content-Type', 'application/json');

                                    res.json(dish);

                                })

                        }, (err) => next(err));

                }

                else if (dish == null) {

                    err = new Error('Dish ' + req.params.dishId + ' not found');

                    err.status = 404;

                    return next(err);

                }

                else if (dish.comments.id(req.params.commentId) == null) {

                    err = new Error('Comment ' + req.params.commentId + ' not found');

                    err.status = 404;

                    return next(err);

                } else {

                    err = new Error('you are not authorized to update this comment!');

                    err.status = 403;

                    return next(err);

                }

            }, (err) => next(err))

            .catch((err) => next(err));

    })

    .delete(authenticate.verifyUser, (req, res, next) => {

        Dishes.findById(req.params.dishId)

            .then((dish) => {

                if (dish != null && dish.comments.id(req.params.commentId) != null

                    && dish.comments.id(req.params.commentId).author.equals(req.user.\_id)) {

                    dish.comments.id(req.params.commentId).remove();

                    dish.save()

                        .then((dish) => {

                            Dishes.findById(dish.\_id)

                                .populate('comments.author')

                                .then((dish) => {

                                    res.statusCode = 200;

                                    res.setHeader('Content-Type', 'application/json');

                                    res.json(dish);

                                })

                        }, (err) => next(err));

                }

                else if (dish == null) {

                    err = new Error('Dish ' + req.params.dishId + ' not found');

                    err.status = 404;

                    return next(err);

                }

                else if (dish.comments.id(req.params.commentId) == null) {

                    err = new Error('Comment ' + req.params.commentId + ' not found');

                    err.status = 404;

                    return next(err);

                } else {

                    err = new Error('you are not authorized to delete this comment!');

                    err.status = 403;

                    return next(err);

                }

            }, (err) => next(err))

            .catch((err) => next(err));

    });

Open users.js and updated the code

/\* GET users listing. \*/

router.get('/', authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {

  User.find({})

    .then((users) => {

      res.statusCode = 200;

      res.setHeader('Content-Type', 'application/json');

      res.json(users);

    }, (err) => next(err))

    .catch((err) => next(err));

});