

Node.js: gestire le zone DNS di Cloudflare

In questo tutorial vedremo come gestire una zona DNS su Cloudflare con Node.js.

Le API di Cloudflare utilizzano come meccanismo di autenticazione un header di tipo Bearer con un token che dobbiamo generare nella nostra area utente.

Una volta creato il token dobbiamo aggiungerlo ad ogni richiesta. Le API di Cloudflare sono di tipo REST, quindi utilizzano il formato JSON sia per le richieste che per le risposte.

Creiamo quindi una funzione di utility per effettuare le richieste alle API.

```
'use strict';

const https = require('https');
const TOKEN = 'your API token';

const request = (data = null, method = 'GET',
endpoint = '') => {
  const options = {
    hostname: 'api.cloudflare.com',
    port: 443,
    path: '/client/v4/' + endpoint,
    method: method,
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer ' + TOKEN
```

```

        }
    };

    if(method !== 'GET' && data !== null) {
        const json = JSON.stringify(data);
        options.headers['Content-Length'] =
json.length;
    }

    return new Promise((resolve, reject) => {
        const req = https.request(options, res =>
{
            let body = '';
            res.on('data', d => {
                body += d;
            });
            res.on('end', () => {
                resolve(JSON.parse(body));
            });
        });

        req.on('error', err => {
            reject(err);
        });

        if(method !== 'GET' && data !== null) {
            req.write(JSON.stringify(data));
        }

        req.end();
    });
};

```

Definiamo ora una classe che userà la funzione definita in precedenza per gestire le operazioni su una zona DNS di Cloudflare.

```
class Cloudflare {
  static async getDNSZones(req, res) {
    try {
      const response = await request(null,
'GET', 'zones');
      res.json(response);
    } catch(err) {
      res.json(err);
    }
  }
  static async getDNSZoneRecords(req, res) {
    const { id } = req.body;
    try {
      const response = await request(null,
'GET', `zones/${id}/dns_records`);
      res.json(response);
    } catch(err) {
      res.json(err);
    }
  }

  static async addDNSRecord(req, res) {
    const { id, name, content, priority, ttl,
type } = req.body;
    const prior = parseInt(priority, 10);
    const data = { name, content, priority:
prior, ttl, type };
    try {
      const response = await request(data,
'POST', `zones/${id}/dns_records`);
```

```

        res.json(response);
    } catch(err) {
        res.json(err);
    }
}

static async deleteDNSRecord(req, res) {
    const { zone_id, record_id } = req.body;
    try {
        const response = await request(null,
'DELETE',
`zones/${zone_id}/dns_records/${record_id}`);
        res.json(response);
    } catch(err) {
        res.json(err);
    }
}
}

module.exports = Cloudflare;

```

Una zona DNS ha un ID e a sua volta anche un record DNS ha un proprio ID restituiti da Cloudflare insieme al tipo di record, al nome, al contenuto, al TTL e alla priorità (quest'ultima obbligatoria per i record MX).

Non ci resta che usare la logica della nostra classe nell'applicazione che la contiene.

```

'use strict';

const express = require('express');
const bodyParser = require('body-parser');

```

```
const path = require('path');
const port = process.env.PORT || 3000;
const app = express();
const Cloudflare = require('./classes/Cloudflare');

app.disable('x-powered-by');

app.use('/public',
express.static(path.join(__dirname, '/public'), {
  maxAge: 0,
  dotfiles: 'ignore',
  etag: false
}));

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));

app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname) +
'/views/index.html');
});

app.get('/api/dns-zones', Cloudflare.getDNSZones);
app.post('/api/zone-records',
Cloudflare.getDNSZoneRecords);
app.post('/api/add-record', Cloudflare.addDNSRecord);
app.post('/api/delete-record',
Cloudflare.deleteDNSRecord);

if (app.get('env') === 'development') {
  app.use((err, req, res, next) => {
    res.status(err.status || 500);
```

```
    });  
  }  
  
  app.use((err, req, res, next) => {  
    res.status(err.status || 500);  
  });  
  
  app.listen(port);
```

Conclusione

Gestire una zona DNS di Cloudflare con Node.js si rivela essere un compito relativamente semplice data la natura RESTful delle API di Cloudflare.

Codice di esempio

GitHub