Team manager:

Eden Laor

Team members:

- Full Name: Eden Laor
- Id: 208939629
- Email: edenlaor1@hotmail.com

- Full Name: Yarin Yahav
- Id: 207952516
- Email: yarinyahav0@gmail.com

Video Link: https://youtu.be/VZQe7_CPxFk

```
// Developers:  Eden Laor - 208939629, Yarin Yahav - 207952516
// app.js


const express = require('express');
const mongoose = require('mongoose');
const app = express();
const port = process.env.PORT || 3000;

const addCalories = require('./addcalories');
const about = require("./about");
const report = require("./report");

// Database Connection
const connectDB = require('./DB');
connectDB();

// Middleware to parse JSON bodies
app.use(express.json());

app.use('/addcalories', addCalories);
app.use("/about", about);
app.use("/report", report);

// Default route
app.get('/', (req, res) =>
{
    const htmlContent = `
      <h3>Welcome to our final Final Project in NodeJS</h3>
      <p>to use the API, use the following in the
<strong>URL</strong>:</p>
      <ul>
        <li>/addcalories</li>
<li>/report?<strong>user_id</strong>=XXX&<strong>year</strong>=XXXX&<st
rong>month</strong>=XX</li>
        <li>/about</li>
      </ul>
    `;

    res.send(htmlContent);
});

app.listen(port, () => console.log(`app listening on port ${port}!`));
```

```
// Developers:  Eden Laor - 208939629, Yarin Yahav - 207952516
// addcalories.js

const express = require('express');
const router = express.Router();
const mongoose = require('mongoose');

// Import the CalorieItem model
const calorieItem = require('./models/CalorieItem');

// Middleware to parse JSON bodies
router.use(express.json());

// POST request to add a new calorie consumption item
router.post('/', async (req, res) =>
{
    const { user_id, year, month, day, description, category, amount }
= req.body;

    // Check if all required parameters are provided
    if (!user_id || !description || !category || !amount) {
        return res.status(400).json({ error: "User ID, description,
category, and amount are required" });
    }

    // Check if date was provided or not
    if ((year && !month) || (year && !day) || (month && !year) ||
(month && !day) || (day && !year) || (day && !month)) {
        return res.status(400).json({ error: "Can't provide 'half'
date. Please either provide full date (day, month and year) or none to
use today's date." });
    }

    // Set default values for year, month, and day if not provided
    const currentDate = new Date();
    const currentYear = year || currentDate.getFullYear();
    const currentMonth = month || currentDate.getMonth() + 1;
    const currentDay = day || currentDate.getDate();

    try
    {
        // Check if the user exists in the database
        const user = await findUserById(user_id);
```

```
        if (!user)
        {
            return res.status(404).json({ error: "User not found" });
        }

        // Create a new calorieItem object (based on the model)
        const newCalorieItem = new calorieItem({
            user_id: parseInt(user_id),
            year: parseInt(currentYear),
            month: parseInt(currentMonth),
            day: parseInt(currentDay),
            description,
            category,
            amount: parseInt(amount)
        });

        // Save the new calorie consumption item to the database
        await newCalorieItem.save();

        res.json({ message: "Calorie consumption item added
successfully" });
    } catch (error)
    {
        console.error(error);
        res.status(500).json({ error: "Internal server error" });
    }
});

// Function to find user by ID
function findUserById(user_id)
{
    return new Promise((resolve, reject) =>
    {
        mongoose.connection.collection('users').findOne({ id:
parseInt(user_id) }, (err, user) =>
        {
            if (err)
            {
                reject(err);
            } else
            {
                resolve(user);
```

```
            }
        });
    });
}

module.exports = router;
```

```
// Developers:  Eden Laor - 208939629, Yarin Yahav - 207952516
// about.js


const express = require('express');
const router = express.Router();

// Array of developer objects
const developers = [
  {
    firstname: "Eden",
    lastname: "Laor",
    id: 208939629,
    email: "edenlaor1@hotmail.com"
  },
  {
    firstname: "Yarin",
    lastname: "Yahav",
    id: 207952516,
    email: "yarinyahav0@gmail.com"
  }
];

// GET request to get developers' information
router.get('/', (req, res) =>
{
  res.json(developers);
});

module.exports = router;
```

```
// Developers:  Eden Laor - 208939629, Yarin Yahav - 207952516
// report.js


const express = require('express');
const router = express.Router();
const CalorieItem = require('./models/CalorieItem');

// GET request to get detailed report
router.get('/', async (req, res) =>
{
  const { user_id, month, year } = req.query;

  // Check if all required parameters are provided
  if (!user_id || !month || !year)
  {
    return res.status(400).json({ error: "User ID, month, and year are
required" });
  }

  try
  {
    // Find all calorie items for the specified user, month, and year
    const calorieItems = await CalorieItem.find({
      user_id: parseInt(user_id),
      month: parseInt(month),
      year: parseInt(year)
    });

    // Create the report object
    const report = {
      breakfast: [],
      lunch: [],
      dinner: [],
      other: []
    };

    // Populate the report with the fetched calorie items
    calorieItems.forEach(item =>
    {
      switch (item.category)
      {
        case "breakfast":
```

```
            report.breakfast.push({
              day: item.day,
              description: item.description,
              amount: item.amount
            });
            break;
          case "lunch":
            report.lunch.push({
              day: item.day,
              description: item.description,
              amount: item.amount
            });
            break;
          case "dinner":
            report.dinner.push({
              day: item.day,
              description: item.description,
              amount: item.amount
            });
            break;
          case "other":
            report.other.push({
              day: item.day,
              description: item.description,
              amount: item.amount
            });
            break;
          default:
            break;
        }
      });

      res.json(report);
    } catch (error)
    {
      console.error(error);
      res.status(500).json({ error: "Internal server error" });
    }
});

module.exports = router;
```

```
// Developers:  Eden Laor - 208939629, Yarin Yahav - 207952516
// DB.js

const mongoose = require('mongoose');

// MongoDB connection URI
const uri =
'mongodb+srv://Eden:zCxId3WMmcExfBpV@cluster0.21yxdtj.mongodb.net/Clust
er?retryWrites=true&w=majority';

const connectDB = async () =>
{
  try
  {
    await mongoose.connect(uri);
    console.log('MongoDB connected successfully');
  } catch (error) // any error that occured while attempting to connect
to the database
  {
    console.error('MongoDB connection error:', error);
  }
};

module.exports = connectDB;
```

```
// Developers:  Eden Laor - 208939629, Yarin Yahav - 207952516
// models/CalorieItem.js

const mongoose = require('mongoose');


// the schema for an item to add using addCalories
const calorieItemSchema = new mongoose.Schema({
    user_id: {
        type: Number,
        required: true
    },
    year: {
        type: Number,
        required: true
    },
    month: {
        type: Number,
        required: true
    },
    day: {
        type: Number,
        required: true
    },
    description: {
        type: String,
        required: true
    },
    category: {
        type: String,
        enum: ['breakfast', 'lunch', 'dinner', 'other'],
        required: true
    },
    amount: {
        type: Number,
        required: true
    }
});

const CalorieItem = mongoose.model('CalorieItem', calorieItemSchema,
'calories');

module.exports = CalorieItem;
```

```
// Developers:  Eden Laor - 208939629, Yarin Yahav - 207952516
// models/User.js

const mongoose = require('mongoose');


//Schema for creating a user in the database
const userSchema = new mongoose.Schema({
    id: { type: Number, required: true, unique: true },
    first_name: { type: String, required: true },
    last_name: { type: String, required: true },
    birthday: { type: Date, required: true }
});

const User = mongoose.model('User', userSchema);

module.exports = User;
```