

Soutenance du TB

BeePlace - Recréer l'expérience collaborative du r/place de Reddit

Sommaire

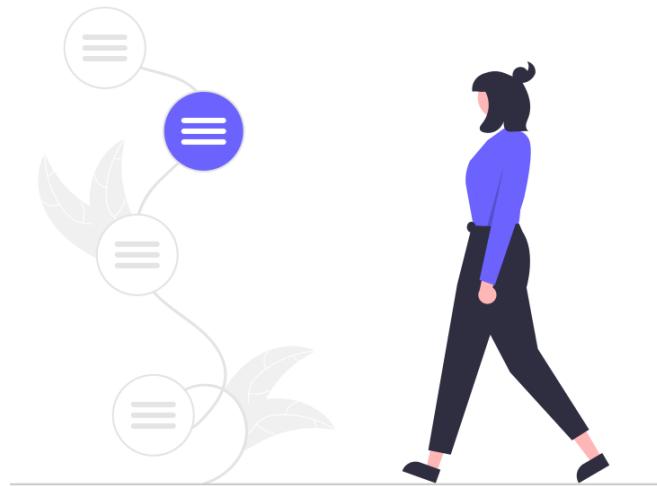
Contexte

Problématique

Solution et contraintes

Conception et réalisation

Conclusion



Contexte

Baleinev Festival

Association Baleinev

- À la HEIG-VD
- Festival de musique depuis près de 30 ans

Pimp My Wall

- Nouveau concept depuis 2014 : Pimp My Wall
- Application de dessin collaboratif

Depuis 2018

- BeeScreens : nouvelle version open source
- Collection d'applications interactives
- Cadre pouvant sortir du festival



Problématique

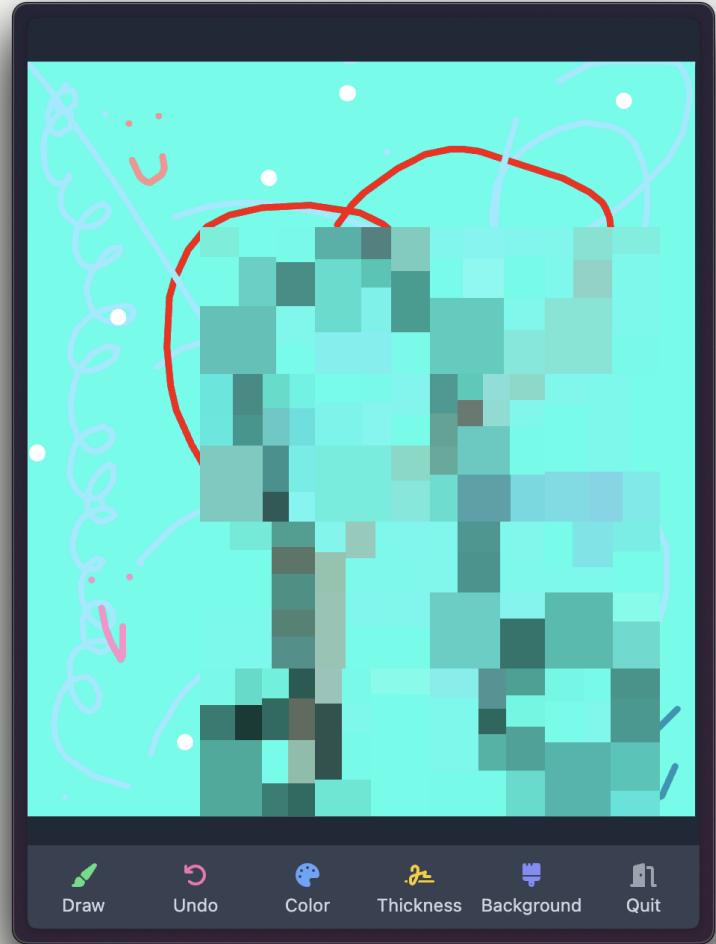
Liberté de Pimp My Wall

Débordements en fin de soirée (oeuvres inappropriées)

Modération difficile et chronophage

Affecte les autres utilisateurs : gâche les dessins

↳ **Expérience frustrante pour nous et les festivaliers**



Solution et contraintes

Solution

r/place de Reddit

Toile partagée par des millions d'utilisateurs

1 pixel par personne toutes les 5 minutes

Encourage la collaboration

Occasionnel (2017, 2022 et 2023) ⇒ forte rivalité

↳ S'inspirer de r/place pour créer sa variante open source : BeePlace



Contexte physique

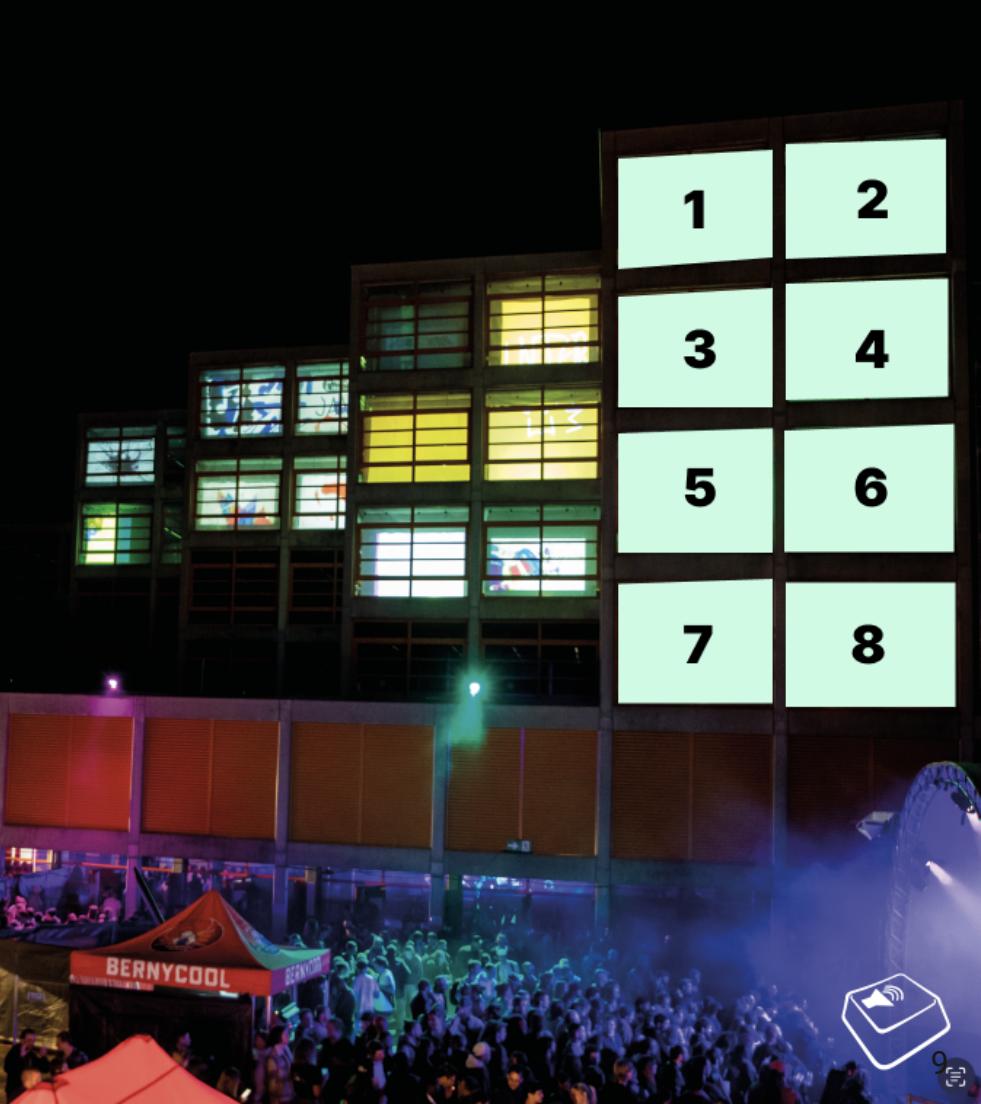
Les festivaliers

- Festivaliers utilisent leur smartphone
 - ↳ Optimisation pour ce médium
- Utilisent leur propre connexion internet

La HEIG-VD

- Diffusion sur les murs de l'école
- Mode affichage nécessaire
- Séparer l'affichage entre plusieurs écrans

↳ Web app optimisée pour le desktop et le mobile



Garantir le fonctionnement de l'application

Assurer la scalabilité

Garder une latence faible (bonne UX)

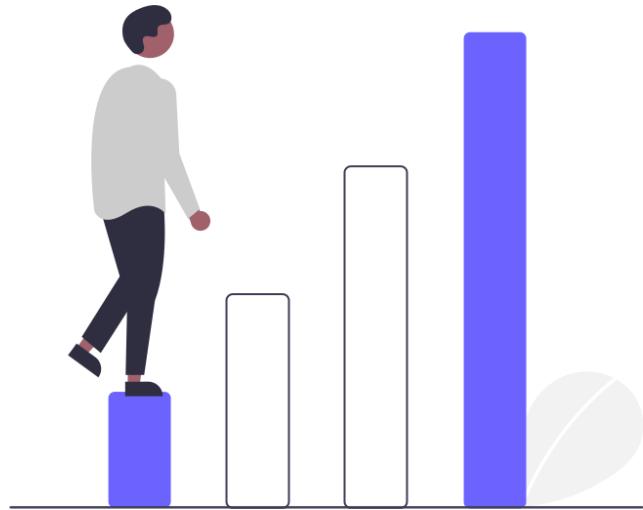
Pouvoir gérer un nombre élevé d'utilisateurs simultanés

Pics de fréquentation lors du festival

Ordre de grandeur

Reddit 10.5 millions de participants

Baleinev Festival 1500 festivaliers



Conception et réalisation

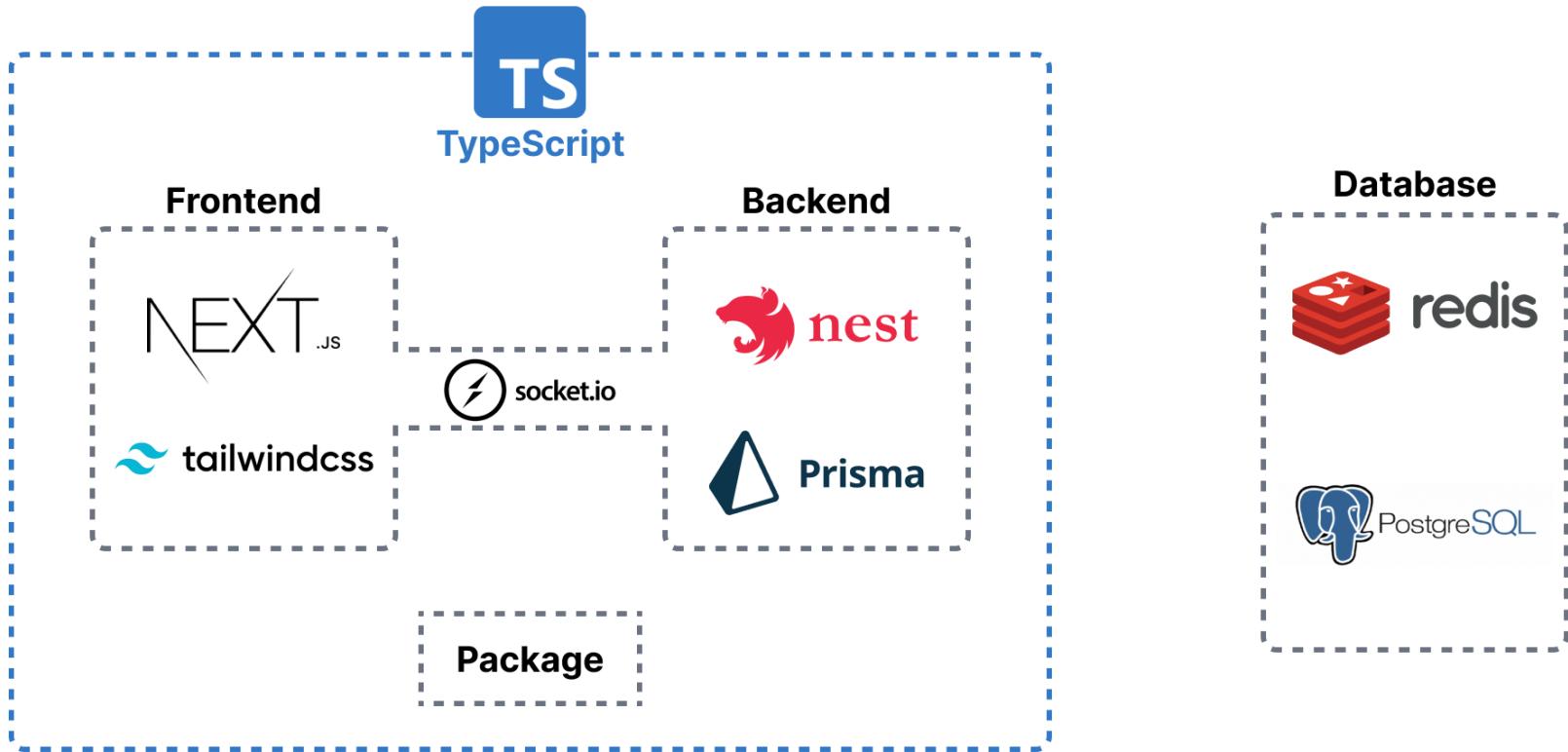
Besoins de l'application

Fonctionnalités

- Temps réel
- Stocker les pixels
- Interface pour naviguer dans le canvas
- Identifier les utilisateurs



Technologies



Identification

Pour limiter la fréquence d'ajout de pixels

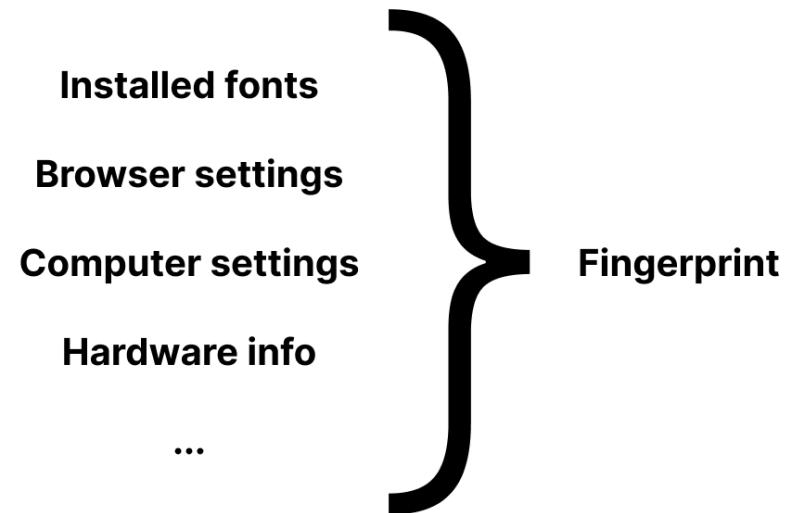
- Plus simple possible pour les festivaliers
- Sans nuire à la fluidité de l'expérience

Solution

- Authentification par empreinte digitale (fingerprint)
- Librairie FingerprintJS (open source)

Problèmes

- Risque de collisions
- Possibilité de bots (générée côté client)



Canvas

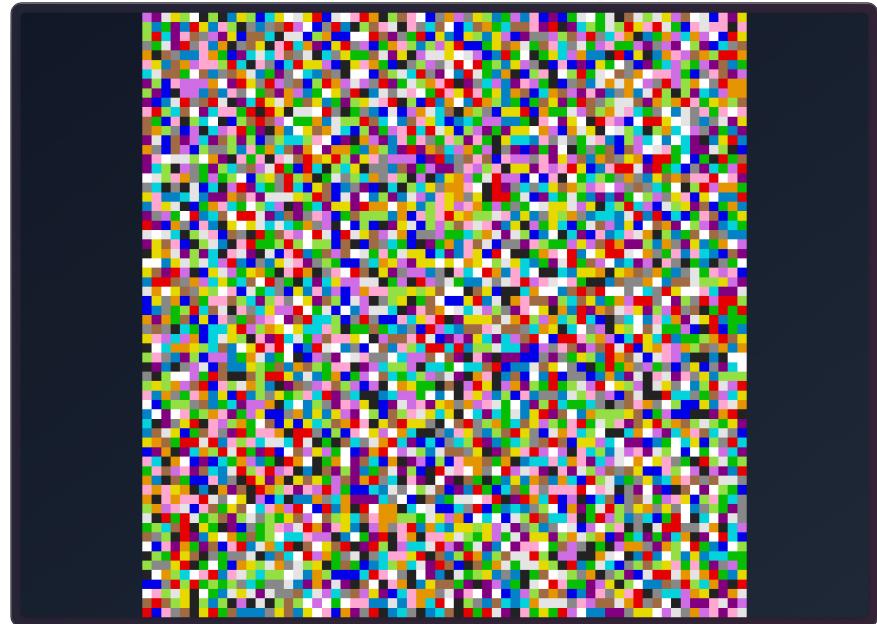
Canvas HTML5

Permet d'afficher une image matricielle

Représentation des pixels de la toile

Possible de dessiner les pixels individuellement ou plusieurs d'un bloc

Mais comment stocker les pixels ? 🤔



Stockage

Quoi stocker ?

La position et la couleur de chaque pixel

Bitfield Redis

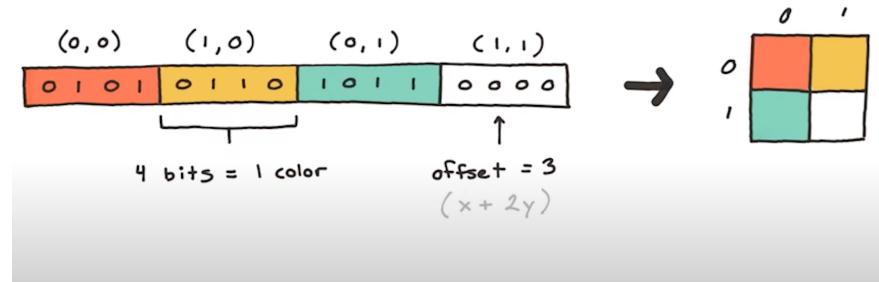
Permet de ne stocker que la couleur

La position est implicite

3 niveaux de stockage

1. Dans la mémoire de l'application
2. Bitfield Redis
3. Base de données PostgreSQL

Structure du Bitfield Redis



Crédits: Daniel Ellis (Reddit)

Structure SQL

pixels	
id	text
x	int4
y	int4
color	text
created_at	timestamp
user_id	text

Backend

NestJS

- Architecture DDD
- ORM Prisma (PostgreSQL)
- Configuration par variables d'environnement
- Communication avec les clients via Socket.IO

Administration

- Endpoints HTTP protégés (stratégie d'API Key)
- Actions :
 - Lecture seule
 - Remise à zéro de la toile
 - Recouvrir une zone de pixels

Backend

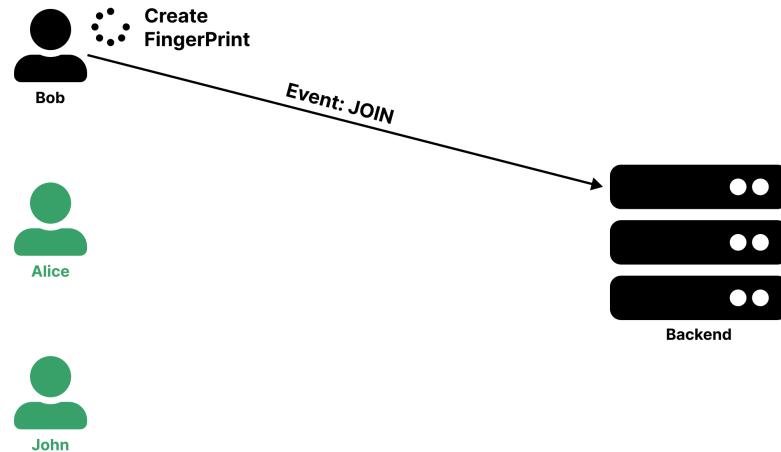
NestJS

- Architecture DDD
- ORM Prisma (PostgreSQL)
- Configuration par variables d'environnement
- Communication avec les clients via Socket.IO

Administration

- Endpoints HTTP protégés (stratégie d'API Key)
- Actions :
 - Lecture seule
 - Remise à zéro de la toile
 - Recouvrir une zone de pixels

Événements Socket.IO



Backend

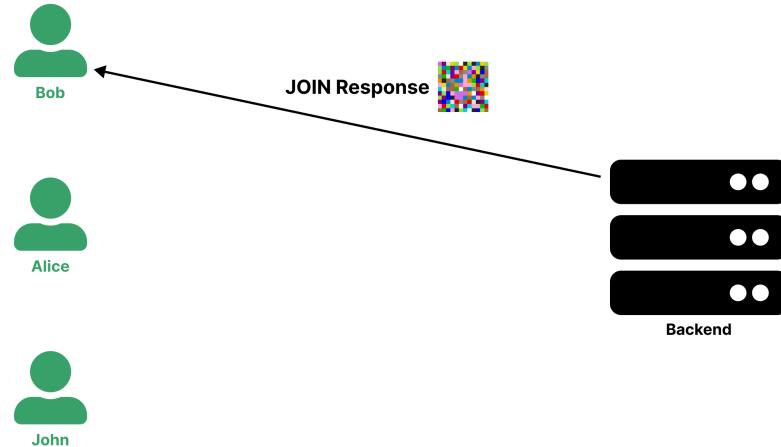
NestJS

- Architecture DDD
- ORM Prisma (PostgreSQL)
- Configuration par variables d'environnement
- Communication avec les clients via Socket.IO

Administration

- Endpoints HTTP protégés (stratégie d'API Key)
- Actions :
 - Lecture seule
 - Remise à zéro de la toile
 - Recouvrir une zone de pixels

Événements Socket.IO



Backend

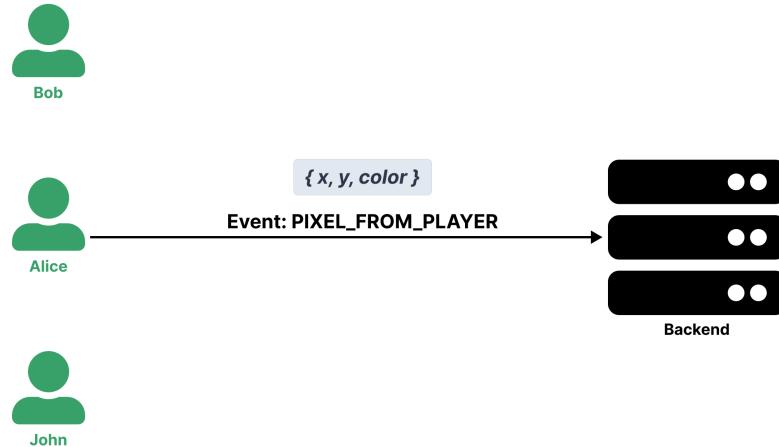
NestJS

- Architecture DDD
- ORM Prisma (PostgreSQL)
- Configuration par variables d'environnement
- Communication avec les clients via Socket.IO

Administration

- Endpoints HTTP protégés (stratégie d'API Key)
- Actions :
 - Lecture seule
 - Remise à zéro de la toile
 - Recouvrir une zone de pixels

Événements Socket.IO



Backend

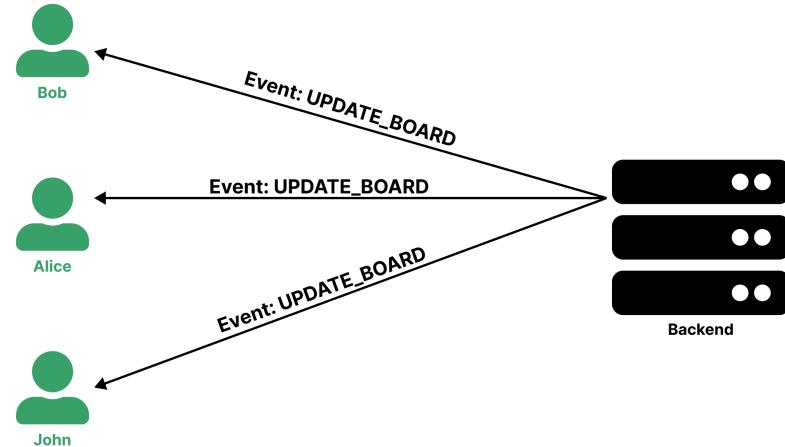
NestJS

- Architecture DDD
- ORM Prisma (PostgreSQL)
- Configuration par variables d'environnement
- Communication avec les clients via Socket.IO

Administration

- Endpoints HTTP protégés (stratégie d'API Key)
- Actions :
 - Lecture seule
 - Remise à zéro de la toile
 - Recouvrir une zone de pixels

Événements Socket.IO



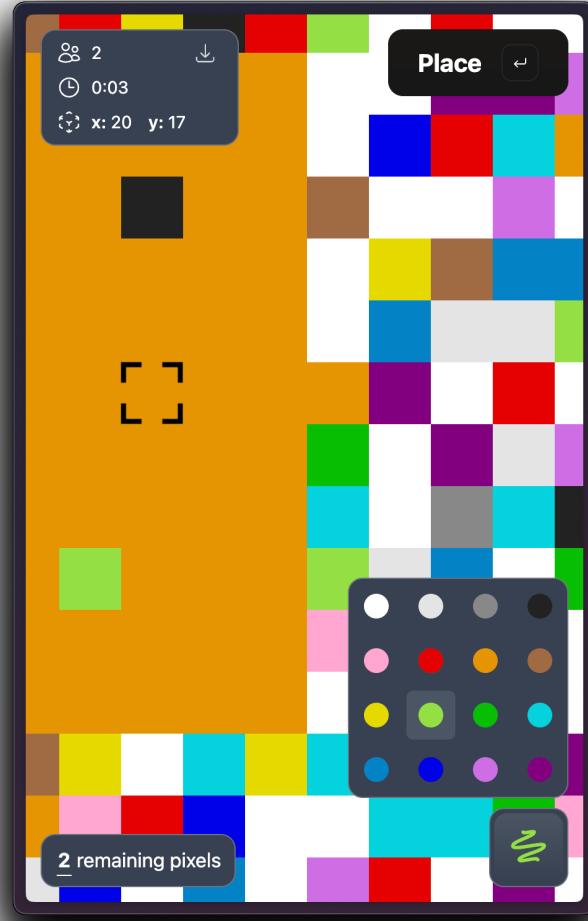
Frontend

Next.js

- Communication : se greffe aux WebSockets du Backend avec Socket.IO
- Stockage de l'état dans un state global
- PinchZoom du canvas
- Mode affichage

Design

- Utilisation de Tailwind CSS



Montée en charge

Outils utilisés



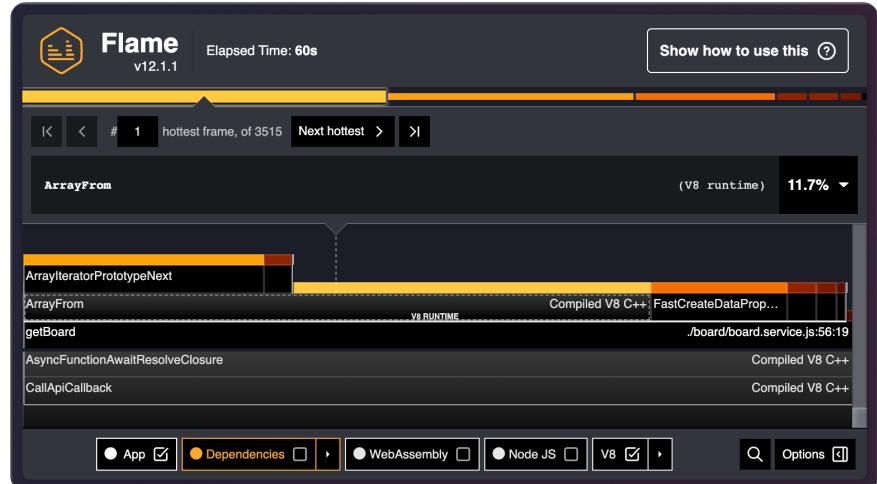
- k6 pour les tests de montée en charge
- Métriques (avant que la latence soit $> 1.2\text{s}$):
 - Nombre d'utilisateurs virtuels
 - Nombre de pixels dessinés
- Clinic.js pour le profiling (flame graph)

```
execution: local
script: dist/tests/breakpoint-test.js
output: - 

scenarios: (100.00%) 1 scenario, 10000 max VUs, 5m0s max duration (incl. graceful stop):
  * Breakpoint_test: Up to 1000.00 iterations/s for 5m0s over 1 stages (maxVUs: 10000)

✓ error

data_received.....: 187 MB 6.4 MB/s
data_sent.....: 1.6 MB 57 kB/s
drawnPixels.....: 2822 96.908675/s
Errors.....: 0 0/s
http_req_blocked.....: avg=280.63ms min=59.1ms med=194.65ms max=1.56s p(90)=616.96ms p(95)=780.41ms
http_req_connecting.....: avg=71.71ms min=17.66ms med=41.75ms max=1.03s p(90)=172.67ms p(95)=237.06ms
http_req_duration.....: avg=203.73ms min=29.29ms med=131.84ms max=1.55s p(90)=483.22ms p(95)=605.44ms
  { expected_response:true }
  avg=203.73ms min=29.29ms med=131.84ms max=1.55s p(90)=483.22ms p(95)=605.44ms
http_req_failed.....: 0.00% ✓ 0 x 801
http_req_receiving.....: avg=25.94ms min=11μs med=1.22ms max=597.52ms p(90)=66.39ms p(95)=155.76ms
http_req_sending.....: avg=236.16μs min=26μs med=56μs max=68.16ms p(90)=86μs p(95)=127μs
http_req_tls_handshaking.....: avg=208.54ms min=37.78ms med=130.32ms max=1.37s p(90)=451.38ms p(95)=598.55ms
http_req_waiting.....: avg=177.55ms min=29.02ms med=115.24ms max=1.39s p(90)=399.48ms p(95)=517.9ms
http_reqs.....: 801 27.506679/s
vus.....: 1231 min=0 max=1231
vus_max.....: 10000 min=6044 max=10000
ws_connecting.....: avg=587.59ms min=80.83ms med=438.99ms max=2.78s p(90)=1.39s p(95)=1.66s
ws_msgs_received.....: 4895 168.096373/s
ws_msgs_sent.....: 5090 174.792756/s
```



Méthodologie

Méthodologie

1.

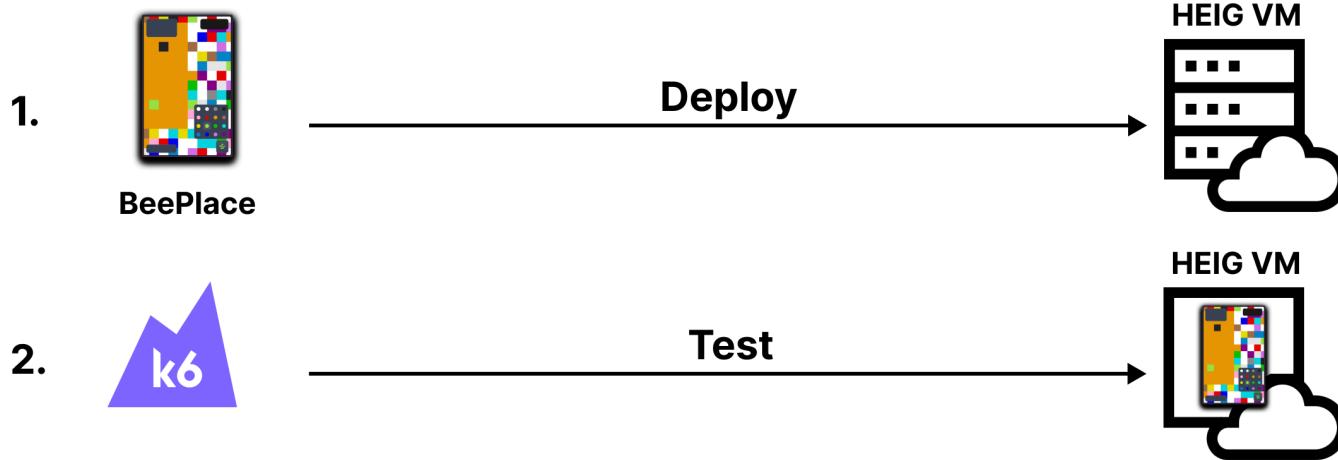


BeePlace

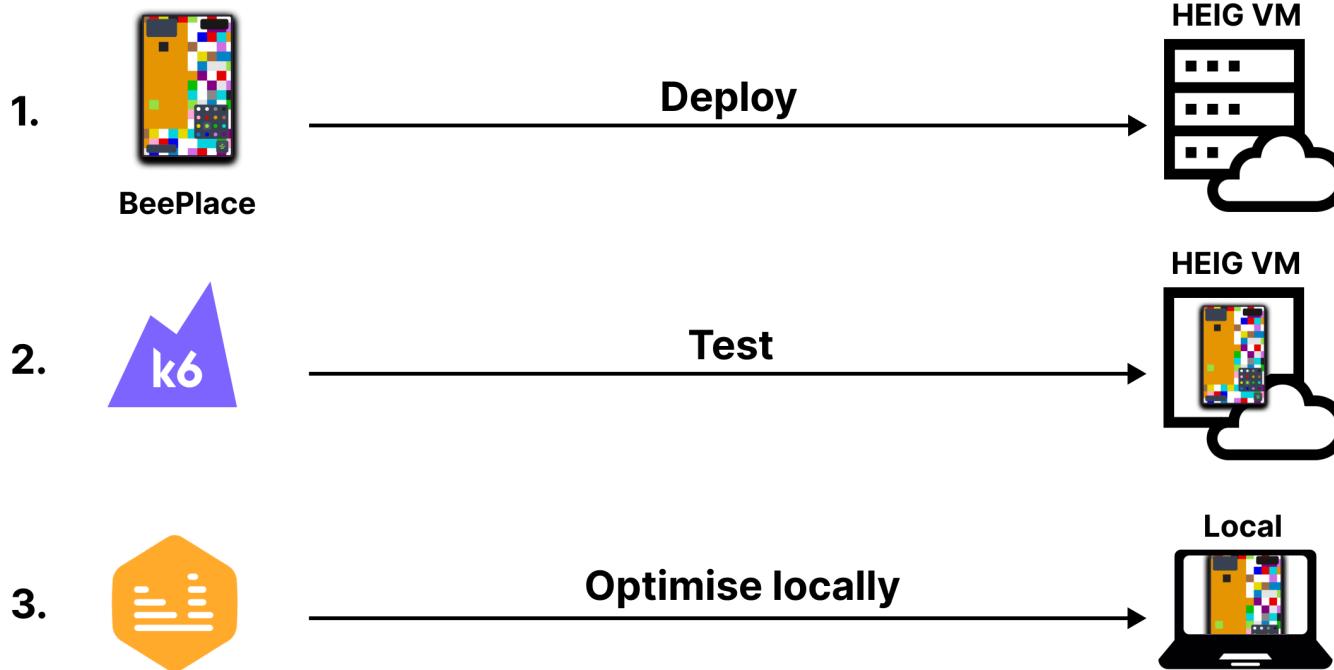
Deploy



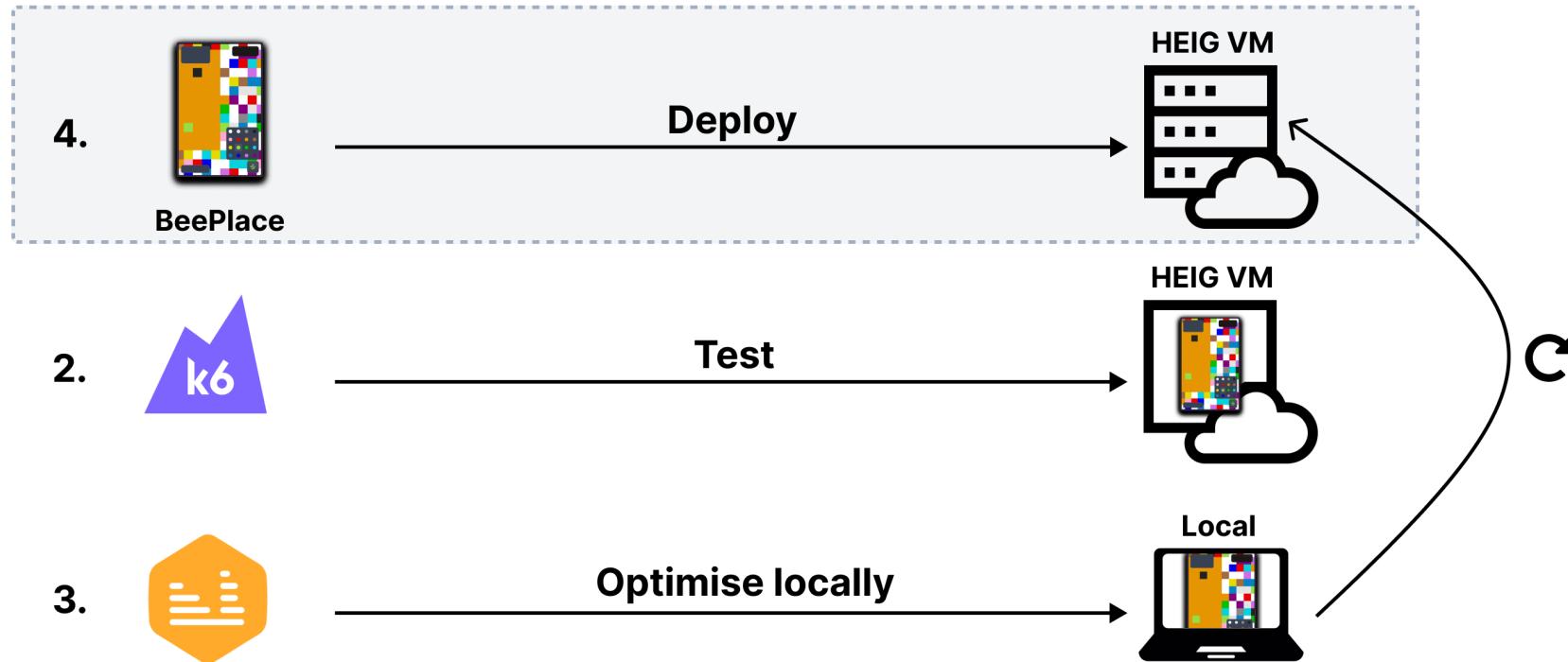
Méthodologie



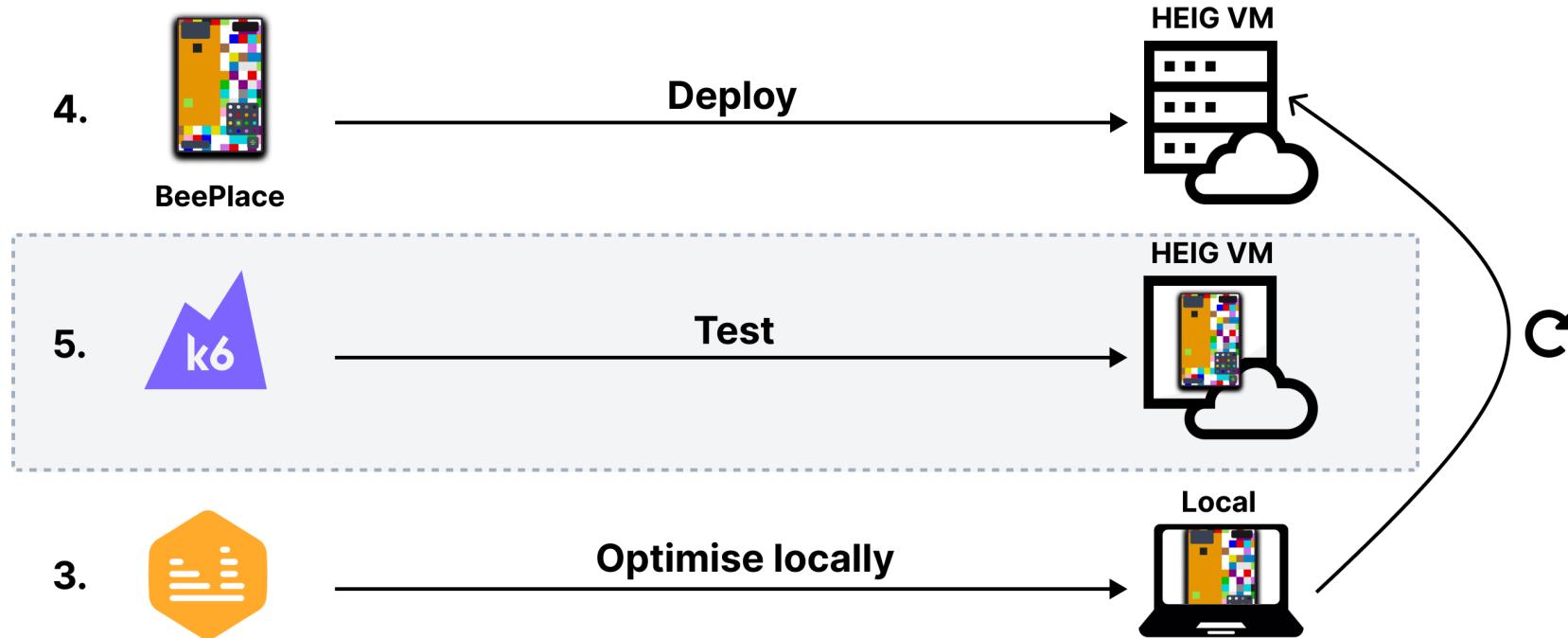
Méthodologie



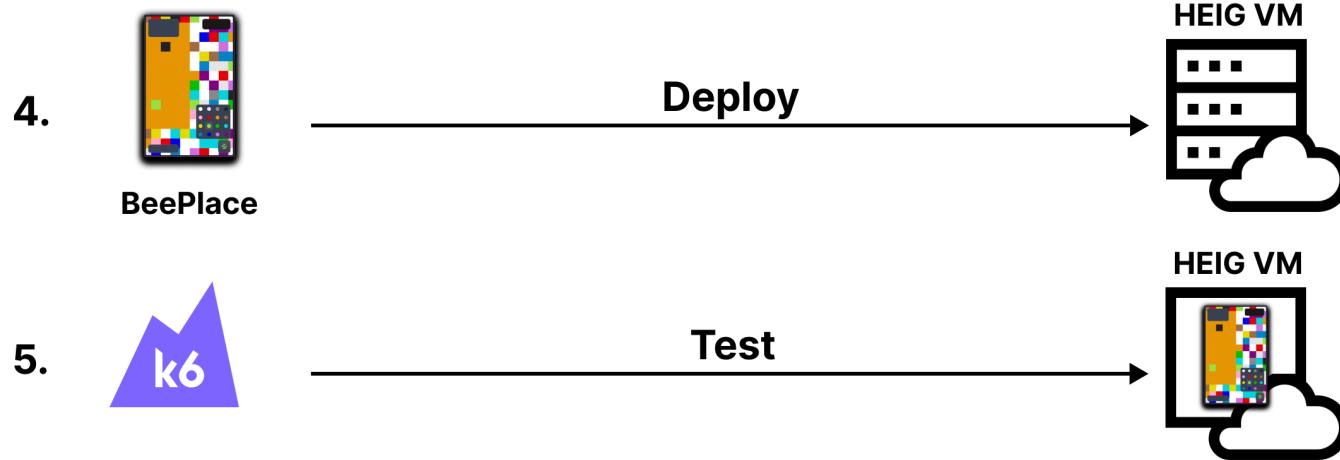
Méthodologie



Méthodologie



Méthodologie



6. Compare results ! 🎉

(repeat all steps until the application is as optimized as desired)

Résultats

Résultats initiaux

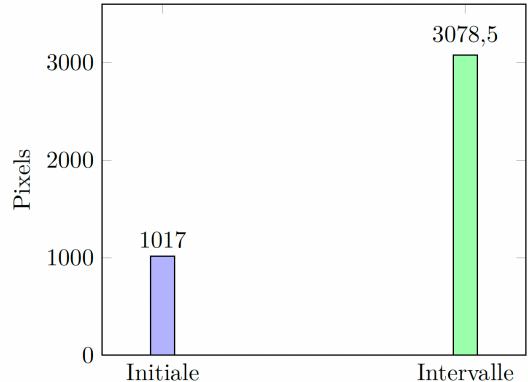
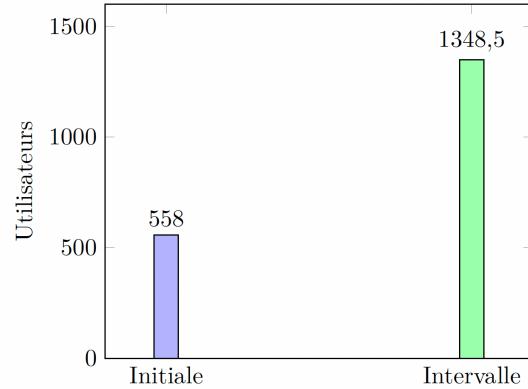
- Utilisateurs virtuels : **558**
- Pixels dessinés : **1017**

Résultats finaux

- Utilisateurs virtuels : **1348.5 (+ 141.67%)**
- Pixels dessinés : **3078.5 (+ 202.70%)**

Optimisations

- Broadcast des pixels avec un intervalle de temps
- Format plus léger pour l'envoi des pixels
- Cache du Bitfield Redis en mémoire
- Configuration de l'OS Linux



Conclusion

Conclusion technique

Cahier des charges

- Fonctionnalités *required* et *essential*
- Moitié des fonctionnalités *nice to have*
- Fonctionnalités non prévues

Objectifs atteints (Baleinev 2023)

- Moins de débordements
- Plus de collaboration
- Expérience plus positive que Pimp My Wall

Retour d'expérience

- Application fonctionnelle et déployée
- Technologies bien choisies, aucun réel blocage
- Optimisations concluantes, permet de tenir tous les festivaliers 
- Apprentissages utiles pour le futur

Améliorations possibles

- Accessibilité de l'app:
 - Tutoriel, textes informatifs
 - Internationalisation
- Tests unitaires et d'intégration

Conclusion personnelle

Technique

- Projet qui me tient à cœur
- Bénéfices de réaliser un projet plus long et conséquent
- Retours lors du Baleinev Festival 2023 encourageants



Organisationnel

- Travail seul sur un projet mais au sein d'une équipe
- Bonnes pratiques du monde professionnel (daily meeting, sprint review, ...)
- Review du code bénéfique, permet d'améliorer la qualité

Perspectives futures

Organisation

- Tests lors du Baleinev 2024
- Lier le monde physique pour résoudre les problèmes d'authentification

Développement

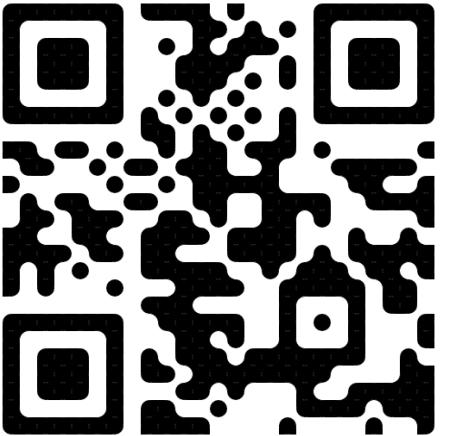
- Dashboard d'administration
- Statistiques
- Toile rectangulaire



Merci de votre attention !

À vos pixels !

place.beescreens.ch



Sources

- Photos : Antoine Kaelin & Kevin Pradervand
- Illustrations : unDraw



Annexes

1 Configuration

Display width: 960 Display height: 442

Number of columns: 3 Number of rows: 2

2 Choose a display

Show display 1 Show display 2 Show display 3

Show display 4 Show display 5 Show display 6

Display resolution: 960x442



