```
// backend/routes/weather.js
const express = require('express');
const axios = require('axios');
const router = express.Router();

const API_KEY =
process.env.OPENWEATHER_API_KEY;
const BASE_URL =
process.env.OPENWEATHER_BASE_URL;

// GET /api/weather/current?
city=London&units=metric
// Fetches current weather for a city
router.get('/current', async (req, res) => {
  const { city, lat, lon, units = 'metric' } =
req.query;  // units: metric (Celsius) or imperial
(Fahrenheit)

  if (!city && !(lat && lon)) {
    return res.status(400).json({ error: 'City name or
lat/lon coordinates required' });
```

```
  }

  let url = `${BASE_URL}/weather?appid=${API_KEY}&units=${units}`;
  if (city) {
    url += `&q=${encodeURIComponent(city)}`;
  } else {
    url += `&lat=${lat}&lon=${lon}`;
  }

  try {
    const response = await axios.get(url);
    const data = response.data;

    // Format response for frontend
    const formatted = {
      location: data.name || `${lat}, ${lon}`,
      country: data.sys.country,
      weather: {
        main: data.weather[0].main,
        description: data.weather[0].description,
```

```
      icon:
`https://openweathermap.org/img/wn/${data
.weather[0].icon}@2x.png`,
    },
    main: {
      temp: Math.round(data.main.temp),
      feelsLike: Math.round(data.main.feels_like),
      tempMin: Math.round(data.main.temp_min),
      tempMax: Math.round(data.main.temp_max),
      pressure: data.main.pressure,
      humidity: data.main.humidity,
    },
    wind: {
      speed: data.wind.speed,
      direction: data.wind.deg,
    },
    visibility: data.visibility,
    timestamp: data.dt,
  };

  res.json(formatted);
```

```
  } catch (error) {
    if (error.response?.status === 404) {
      res.status(404).json({ error: 'Location not
found' });
    } else if (error.response?.status === 401) {
      res.status(401).json({ error: 'Invalid API key' });
    } else {
      console.error('Weather API error:', error.message);
      res.status(500).json({ error: 'Failed to fetch
current weather' });
    }
  }
});

// GET /api/weather/forecast?
city=London&units=metric
// Fetches 5-day forecast (3-hour intervals)
router.get('/forecast', async (req, res) => {
  const { city, lat, lon, units = 'metric' } = req.query;

  if (!city && !(lat && lon)) {
```

```
    return res.status(400).json({ error: 'City name or
lat/lon coordinates required' });
  }

  let url = `${BASE_URL}/forecast?appid=$
{API_KEY}&units=${units}`;
  if (city) {
    url += `&q=${encodeURIComponent(city)}`;
  } else {
    url += `&lat=${lat}&lon=${lon}`;
  }

  try {
    const response = await axios.get(url);
    const data = response.data;

    // Format forecast: Group by day for simplicity (5
days)
    const dailyForecast = [];
    const fiveDayData = data.list.slice(0, 40); // ~5
days (8 entries/day * 5 = 40)
```

```
    fiveDayData.forEach((item) => {
      const date = new Date(item.dt *
1000).toLocaleDateString('en-US', { weekday:
'short', month: 'short', day: 'numeric'
```

October 16, 2025 12:25:48

00:00:04

```
});
      const existingDay = dailyForecast.find(day =>
day.date === date);

      if (!existingDay) {
        dailyForecast.push({
          date,
          temp: Math.round(item.main.temp),
          feelsLike: Math.round(item.main.feels_like),
          description: item.weather[0].description,
          icon:
`https://openweathermap.org/img/wn/${item
.weather[0].icon}@2x.png`,
          humidity: item.main.humidity,
```

```
      });
    }
  });


  res.json({
    city: data.city.name,
    country: data.city.country,
    list: dailyForecast,
  });
 } catch (error) {
  if (error.response?.status === 404) {
    res.status(404).json({ error: 'Location not
found' });
  } else if (error.response?.status === 401) {
    res.status(401).json({ error: 'Invalid API key' });
  } else {
    console.error('Forecast API error:', error.message);
    res.status(500).json({ error: 'Failed to fetch
forecast' });
  }
 }
```

```
    });
  } catch (error) {
    if (error.response?.status === 404) {
      res.status(404).json({ error: 'Location not
found' });
    } else if (error.response?.status === 401) {
      res.status(401).json({ error: 'Invalid API key' });
    } else {
      console.error('Forecast API error:', error.message);
      res.status(500).json({ error: 'Failed to fetch
forecast' });
    }
  }
});

module.exports = router;
```