Sommaire:

Pour faire pousser sa propre digitale **2**
- Tout d'abord vérifier que vous avez installé toute les dépendances: 2
  - Installer docker : 2
  - Installer net-tools : 3
  - Installer make : 3
  - Installer le tableau de bord : 3
- Utiliser les commande suivante à la racine du projet: 3
  - Afficher l'aide intégré (en anglais): 3
  - Créer le fichier d'initialisation du projet (ladigitale.toml) : 3
  - Construire les images : 3
  - Générer les conteneurs: 3
- Ajout des websocket 4
  - Pour les projets PHP 4
  - Ils sont réalisés par l'intégration du fichier recupere_ip.php qui appelle NewWebsocket.js (le php récupère l'ip et le javascipt fait la connexion websocket) 4
  - NewWebsocket.js 4
  - recupere_ip.php 5
  - Pour les projet Node.js 5
- Projets hors ladigitale 7
  - Etherpad 7
  - Installation 7
- Projets PHP, serveur apache et base de données SQLite 7
  - Digiflashcards 7
  - Digimindmap 7
  - Digiscreen 7
  - Digisteps 8
  - Digiwords 8
  - Serveur PHP nécessaire pour l'API 8
  - Production 8
- Projets Nuxt.js avec serveur Node.js (Express) et base de données Redis 9
  - Digiboard 9
  - Digipad 9
  - Digistorm 9
  - Préparation et installation des dépendances 9
  - Compilation, minification des fichiers et lancement du serveur de production 9
    - Avec NPM 9
    - Avec PM2 9
  - Variables d'environnement pour la mise en production 10
    - Digiboard 10
    - Digistorm 10
    - Digipad 10
- Remerciements et crédits des applications originelles 11
- Remerciements et crédits des codes sources 11

Pour faire pousser sa propre digitale

Pour installer les 9 applications, utilisez le Makefile inclus.
Il se chargera de la séquence de commandes pour créer, lancer et orchestrer les images et conteneurs Docker !

Tout d'abord vérifier que vous avez installé toute les dépendances:

Installer docker :

```
sudo apt-get update

sudo apt-get install ca-certificates curl gnupg

sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg
--dearmor -o /etc/apt/keyrings/docker.gpg

sudo chmod a+r /etc/apt/keyrings/docker.gpg

echo \
  "deb [arch="$(dpkg --print-architecture)"
signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin
```

Installer net-tools :

```
sudo apt install net-tools
```

Installer make :

```
sudo apt install make
```

Installer le tableau de bord :

Il est nécessaire pour la stabilité de certaines applications (certaine plante a l'initialisation) et pour pouvoir suivre en temps réel l'utilisation des applications.

Utiliser les commande suivante à la racine du projet:

Afficher l'aide intégré (en anglais):

```
make help
```

Créer le fichier d'initialisation du projet (ladigitale.toml) :

Contenant les spécifications réseau du projet (ports, adresses ip et carte réseau).

- il est nécessaire pour générer les fichier .env et les commande build et run !

```
make env
```

Construire les images :

```
make all
make <application>
```

Générer les conteneurs:

* Les commandes compose utilisent les fichiers docker-compose.
* Les commandes run utilisent des commandes docker run standard.

```
make compose_all
make compose_<application>
ou
make run_all
make run_<application>
```

Pour installer des outils de La Digitale sur leur propre serveur sans passer par notre makefile et docker ou depuis les sources originales voici la marche à suivre:

- Lisez cet article de ladigitale qui contient des projets pré-compilés.
- Le dépôt Codeberg du projet contenant les codes source des applications.

Ajout des websocket

Pour les projets PHP

Ils sont réalisés par l'intégration du fichier recupere_ip.php qui appelle NewWebsocket.js (le php récupère l'ip et le javascipt fait la connexion websocket).

NewWebsocket.js

```
const ip = "ip";
const wss = new WebSocket("ws://172.19.0.2:50001")

$('document').ready(function(){
      $.ajax({
      type: 'GET',
      url: '../inc/recupere_ip.php',
      dataType: 'html',
      timeout: 500,
      success: function(reponse){
            json(reponse);
      }

      });
});

async function json(ip){
      setInterval(() => {
      console.log(ip)
      wss.onopen = function (event) {
            wss.send(JSON.stringify({ //send a JSON through
websocket that contains a list of students and the app they're in
                  type:"type_eleve",
                  name:"digiflashcard",
                  data:{id:ip, name:""}
```

```
            },
            {
                  type:"integration",
                  name:"digiboard",

logo:"https://pouet.chapril.org/system/accounts/avatars/000/096/847/original/841401129f94028b.png",
            }
            ));
      }
      }, 10000);
}
```

recupere_ip.php

```
<?php
function getIp(){
      if(!empty($_SERVER['HTTP_CLIENT_IP'])){
      $ip = $_SERVER['HTTP_CLIENT_IP'];
      }elseif(!empty($_SERVER['HTTP_X_FORWARDED_FOR'])){
      $ip = $_SERVER['HTTP_X_FORWARDED_FOR'];
      }else{
      $ip = $_SERVER['REMOTE_ADDR'];
      }
      return $ip;
  }
echo(getIp());
?>
```

Pour les projet Node.js

il faut inclure dans le fichier server.js (celui ou express met en place le router).

```
const WebSocket = require('ws');
const ws = new WebSocket("ws://webapp:50001")
let ipUtilisateurs = new Map();

async function envoiejson(){
      setInterval(() => {
      let data = []
      ipUtilisateurs.forEach(function(value, key) {
            data.push({id:value, name:key})
      });
```

```
	ws.addEventListener("open", () => {
		//console.log("Enjoy") //when the connection is opened
		ws.send(JSON.stringify({ //send a JSON through
websocket that contains a list of students and the app they're in
			type:"type_eleve",
			name:"digiboard",
			data:data
		},
		{
			type:"integration",
			name:"digiboard",

logo:"https://pouet.chapril.org/system/accounts/avatars/000/096/84
7/original/841401129f94028b.png",
		}
		));

	});

	}, 10000);
}

envoiejson()

app.get('/b/:tableau', function (req) {
	if (req.session.identifiant === '' || req.session.identifiant
=== undefined) {
	const identifiant = 'u' + Math.random().toString(16).slice(3)
	req.session.identifiant = identifiant
	req.session.nom = choisirNom() + ' ' + choisirAdjectif()
	req.session.langue = 'fr'
	req.session.statut = 'participant'
	req.session.tableaux = []
	req.session.cookie.expires = new Date(Date.now() +
dureeSession)
	ipUtilisateurs.set(identifiant, req.ip)
	}
	req.next()
})

socket.on('deconnexion', function (tableau) {
	ipUtilisateurs.delete(socket.handshake.session.identifiant)
```

```
      socket.to(tableau).emit('deconnexion',
socket.handshake.session.identifiant)
      })
```

Projets hors ladigitale

Etherpad

Etherpad est une application d'édition de document en ligne collaborative en temps réel, open source et hautement personnalisable.

- Publié sous licence Apache 2.

Installation

```
curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash -
sudo apt install -y nodejs
git clone --branch master
https://github.com/ether/etherpad-lite.git &&
cd etherpad-lite &&
src/bin/run.sh
```

Projets PHP, serveur apache et base de données SQLite

Digiflashcards

Digiflashcards est une application web simple pour concevoir des flashcards facilement.

- Publiée sous licence GNU GPLv3.
- Roboto Slab et Material Icons sous Apache License Version 2.0.
- KGrotesk sous Sil Open Font Licence 1.1.

Digimindmap

Digimindmap est une application web pour concevoir des cartes heuristiques, basée sur une version personnalisée et allégée de js My Mind.

- Publiée sous licence GNU GPLv3.
- librairie js My Mind sous MIT.
- Material Icons, Robot Slab sous Apache License Version 2.0.
- HKGrotesk sous Sil Open Font Licence 1.1.

Digiscreen

Digiscreen est un fond d'écran interactif pour la classe en présence ou à distance.

Cette version n'intègre pas les clés API Pixabay et Google pour YouTube.

- Publiée sous licence GNU GPLv3.
- Abril Fat Face, Orbitron et Material Icons sous Apache License Version 2.0.
- HKGrotesk sous Sil Open Font Licence 1.1.
- Pictographiques du module Histoire sont la propriété du Gouvernement d'Aragon, créé par Sergio Palao pour ARASAAC, distribué sous Licence Creative Commons BY-NC-SA.

Digisteps

Digisteps est une application web pour concevoir des parcours pédagogiques en ligne.

- Publiée sous licence GNU GPLv3.
- Roboto Slabet et Material Icons sous Apache License Version 2.0)
- HKGrotesk sous Sil Open Font Licence 1.1.

Digiwords

Digiwords est une application web pour créer des nuages de mots.

- Publiée sous licence GNU GPLv3.
- Abril Fat Face, Anton, Bahiana, Barrio, Finger Paint, Fredericka The Great, Gloria Hallelujah, Indie Flower, Life Savers, Londrina Sketch, Love Ya Like A Sister, Material Icons, Merienda, Pacifico, Quicksand, Righteous, Roboto et Slab sous Apache License Version 2.0.Robot
- HKGrotesk eT OpenDyslexic SOUS Sil Open Font Licence 1.1.

Serveur PHP nécessaire pour l'API

```
php -S 127.0.0.1:8000 (pour le développement uniquement)
```

Production

Le dossier src peut être déployé directement sur un serveur http/PHP avec l'extension SQLite activée comme Apache par exemple.

Projets Nuxt.js avec serveur Node.js (Express) et base de données Redis

Digiboard

Digiboard est une application web pour concevoir des tableaux blancs collaboratifs.

- Publiée sous licence GNU GPLv3.
- Material Icons sous Apache License Version 2.0.
- PHKGrotesk sous Sil Open Font Licence 1.1.

Digipad

Digipad est une application web pour concevoir des murs collaboratifs.

- Publiée sous licence GNU GPLv3.
- Roboto Slab et Material Icons sous Apache License Version 2.0.
- HKGrotesk sous Sil Open Font Licence 1.1.
- jsPanel4 sous MIT.
- pdf.js et viewer.js sous Apache License Version 2.0.

Digistorm

Digistorm est une application web pour concevoir des sondages, questionnaires, remue-méninges et nuages de mots collaboratifs.

- Publiée sous licence GNU GPLv3.
- Roboto Slab et Material Icons sous Apache License Version 2.0.
- HKGrotesk sous Sil Open Font Licence 1.1.

Préparation et installation des dépendances

```
npm install
```

Compilation, minification des fichiers et lancement du serveur de production

Avec NPM

```
npm run build
npm run start
```

Avec PM2

```
npm run build
pm2 start
```

Variables d'environnement pour la mise en production

Fichier .env à créer à la racine du dossier src.

Digiboard

```
DOMAIN (protocole + domaine. ex : https://digiboard.app)
HOST (IP publique du serveur de production)
PORT (port du serveur local nuxt.js / 3000 par défaut)
DB_HOST (IP du serveur de base de données Redis)
DB_PWD (mot de passe de la base de données Redis)
DB_PORT (port de la base de données Redis / 6379 par défaut)
SESSION_KEY (clé de session Express Session)
SESSION_DURATION (durée de la session de connexion des
utilisateurs en millisecondes)
```

Digistorm

```
DOMAIN (protocole + domaine. ex : https://digistorm.app)
HOST (IP publique du serveur de production)
PORT (port du serveur local nuxt.js / 3000 par défaut)
DB_HOST (IP publique du serveur de base de données Redis)
DB_PWD (mot de passe de la base de données Redis)
DB_PORT (port de la base de données Redis / 6379 par défaut)
SESSION_KEY (clé de session Express Session)
SESSION_DURATION (durée de la session de connexion des
utilisateurs en millisecondes)
(OPTIONAL)
EMAIL_HOST (hôte pour l'envoi d'emails)
EMAIL_ADDRESS (adresse pour l'envoi d'emails)
EMAIL_PASSWORD (mot de passe de l'adresse emails)
```

Digipad

```
DOMAIN (protocole + domaine. ex : https://digipad.app)
HOST (IP publique du serveur de production)
PORT (port du serveur local nuxt.js / 3000 par défaut)
DB_HOST (IP du serveur de base de données Redis)
DB_PWD (mot de passe de la base de données Redis)
DB_PORT (port de la base de données Redis / 6379 par défaut)
SESSION_KEY (clé de session Express Session)
SESSION_DURATION (durée de la session de connexion des
utilisateurs en millisecondes)
(OPTIONAL)
```

```
ETHERPAD (lien vers un serveur Etherpad pour les documents
collaboratifs)
ETHERPAD_API_KEY (clé API Etherpad)
UPLOAD_LIMIT (limite de téléversement des fichiers en Mo)
PAD_LIMIT (nombre maximum de pads par compte utilisateur)
ADMIN_PASSWORD (mot de passe pour accéder à la page
d'administration /admin)
EMAIL_HOST (hôte pour l'envoi d'emails)
EMAIL_ADDRESS (adresse pour l'envoi d'emails)
EMAIL_PASSWORD (mot de passe de l'adresse emails)
EMAIL_PORT (port pour l'envoi d'emails)
EMAIL_SECURE (true ou false)
MATOMO (lien vers un serveur Matomo)
NFS_PAD_NUMBER (id de pad à partir de laquelle les fichiers seront
enregistrés dans un dossier monté NFS - environ 200 000 pour 1 To
de capacité disque)
NFS_FOLDER (nom du dossier monté NFS, obligatoirement situé dans
le dossier /static/)
```

Remerciements et crédits des applications originelles

Traduction en allemand lors d'une action dans « la salle des profs » #twitterlehrerzimmer / #twlz par @uivens (Ulrich Ivens), @eBildungslabor (Nele Hirsch) et @teachitalizer (Holger Skarba).

Traduction en espagnol par Fernando S. Delgado Trujillo.

Traduction en Italien par Paolo Mauri et @nilocram (Roberto Marcolin).

Traduction en Croate par Ksenija Lekić.

Traduction en néerlandais par Erik Devlies.

Remerciements et crédits des codes sources

ladigitale.dev pour les sources originales des applications, vous pouvez soutenir leur travail ici, The Etherpad Foundation pour les sources originales d'Etherpad.

KAZ pour le dockerfile du serveur etherpad.

Et enfin Monsieur Merciole pour son aide, ses précieux conseils ainsi que son chaperonnage du projet et l'organisation des réunions hebdomadaires.