Documentation Technique

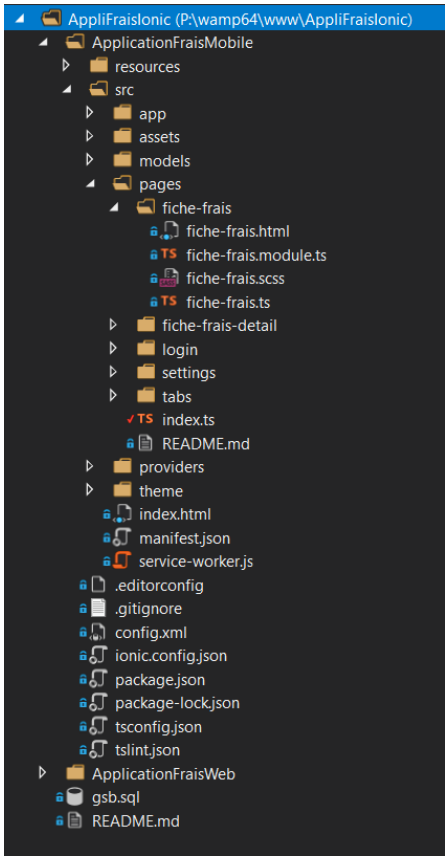
Contexte GSB Mobile - PPE

Sommaire

I) Principes de l'application AppliFrais

Cette application permet aux visiteurs médicaux de visualiser les différents frais qu'ils ont eus dans le cadre de leurs missions.

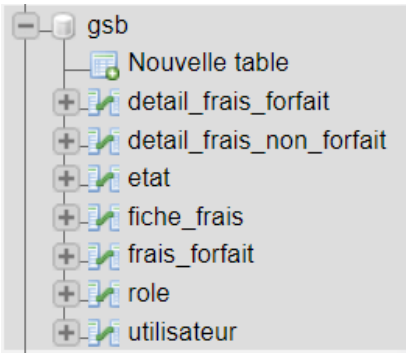
A) Arborescence du site



L'application mobile a été réaliser avec le Framework Ionic qui permet de créer des applications inter-plateformes :

- "models" ⟶ l'endroit où sont définie les objets et le type de données qu'ils contiennent.
- "pages" ⟶ les dossiers contenant les informations nécessaire pour l'affichage de chaque page (HTML, CSS, TS).
- "provider" ⟶ extraction des données à partir de Wamp.
- "ApplicationFraisWeb" ⟶ contient toutes les requetes SQL.
- "gsb.sql" ⟶ la base de données à importer sur phpMyAdmin
- "index.php" ⟶ Page qui gère le Menu de redirection

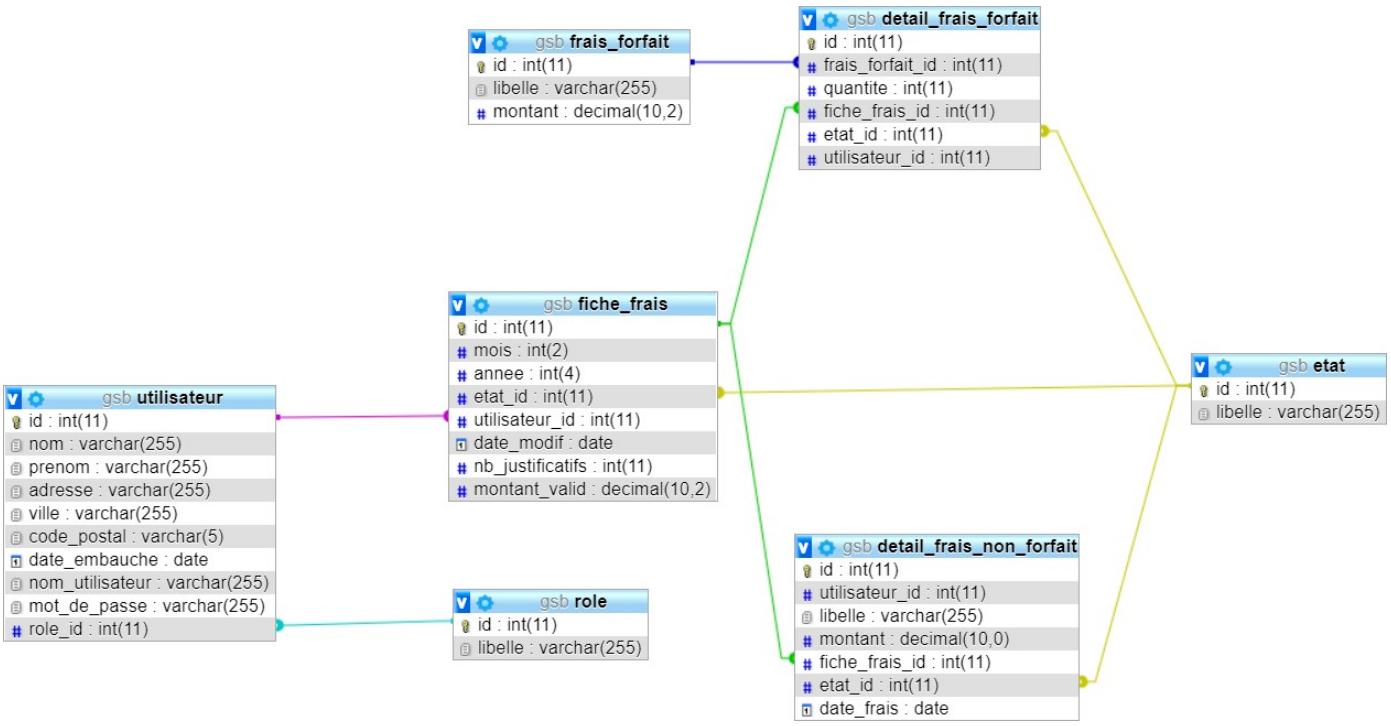
B) La base de données



La base de données est gérée sur phpMyAdmin en local. Elle contient 7 tables qui sont lié entre elle.

Ex : Chaque fiche de frais et lié à un visiteur. Une fiche a elle-même une ou plusieurs ligne de frais hors-forfait et chaque fiche a son propre état

Schéma de conception :



II) Les différentes fonctionnalités

A. Authentification de l'utilisateur

Avant de pouvoir visualiser ses frais, le visiteur doit d'abord s'authentifier à l'aide de son login et de son mot de passe dans le formulaire de connexion.

Lorsqu'un utilisateur est authentifié, son nom et son prénom apparaisse dans l'entête.

La views :

```
<form (submit)="doLogin()">
  <ion-list inset>
    <ion-item>
      <ion-label >
        <ion-icon name="person" style="font-size: xx-large;" color="primary"></ion-icon>
      </ion-label>
      <ion-input type="email" [(ngModel)]="utilisateur.nomUtilisateur" name="email" placeholder="Username"></ion-input>
    </ion-item>
    <br />
    <ion-item>
      <ion-label>
        <ion-icon name="unlock" style="font-size: xx-large;" color="primary"></ion-icon>
      </ion-label>
      <ion-input type="password" [(ngModel)]="utilisateur.motDePasse" name="password" placeholder="Password"></ion-input>
    </ion-item>
    <br />
    <div style="margin-top: 25px;">
      <button ion-button color="primary-shade" block>Connexion</button>
    </div>
  </ion-list>
</form>
```

TypeScript :

```
export class LoginPage {

  utilisateur = new Utilisateur();

  constructor(public navCtrl: NavController,
    public user: User,
    public toastCtrl: ToastController,
    public translateService: TranslateService) {

  }

  doLogin() {
    this.user.login(this.utilisateur).subscribe((resp) => {
      if (resp['succes']) {
        this.user.utilisateurId = resp['utilisateur_id'];
        this.user.nom = resp['nom'];
        this.user.prenom = resp['prenom'];
        this.user.nomUtilisateur = resp['nom_utilisateur'];
        this.navCtrl.push('TabsPage');
      }else{
        let toast = this.toastCtrl.create({
          message: 'Identifiants incorrects',
          duration: 3000,
          position: 'bottom'
        });
        toast.present();
      }
    }, (err) => {
      // Unable to log in
      let toast = this.toastCtrl.create({
        message: 'Une erreur est survenue',
        duration: 3000,
        position: 'bottom'
      });
      toast.present();
    });
  }
}
```

Lorsqu'une demande de connexion arrive on charge la page de connexion puis on une fois le bouton cliqué on passe a la validation .

Grace a la fonction login on recupere les differentes informations posséder sur un utilisateur et on compare si le login et le mot de passe saisi a une correspondance dans la base.
Si ce n'est pas le cas ⟶ afficher message erreur.

Les infos d'un visiteur ⟶ this.user.nom ,prenom

¡d Fonction déconnection ⟶ logout() ;

Provider :

```
login(utilisateur: Utilisateur) {
  let httpParams = new HttpParams();
  httpParams = httpParams.append('nom_utilisateur', utilisateur.nomUtilisateur);
  httpParams = httpParams.append('mot_de_passe', utilisateur.motDePasse);
  let options = {
    headers: new HttpHeaders().set('Content-Type', 'application/x-www-form-urlenc
  };

  return this.api.post('connexion.php', httpParams, options);
}

logout() {
  this.utilisateurId = null;
  this.nom = null;
  this.prenom = null;
  this.nomUtilisateur = null;
}
}
```

B. Visualiser ses fiches de frais (Visiteur)

Views	Pages/fiche-frais.html	Affichage des fiches de frais de l'utilisateur connecté
Méthode du provider fiche-frais.ts	getAll()	On récupère les fiches de frais de l'utilisateur connecté
	getDetailById(id : string)	On récupère toutes les informations sur la fiche souhaitée, grâce à l'ID de la fiche de frais
Méthode de fiche-frais.ts	Refresh()	Actualise l'affichage des fiches de frais
	goToDetail(fiche)	Accède au détail de la fiche sélectionnée.

```
@Injectable()
export class FicheFraisProvider {

  constructor(public api: Api, public userService: User) {
    console.log('Hello FicheFraisProvider Provider');
  }

  getAll(){
    return this.api.get('fiche_frais.php?utilisateur=' + this.userService.utilisateurId);
  }
```

On récupère les fiches de frais de l'utilisateur connecté grâce au getAll(), en utilisant l'ID de l'utilisateur.

Voici également le code en PHP qui récupère les fiches de frais.

fiche_frais.php | fiche-frais.ts | fiche-frais.ts | fiche-frais.scss | fiche-frais.html | api.ts | login.html

```php
<?php

header('Access-Control-Allow-Origin: *');
$host_name = 'localhost';
$database = 'gsb';
$user_name = 'root';
$password = '';

$bdd = new PDO("mysql:host=$host_name; dbname=$database;", $user_name, $password);

$query = $bdd->prepare('SELECT `id`, `mois`, `annee` FROM fiche_frais WHERE utilisateur_id = :utilisateur
$query->bindParam(':utilisateur', $_GET['utilisateur']);
$query->execute();
$fiches = $query->fetchAll();

$retour = [];
for ($i=0; $i < count($fiches); $i++) {
      $retour['fiches_frais'][$i]['id'] = $fiches[$i]['id'];
      $retour['fiches_frais'][$i]['mois'] = $fiches[$i]['mois'];
      $retour['fiches_frais'][$i]['annee'] = $fiches[$i]['annee'];
}


echo json_encode($retour);

```

```
  getDetailById(id: string) {
    return this.api.get('detail.php?fiche_frais_id=' + id);
  }
}
```

detail.php | fiche_frais.php | fiche-frais.ts | fiche-frais.ts | fiche-frais.scss | fiche-frais.html | api.ts

```php
$user_name = 'root';
$password = '';

$bdd = new PDO("mysql:host=$host_name; dbname=$database;", $user_name, $password);

$queryDFF = $bdd->prepare('
    SELECT *
    FROM detail_frais_forfait, frais_forfait
    WHERE detail_frais_forfait.frais_forfait_id = frais_forfait.id
    AND fiche_frais_id = :fiche_frais_id
');
$queryDFF->bindParam(':fiche_frais_id', $_GET['fiche_frais_id']);
$queryDFF->execute();
$detailFraisForfait = $queryDFF->fetchAll();

$queryDFNF = $bdd->prepare('SELECT * FROM detail_frais_non_forfait WHERE fiche_frais_id = :fiche_frais_id');
$queryDFNF->bindParam(':fiche_frais_id', $_GET['fiche_frais_id']);
$queryDFNF->execute();
$detailFraisNonForfait = $queryDFNF->fetchAll();

$total = 0;

$retour = [
    'detail_frais_forfait' => array(),
    'detail_frais_non_forfait' => array()
];

for ($i=0; $i < count($detailFraisForfait); $i++) {
    $total = $total + ($detailFraisForfait[$i]['quantite'] * $detailFraisForfait[$i]['montant']) ;
  /*  $datas = array(
```

90 %

On récupère les informations de la fiche de frais sélectionnée grâce à son ID.

Voici également le code en PHP qui récupère les détails des fiches de frais.