ARTRAGIS

Zeste de savoir

MailDev : la solution pour tester les mails générés par votre application web

samedi 04 mai 2024

Table des matières

Introduction

Je sais pas pour vous, mais j'ai un soucis fréquent lorsque je démarre sur un projet où des mails sont générés automatiquement : comment je fais pour m'assurer que mes modifs n'ont pas cassé le mail ?

La solution la plus fréquemment utilisée consiste à envoyer le mail à un serveur mail public vers mon adresse professionnelle.

Sauf que… je suis maladroit et parfois, je fais une typo dans cette adresse. Et dans mon entreprise actuelle, le "serveur public" c'est Amazon SES. Et SES demande à ce qu'on ne génère pas plus de 10% de bounce. Alors prendre le risque de faire une typo en développement et de casser la prod de l'entreprise, c'est pas cool.

Heureusement, un de mes collègues a trouvé une solution et je me fais aujourd'hui son messager : MailDev.

1. Trêve de chichi, démarrons tout ça

1.1. Présentation

MailDev est un projet open source qui est disponible sur github .

Pour ceux qui aime les spec techniques, il est fait en JavaScript et se base sur les libs `express` et `smtp-server`.

> *i* Kamoulox ! Que vient faire un serveur web dans un projet de serveur mail ?

Bah c'est ça qui est cool avec MailDev, c'est qu'une fois que votre application a envoyé un mail, vous pouvez le voir directement sur une page web servie par MailDev.

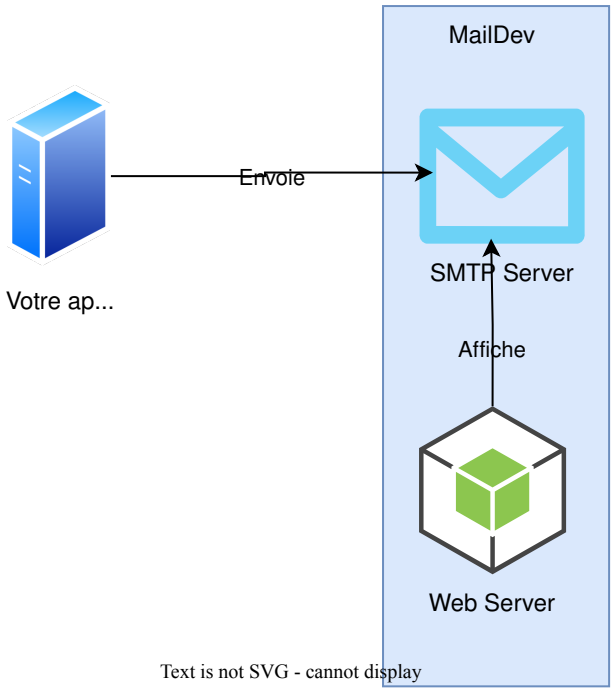

FIGURE 1.1. – L'architecture de MailDev

Pour voir ce que ça donne, la documentation nous donne un joli screenshot que je ne pense pas pouvoir améliorer :

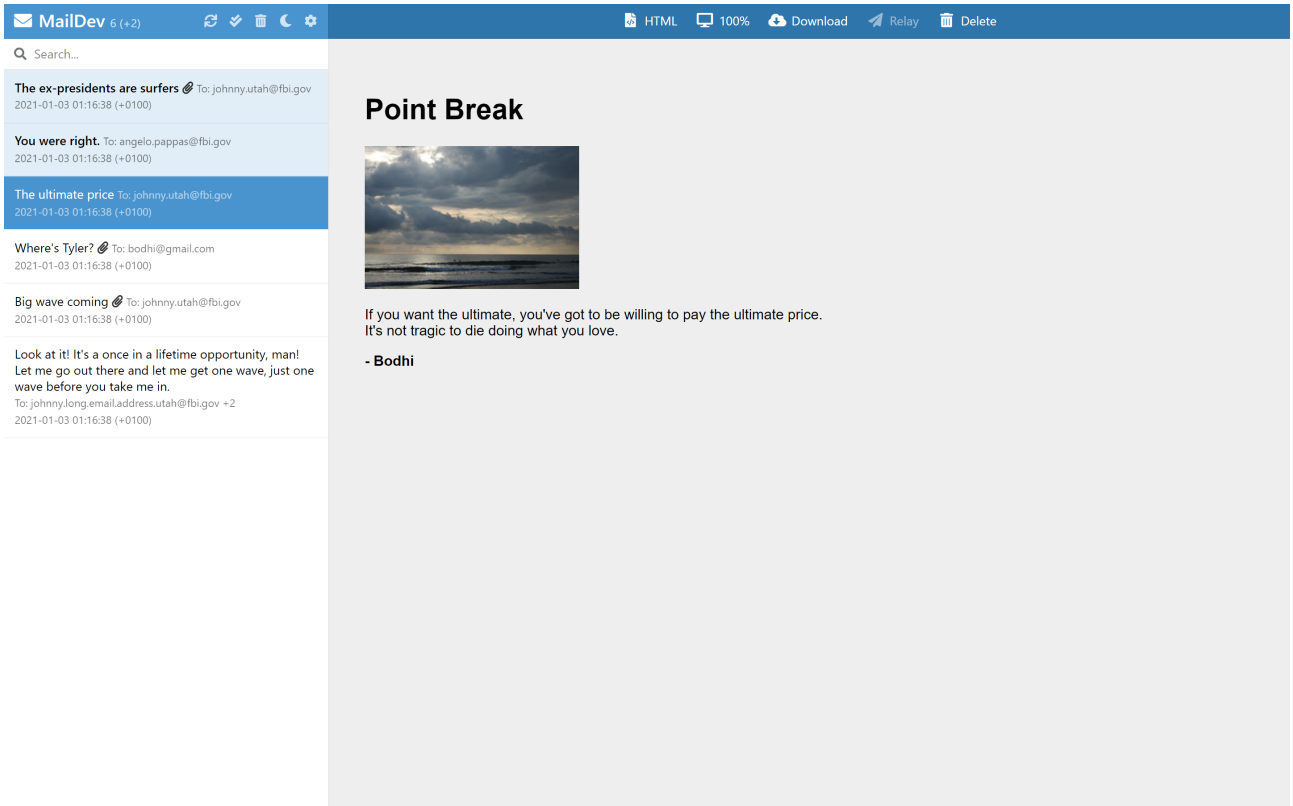

FIGURE 1.2. – Screenshot de l'application Web

1.2. Et en avant Guingamp !

Que ça soit dans la documentation ou dans mon entreprise, c'est au travers de l'image docker que l'ont prend possession de ce super projet, notamment, avec docker compose :

```yaml
maildev:
  image: maildev/maildev
  ports:
    - "1080:1080"
    - "1025:1025"
```

Cela aura pour effet de capturer tous les mails envoyés sur le port 1025 et d'ouvrir l'application web sur le port 1080.

Dans mon entreprise, on le met derrière un traefik qui expose le host `smtp.localhost` et on y accède sur le port 80 pour la facilité grâce à : `traefik.http.services.maildev.loadbalancer.server.port=1080`.

2. Retour d'expérience

Cet outil est vraiment cool, il est en plus très léger à l'exécution ce qui permet d'éviter de surcharger les stack docker.

Par contre, j'ai un petit point, je n'ai jamais réussi à avoir un conteneur "healthy" sur mon PC pro sur windows et docker desktop. Je ne sais pas si c'est lié.

Autre élément, comme nous avons un autre système de capture de mail léger pour nos tests d'intégration, nous n'avons pas réellement profité de la capacité offerte par maildev de fournir une représentation json des mails reçus.

Enfin, petit élément : maildev va tout accepter, donc difficile de tester les cas de rejet d'email. Enfin, il est difficile de tester la manière dont gmail ou office365 rendront votre email.

> **!** Si votre mail est en html et stylé avec du CSS, n'utilisez pas des classes car… les webmails de ce type ajoutent des préfixes aléatoires à ces classes, les rendant inopérantes. Pour faire du CSS, il faut directement utiliser les style inline dans un mail.
> D'ailleurs, petit pro-tip : ne mettez pas de texte invisible (texte blanc sur fond blanc par exemple), les filtres antispam amalgament ces pratiques à des pratiques de phishing.

Conclusion

Et voilà, vous avez un système bien sympa pour tester en local la génération des mails de votre application.

Icône générée par Dall-EE au travers de Copilot.