```
const express = require('express');
const router = express.Router();
const Cost = require('../models/costs');
const User = require('../models/users');
const Report = require('../models/reports');
/**
 * @route POST /api/add
 * @desc Adds a new cost entry and deletes the existing report for that month/year if it exists
 * @access Public
 */
router.post('/add', async (req, res) => {
    try {
        const { userid, category, description, sum, date } = req.body;

        if (!userid || !description || !sum) {
            return res.status(400).json({ error: 'Missing required fields: userid, description, sum' });
        }

        if (!['food', 'health', 'housing', 'sport', 'education'].includes(category)) {
            return res.status(400).json({ error: 'Category not supported. Please choose from: food, health, hou
        }

        const costDate = date ? new Date(date) : new Date();
        const cost = new Cost({ userid, category, description, sum, date: costDate });
        const savedCost = await cost.save();

        // Delete existing report for this user, month, and year (invalidate cache)
        const year = costDate.getFullYear();
        const month = costDate.getMonth() + 1; // Get correct month

        await Report.findOneAndDelete({ userid, year, month });

        // Update total cost for the user
        await User.findOneAndUpdate(
            { id: userid },
            { $inc: { total_cost: sum } },
            { new: true }
        );

        res.status(201).json(savedCost);
    } catch (error) {
        res.status(500).json({ error: error.message });
    }
});


/**
 * @route GET /api/report
 * @desc Fetches monthly report for a specific user, returns from cache if available
 * @access Public
 */
router.get('/report', async (req, res) => {
    const { id: userid, year, month } = req.query;

    if (!userid || !year || !month) {
```

```
            return res.status(400).json({ error: 'Missing required query parameters: id, year, month' });
        }

        try {
            // Check if a report already exists in the reports collection
            const existingReport = await Report.findOne({ userid, year, month });

            if (existingReport) {
                return res.json(existingReport); // Return cached report if found
            }

            // If no existing report, generate a new one
            const startOfMonth = new Date(year, month - 1, 1);
            const endOfMonth = new Date(year, month, 0);

            const costs = await Cost.find({
                userid: userid,
                date: { $gte: startOfMonth, $lte: endOfMonth }
            });

            if (costs.length === 0) {
                return res.status(404).json({ error: 'No costs found for this user in the specified month and year'
            }

            // Categorize costs
            const categories = {
                food: [],
                education: [],
                health: [],
                housing: [],
                sport: []
            };

            costs.forEach(cost => {
                const { category, sum, description, date } = cost;
                const day = new Date(date).getDate();
                if (categories[category]) {
                    categories[category].push({ sum, description, day });
                }
            });

            const reportData = { userid, year, month, costs: categories };

            // Save report to the database
            const newReport = new Report(reportData);
            await newReport.save();

            res.json(reportData);
        } catch (error) {
            res.status(500).json({ error: error.message });
        }
    });

    /**
     * @route GET /api/users/:id
     * @desc Retrieves user details and their total spending
     * @access Public
     */
    router.get('/users/:id', async (req, res) => {
```

```
    const userId = req.params.id;

    try {
        const user = await User.findOne({ id: userId });
        if (!user) {
            return res.status(404).json({ error: 'User not found' });
        }

        // Calculate total costs


        res.json({
            first_name: user.first_name,
            last_name: user.last_name,
            id: user.id,
            marital_status: user.marital_status,
            total: user.total_cost,
        });
    } catch (error) {
        res.status(500).json({ error: error.message });
    }
});

/**
 * @route GET /api/about
 * @desc Provides information about the development team
 * @access Public
 */
router.get('/about', async (req, res) => {
    const team = [
        { first_name: 'Omri', last_name: 'Shmuel'},
        { first_name: 'Omer', last_name: 'Tzaadi' },
        { first_name: 'Astewuol', last_name: 'Tsagaow' },
    ];

    res.json(team);
});

module.exports = router;
```
