@push.rocks/smartsmtp

A module for handling SMTP operations such as sending emails via SMTP or sendmail transport.

- readme.md for @push.rocks/smartsmtp

readme.md for @push.rocks/smartsmtp

a module for handling smtp stuff

Install

To install `@push.rocks/smartsmtp`, use the following command with npm:

```
npm install @push.rocks/smartsmtp --save
```

Ensure that you are installing the package in a project set up with TypeScript and support for ECMAScript modules, as the usage examples provided will rely on this configuration.

Usage

`@push.rocks/smartsmtp` simplifies handling SMTP-based emailing in Node.js applications, offering a streamlined interface to create transporters and send emails using popular services like Nodemailer and enhancements for template-based emails via `@pushrocks/smartmail`. This guide walks you through setting up a `Smartsmtp` instance and sending emails.

Setting Up

First, ensure you import the necessary classes from the module. Here's how you set up your imports using ESM syntax:

```
import { Smartsmtp } from '@push.rocks/smartsmtp';
```

Creating SMTP Transport

`@push.rocks/smartsmtp` provides two primary ways to set up an SMTP transporter: through direct SMTP server credentials or utilizing the local `sendmail` command.

SMTP Server Credentials

To connect to an SMTP server directly, you'll need the server address, username, and password. Here's how you can create a `Smartsmtp` instance using SMTP server credentials:

```
// Define your SMTP configuration
const smtpOptions = {
  smtpServer: 'smtp.example.com',
  smtpUser: 'user@example.com',
  smtpPassword: 'yourPassword'
};

// Async function to create and use a Smartsmtp instance
async function setupSmtp() {
  const smtpInstance = await Smartsmtp.createSmartsmtpWithRelay(smtpOptions);

  // smtpInstance is now ready to use
}
```

Using Sendmail

If you wish to use the local `sendmail` command, which is common in UNIX environments, you can create a `Smartsmtp` instance dedicated to that:

```
async function setupSendmail() {
  const sendmailInstance = await Smartsmtp.createSmartsmtpSendmail();

  // sendmailInstance is now ready to use for sending emails
}
```

Sending Emails

With a `Smartsmtp` instance ready, you can send emails. This requires creating a `Smartmail` instance (from the `@pushrocks/smartmail` package) that defines the email configuration, including subjects, recipients, and body content.

```
import { Smartmail } from '@pushrocks/smartmail';

async function sendEmail(smtpInstance: Smartsmtp) {
  // Create a Smartmail instance
```

```
  const myEmail = new Smartmail({
    from: 'me@example.com',
    subject: 'Test Email',
    body: 'This is a test email sent using @push.rocks/smartsmtp.'
  });

  // Use the smtpInstance to send the email
  const result = await smtpInstance.sendSmartMail(myEmail, 'recipient@example.com');

  console.log(result); // Check the result
}
```

In the example above, `Smartmail` is utilized to define the base content of the email being sent. The `sendSmartMail` method of `Smartsmtp` takes this email configuration, alongside recipient details, and performs the sending operation.

This completes the basic usage guide for `@push.rocks/smartsmtp`. With these steps, you can integrate straightforward SMTP email sending capabilities into your Node.js applications, leveraging modern TypeScript syntax and ESM modules. For further customization and advanced features, refer to the documentation of Nodemailer and `@pushrocks/smartmail`.

License and Legal Information

This repository contains open-source code that is licensed under the MIT License. A copy of the MIT License can be found in the license file within this repository.

Please note: The MIT License does not grant permission to use the trade names, trademarks, service marks, or product names of the project, except as required for reasonable and customary use in describing the origin of the work and reproducing the content of the NOTICE file.

Trademarks

This project is owned and maintained by Task Venture Capital GmbH. The names and logos associated with Task Venture Capital GmbH and any related products or services are trademarks of Task Venture Capital GmbH and are not included within the scope of the MIT license granted herein. Use of these trademarks must comply with Task Venture Capital GmbH's Trademark Guidelines, and any usage must be approved in writing by Task Venture Capital GmbH.

Company Information

Task Venture Capital GmbH
Registered at District court Bremen HRB 35230 HB, Germany

For any legal inquiries or if you require further information, please contact us via email at hello@task.vc.

By using this repository, you acknowledge that you have read this section, agree to comply with its terms, and understand that the licensing of the code does not imply endorsement by Task Venture Capital GmbH of any derivative works.