```
/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        users.controller.js
/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
const User = require('../../models/user');
const bcrypt = require('bcryptjs');

/* Get list of users */
/* Returns a list of users */
exports.getUsers = async (req, res) => {
    console.log('/admin/users');
    /* Finds all users in db, includes all relevant fields except for hash */
    await User.find()
        .select('username admin apartmentId roomId status')
        .then(users => res.status(200).json({
            users,
        }))
        .catch(err => {
            res.statusMessage = err;
            res.status(400).end();
        });
};

exports.getUser = (req, res) => {
    console.log(`admin/user/${req.params.id}`);
    User.findOne({username: req.params.id})
        .select('username admin apartmentId roomId status')
        .then(user => res.status(200).send(user));
};

/* Add user */
/* Validates and adds user to db */
exports.registerUser = async (req, res, next) => {
    console.log('/admin/users/register');

    if (!req.body.admin && (!req.body.apartmentId || !req.body.roomId)) {
        console.info(req.body.admin);
        next('Error: Non-admin user must be assigned Apartment and room.');
    } else if (req.body.admin && (req.body.apartmentId === null || req.body.roomId === null)) {
        console.info(req.body.admin);
        next('Error: Admin can not be assigned to a Apartment and room.');
    }

    await bcrypt.hash(req.body.password, 10).then(hash => {
        const user = new User({
            username: req.body.username,
            hash,
            admin: req.body.admin,
            apartmentId: req.body.apartmentId,
            roomId: req.body.roomId,
            status: req.body.status,
        });
        user
            .save()
            .then(result => {
                if (result) {
                    console.log('User created!');
                    return res.status(201).json({
                        data: 'User successfully created',
                    });
                }

                next('Error registering user');
            })
            .catch(err => {
                next(err);
            });
    });
};

/* Modify user */
/* Validates and adds changes to user to db */
exports.modifyUser = async (req, res) => {
    console.log('/admin/user/id/modify');
    /* Checks if a password change was requested. */
    /* If so, hash the new password */
```

```
    let hashed = '';
    if (req.body.password) {
        await bcrypt.hash(req.body.password, 10).then(hash => {
            hashed = hash;
        });
    }

    const updates = {
        hash: hashed,
        admin: req.body.admin,
        apartmentId: req.body.apartmentId,
        roomId: req.body.roomId,
        status: req.body.status,
    };
    console.log(Object.keys(updates));
    console.log(updates[0]);
    console.log(Object.values(updates));

    for (const key in updates) {
        if (updates[key] === null || updates[key] === undefined || updates[key] === '') {
            console.log(`deleted ${updates[key]}`);
            delete updates[key];
        }
    }

    await User.updateOne(
        {username: req.params.id},
        {$set: {...updates}},
        {new: true},
    ).then(update => {
        if (!update) {
            res.statusMessage = 'Error updating user';
            res.status(400).end();
        }

        return res
            .status(201)
            .json({data: `Successfully updated user ${req.params.id}`});
    });
};

/* Delete user */
/* Validates and deletes user in db */
exports.deleteUser = async (req, res) => {
    console.log('/user/:id/delete');
    await User.deleteOne({username: req.params.id})
        .then(err => {
            if (!err.deletedCount) {
                res.statusMessage = 'Error deleting user';
                res.status(400).end();
            }

            return res.status(200).json({
                data: `User ${req.params.id} deleted successfully`,
            });
        })
        .catch(err => {
            res.statusMessage = err;
            res.status(400).end();
        });
};


/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        cages.controller.js
/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
const Cage = require('../../models/cage');

/* Get list of cages */
/* Returns a list of cages */
exports.getCages = async (req, res) => {
    await Cage.find()
        .then(cages => res.status(200).json({
            data: cages,
        }))
        .catch(err => {
            res.statusMessage = err;
```

```javascript
            res.status(400).end();
        });
};

/* Get specified cage */
/* Returns a single cage object */
exports.getCage = async (req, res) => {
    console.info(`>>/admin/cage/${req.params.cageId}`);
    await Cage
        .findOne({id: req.params.cageId})
        .then(cage => res.status(200).send(cage))
        .catch(err => res.status(400).send(err));
};

/* Add cage */
/* Validates and adds cage to db */
exports.registerCage = async (req, res) => {
    const cage = new Cage({
        id: req.body.id,
    });
    cage
        .save()
        .then(cage => {
            if (cage) {
                return res.status(200).json({data: 'Cage successfully added'});
            }

            res.statusMessage = 'Error adding cage';
            res.status(400).end();
        })
        .catch(err => {
            res.statusMessage = err;
            res.status(400).end();
        });
};

/* Update cage */
/* Validates and adds changes to cage in db */
exports.updateCage = (req, res) => {
    Cage.where({id: req.params.id})
        .findOne()
        .then(cage => {
            const statusChange = cage.get('status') !== req.body.status;

            if (!cage || !statusChange) {
                res.statusMessage = 'Error updating cage';
                res.status(400).end();
            } else {
                const updates = {
                    status: req.body.status,
                    lastUsed: Date.now(),
                };

                cage
                    .updateOne(
                        {id: req.params.id},
                        {$set: {...updates}},
                        {new: true},
                    )
                    .then(() => res.status(200).json({
                        data: `successfully updated box ${req.params.id} to ${req.body.status}`,
                    }))
                    .catch(err => {
                        res.statusMessage = err;
                        res.status(400).end();
                    });
            }
        });
};

/* Delete cage */
/* Validates and deletes cage in db */
exports.deleteCage = async (req, res) => {
    await Cage.deleteOne({id: req.params.id})
        .then(del => {
            if (del) {
                return res.status(200).json({
                    data: 'Cage deleted successfully',
                });
            }

            res.statusMessage = 'Error deleting cage';
            res.status(400).end();
        })
        .catch(err => res.status(404).json({data: err.message}));
};

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        boxes.controller.js
/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

const Cage = require('../../models/cage');
const User = require('../../models/user');
/* Get list of boxes in a cage */
/* Returns a list of boxes for specified cage */
exports.getBoxes = async (req, res, next) => {
    try {
        Cage.findOne({id: req.params.id})
            .populate({
                path: 'boxes.assignedUser',
                select: 'username -_id',
            })
            .exec((err, cages) => {
                if (err) {
                    console.log(err);
                    return next(err);
                }

                console.log(cages.boxes);
                return res.status(200).send(cages.boxes);
            });
    } catch (err) {
        console.log(err);
        next(err);
    }
};

/* Add box to a cage */
/* Validates and adds a box to specified cage */
exports.addBox = async (req, res, next) => {
    console.info(`/admin/cage/${req.params.cageId}/box/${req.params.boxId}/add`);

    try {
        Cage
            .updateOne(
                {id: req.params.cageId},
                {$addToSet: {boxes: {box: req.params.boxId}}},
            )
            .exec((err, result) => {
                console.log(`>>result: ${result.nModified}`);
                console.log(`>>err: ${err}`);
                if (err || result.nModified === 0) {
                    return next('Error adding box');
                }

                res.status(200).send(`Successfully added ${req.params.boxId} to cage ${req.params.cageId}`);
            });
    } catch (err) {
        console.log(err);
        next(err);
    }
};

/* Assigns a user to a box */
exports.assignBox = async (req, res, next) => {
    console.info(`>>/admin/cage/${req.params.cageId}/box/${req.params.boxId}/assign`);

    try {
        const user = await User.findOne({username: req.body.username});

        Cage.updateOne(
            {'boxes.box': req.params.boxId},
            {$set: {'boxes.$.assignedUser': user._id}},
        ).exec((err, result) => {
```

```
            console.log(`>>result: ${result}`);
            console.log(`>>err: ${err}`);
            if (err || result.nModified === 0) {
                return next('>>Error assigning box');
            }

            user.cageId = req.params.cageId;
            user.boxId = req.params.boxId;
            user.save()
                .then(result => {
                    if (result) {
                        console.log(`>>Successfully unassigned ${user.username} from cage ${req.params.cageId}`);
                    }
                });
            res.status(200).send(`>>Successfully assigned ${user.username} to cage ${req.params.boxId}`);
        });
    } catch (err) {
        next(err);
    }
};

exports.unassignBox = async (req, res, next) => {
    console.log(`Attempting to unassign ${req.body.username} from ${req.params.boxId}`);
    try {
        const user = await User.findOne({username: req.body.username});

        Cage.updateOne(
            {'boxes.assignedUser': user._id},
            {$set: {'boxes.$.assignedUser': null}},
        ).exec((err, result) => {
            if (err || result.nModified === 0) {
                return next('>>Error unassigning box');
            }

            user.cageId = null;
            user.boxId = null;
            user.save()
                .then(result => {
                    if (result) {
                        console.log(`>>Successfully unassigned ${user.username} from cage ${req.params.cageId}`);
                    }
                });
            return res.status(200).send(`>>Successfully unassigned ${user.username} from box ${req.params.boxId}`);
        });
    } catch (err) {
        next(err);
    }
};

/* Delete box from a cage */
/* Validates and deletes box from specified cage */
exports.deleteBox = async (req, res, next) => {
    try {
        Cage
            .updateOne(
                {'boxes.box': req.params.boxId},
                {$pull: {boxes: {box: req.params.boxId}}},
            )
            .exec((err, result) => {
                console.log(`>>result: ${result}`);
                console.log(`>>err: ${err}`);
                if (err) {
                    return next('>>Error deleting box');
                }

                res.status(200).send(`>>Successfully deleted ${req.params.boxId} from cage ${req.params.cageId}`);
            });
    } catch (err) {
        console.log(err);
        next(err);
    }
};
```