Compte rendu Développement en PHP de la partie web de l'application

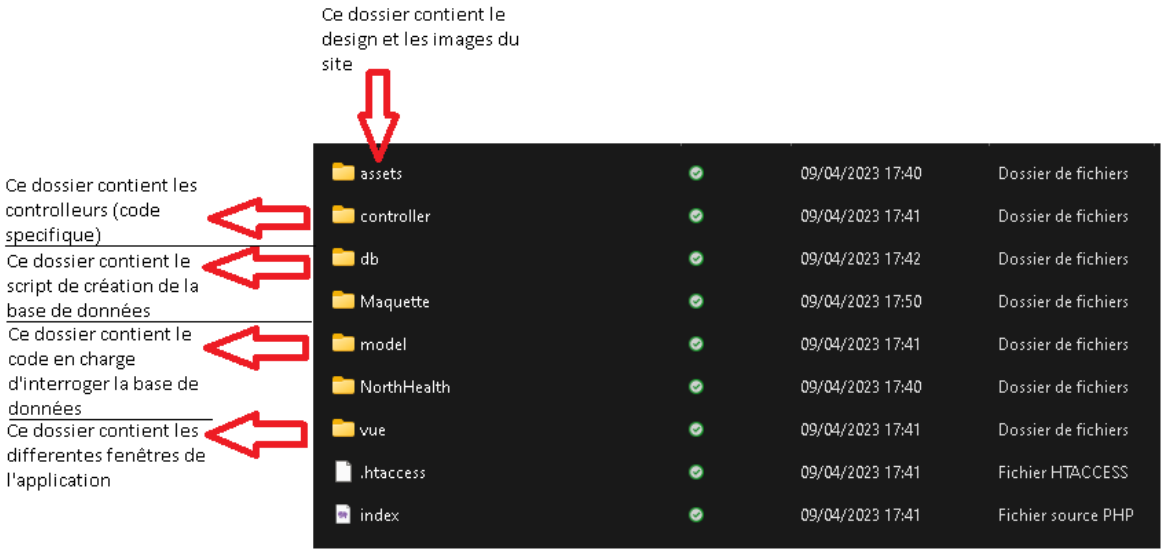
Table des matières :

1. Logiciels utilisés :

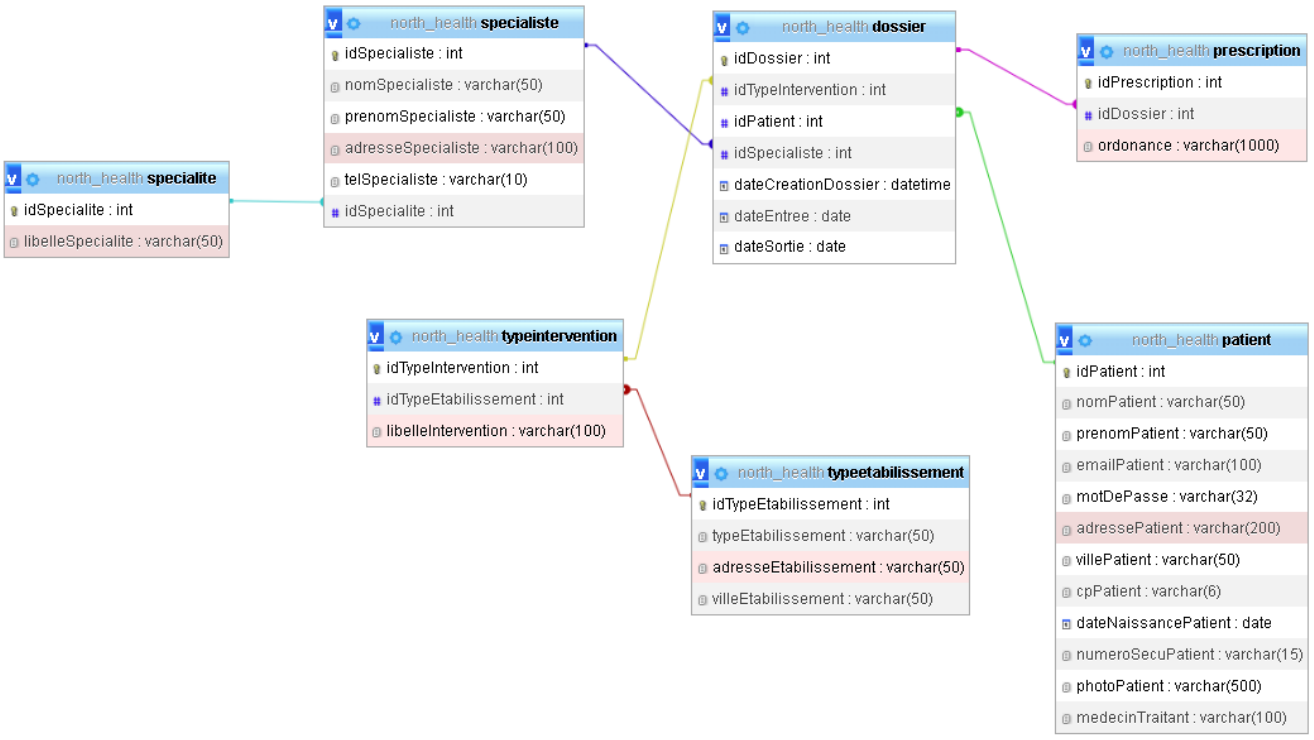
- Visual Studio Code
- MySQL
- Wampserver 64
- Windows 64 bit
- Git et Github
-

2. Arborescence des fichiers

Compte rendu Développement en PHP de la partie web de l'application



3. Schéma de la base de donnés :



Dossier : Cette table contient les données relatives aux consultations des patients.

Patient : Cette table contient les informations relatives aux patients.

Spécialiste : Cette table contient les informations relatives aux spécialistes.

4. Modification code source

Mission 1 : Création de la méthode de classe « addPatient » qui va permettre au patient de créer un nouveau compte.

```php
class NorthHealthContext{

    private $connectionStat = null;
    private $OPTIONS = array (
        PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8mb4"
    );

    private function connectToMySQLDataBase(){
        try{
            $this->connectionStat = new PDO(DSN, USER, PWD, $this->OPTIONS);
        }catch(PDOException $e){
            throw "Error: ".$e->getMessage();
        }
    }

    public function addPatient($data){
        $this->connectToMySQLDataBase();
        if($this->connectionStat != null){
            $queryString = "INSERT INTO `patient` (`nomPatient`, `prenomPatient`, `emailPatient`, `motDePasse`, `adressePatient`, `villePatient`, `cpPatient`, `dateNaissan
            $queryPrepared = $this->connectionStat->prepare($queryString, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
            $row = $queryPrepared->execute($data);
            $this->connectionStat = null;
            if($row > 0){
                return True;
            }
            return False;
        }
        return null;
    }
```

Création de la méthode de classe « ifUserExist » qui va permettre à un patient qui possède déjà un compte de se connecter.

Compte rendu Développement en PHP de la partie web de l'application

```php
public function ifUserExist($data){
    $this->connectToMySQLDataBase();
    if($this->connectionStat != null){
        $queryString = "SELECT * FROM `patient` WHERE `emailPatient` = :email AND `motDePasse` = :pwd";
        $queryPrepared = $this->connectionStat->prepare($queryString, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
        $queryPrepared->execute($data);
        $user = $queryPrepared->fetch(PDO::FETCH_ASSOC);
        if(!empty($user)){
            return $user;
        }
        return false;
    }
    return false;
}
```

Mission 2 :Afin que le patient puisse modifier ses informations personnelles j'ai créé la méthode de classe «modifierInfoPatient »

```php
public function modifierInfoPatient($idPatient, $data){

    $this->connectToMySQLDataBase();
    if($this->connectionStat != null){

        try{
            $data['idPatient'] = $idPatient;
            if(isset($data['numSecu'])){
                $queryString = "UPDATE `patient` SET `emailPatient`=:email, `adressePatient`=:adresse, `villePatient`=:ville, `cpPatient`=:cp, `numeroSecuPatient`=:numSe
            }else{
                $queryString = "UPDATE `patient` SET `emailPatient`=:email, `adressePatient`=:adresse, `villePatient`=:ville, `cpPatient`=:cp WHERE `idPatient` =:idPatie
            }

            $queryPrepared = $this->connectionStat->prepare($queryString, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
            $queryPrepared->execute($data);
            return true;
        }catch(PDOException $ex){
            return false;
        }
        return false;
    }
```

Mission 3 : Création de la méthode de classe « addRdv » et qui va permettre à un patient de prendre un rdv.

```php
public function addRdv($data){
    $this->connectToMySQLDataBase();
    if($this->connectionStat != null){
        try{
            $queryString = "INSERT INTO `dossier` (`codeCommune`, `libelleIntervention`, `idTypeEtabilissement`, `idSpecialiste`, `creneau`, `idPatient`)
            VALUES (:ville, :typeexam, :typeetab, :praticien, :creneau, :idPatient)";
            $queryPrepared = $this->connectionStat->prepare($queryString, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
            $row = $queryPrepared->execute($data);
            $this->connectionStat = null;
            if($row > 0){
                $this->updateCreneau($data["creneau"], $data["praticien"], "R");
                return True;
            }
        }catch(PDOException $e){
            throw "Error: ".$e->getMessage();
        }

        return False;
    }
    return null;
}
```

Mission 4 : Les fonctions « nextRdv » et « previousRDV » vont permettre au patient de consulter leurs anciens et prochains rdv.

```php
public function nextRdv($idPatient){

    $this->connectToMySQLDataBase();
    if($this->connectionStat != null){
        $queryString = "SELECT d.*, s.nomSpecialiste, s.prenomSpecialiste, s.telSpecialiste, c.nomCommune, c.codePostal, te.typeEtabilissement, te.adresseEtabilissement
        FROM dossier d
        INNER JOIN specialiste s
        ON d.idSpecialliste = s.idSpecialiste
        INNER JOIN typeetabilissement te
        ON te.idTypeEtabilissement = d.idTypeEtabilissement
        INNER JOIN commune c
        ON c.codePostal = d.codeCommune
        WHERE d.idPatient = :idPatient
        AND d.creneau > CURRENT_DATE";

        $queryPrepared = $this->connectionStat->prepare($queryString, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
        $queryPrepared->execute(array("idPatient"=>$idPatient));
        $nextRdv = $queryPrepared->fetchAll(PDO::FETCH_ASSOC);
        if(!empty($nextRdv)){
            return $nextRdv;
        }
        return false;
    }
    return null;
}
```

Compte rendu Développement en PHP de la partie web de l'application

```php
public function previousRdv($idPatient){

    $this->connectToMySQLDataBase();
    if($this->connectionStat != null){
        $queryString = "SELECT d.*, s.nomSpecialiste, s.prenomSpecialiste, s.telSpecialiste, c.nomCommune, c.codePostal, te.typeEtabilissement, te.adresseEtabilissement
        FROM dossier d
        INNER JOIN specialiste s
        ON d.idSpecialiste = s.idSpecialiste
        INNER JOIN typeetabilissement te
        ON te.idTypeEtabilissement = d.idTypeEtabilissement
        INNER JOIN commune c
        ON c.codePostal = d.codeCommune
        WHERE d.idPatient = :idPatient
        AND d.creneau <= CURRENT_DATE";

        $queryPrepared = $this->connectionStat->prepare($queryString, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
        $queryPrepared->execute(array("idPatient"=>$idPatient));
        $nextRdv = $queryPrepared->fetchAll(PDO::FETCH_ASSOC);
        if(!empty($nextRdv)){
            return $nextRdv;
        }
        return false;
    }
    return null;
}
```

Mission 5 : Création de la méthode de classe « addExam » qui va permettre à un admin d'ajouter des examens et « addSpecialiste » pour ajouter un spécialiste

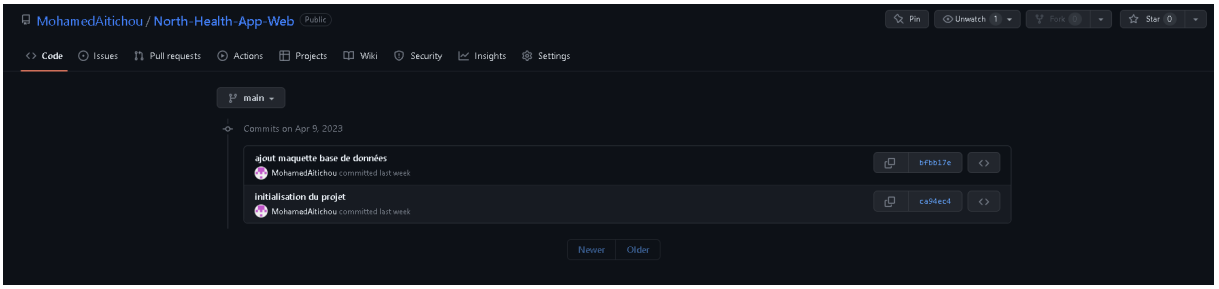
```php
public function addExam($data){

    $this->connectToMySQLDataBase();
    if($this->connectionStat != null){
        $queryString = "INSERT INTO `typeintervention` (`idTypeEtabilissement`, `libelleIntervention`) VALUES (:etabId, :libelleExam)";
        $queryPrepared = $this->connectionStat->prepare($queryString, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
        $row = $queryPrepared->execute($data);
        $this->connectionStat = null;
        if($row > 0){
            return True;
        }
        return False;
    }
    return null;
}
```

```php
public function addSpecialiste($data){
    $this->connectToMySQLDataBase();
    if($this->connectionStat != null){
        $queryString = "INSERT INTO `specialiste` (`nomSpecialiste`, `prenomSpecialiste`, `adresseSpecialiste`, `codePostalSpecialiste`, `telSpecialiste`, `idSpecialite`
        $queryPrepared = $this->connectionStat->prepare($queryString, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
        $row = $queryPrepared->execute($data);
        $this->connectionStat = null;
        if($row > 0){
            return True;
        }
        return False;
    }
    return null;
}
```

5. Gestion de version

Git :



Compte rendu Développement en PHP de la partie web de l'application

Github :

https://github.com/MohamedAitichou/North-Health-App-Web

6. Compétences

A4.1.1 , Proposition d'une solution applicative.

A4.1.2 , Conception ou adaptation de l'interface utilisateur d'une solution applicative.

A4.1.3 , Conception ou adaptation d'une base de données.

A4.1.6 , Gestion d'environnements de développement et de test.

A4.1.7 , Développement, utilisation ou adaptation de composants logiciels.

A4.1.9 , Rédaction d'une documentation technique.

A4.1.10 , Rédaction d'une documentation d'utilisation.

A4.2.1 , Analyse et correction d'un dysfonctionnement, d'un problème de qualité de …

A4.2.2 , Adaptation d'une solution applicative aux évolutions de ses composants.

A4.2.3 , Réalisation des tests nécessaires à la mise en production d'éléments mis à jour.

A4.2.4 , Mise à jour d'une documentation technique.