

Activité professionnel 2023-2024

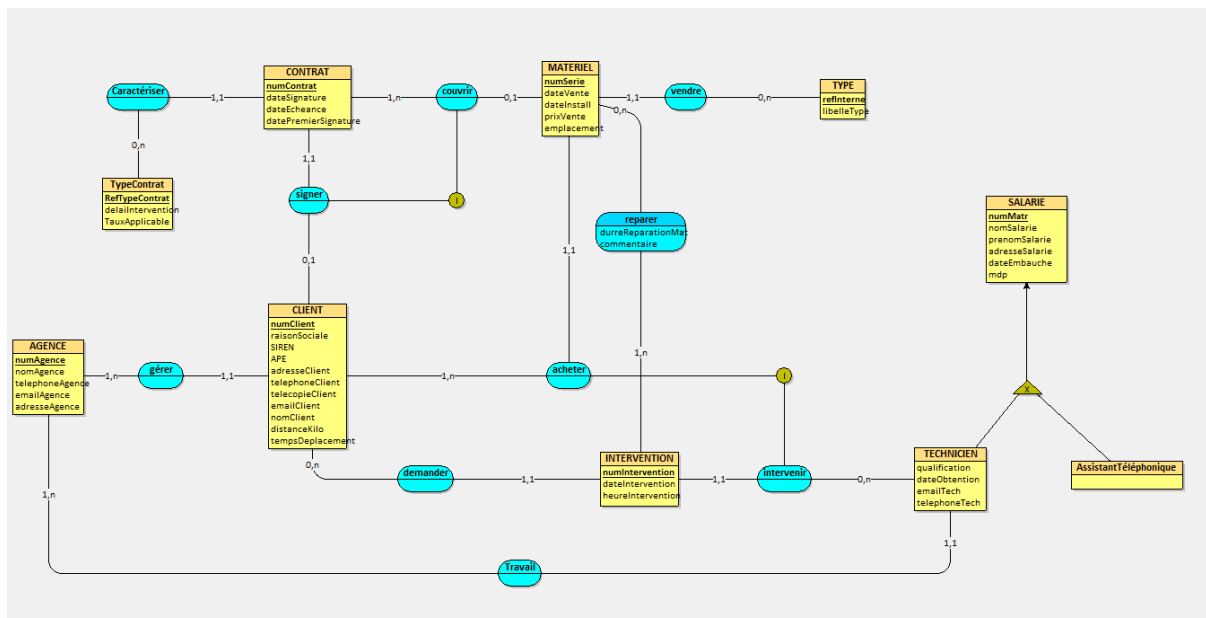
Contexte

La société CashCash vend des terminaux « point de vente » (TPV) à des hypermarchés, supermarchés et petits commerçants. CashCash compte de nombreux sites en France (centres régionaux et agences). Elle dispose d'un SI (système d'information) composé de plateformes hétérogènes (Mac OS, Linux, Windows, IOS, Android, etc) construit sur un réseau IP.

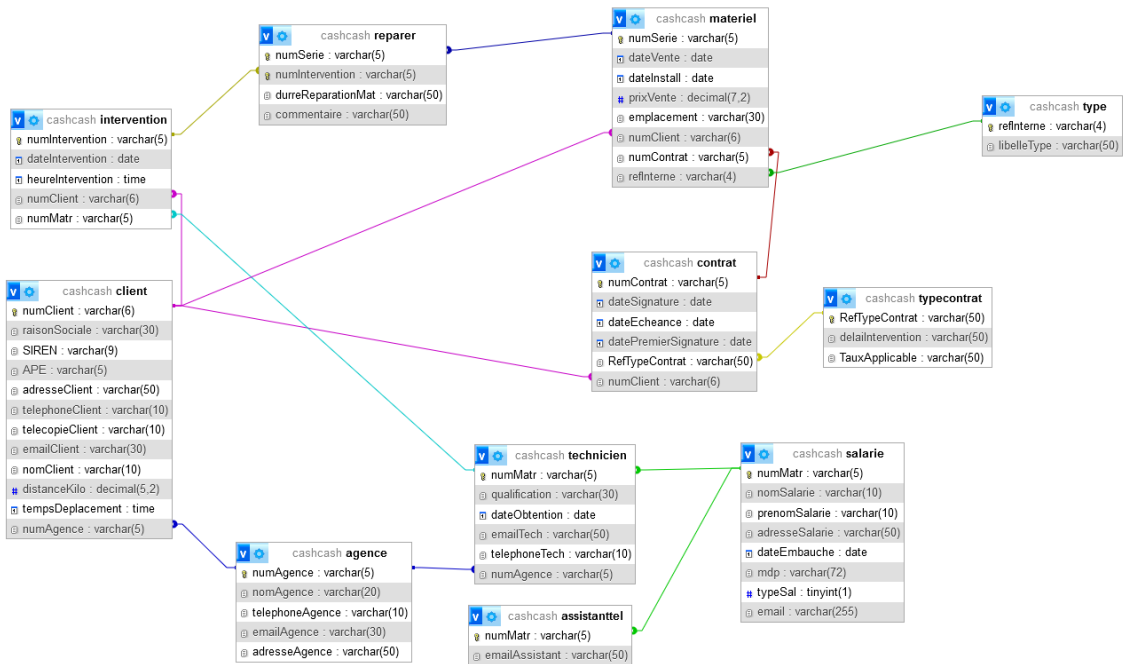
Il nous a été demandé de mettre en place un site web qui permet à l'assistant téléphonique et techniciens de gérer la réparation des machines sous contrat de réparation.

Base de données

MCD :



MLD :



Détails

Nom du déclencheur

verifSal

Table

intervention

Moment

BEFORE

Évènement

UPDATE

Définition

```

BEGIN
  DECLARE verif INT;
  IF new.numMatr IS NOT NULL THEN
    SELECT COUNT(*) INTO verif FROM technicien WHERE numMatr = new.numMatr;
    IF verif = 0 THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Le technicien n existe pas';
    END IF;
  END IF;
END

```

Créateur

root@localhost

Détails

Nom du déclencheur

verifSalCli

Table

intervention

Moment

BEFORE

Évènement

UPDATE

Définition

```

BEGIN
  DECLARE getNumAgSal VARCHAR(10);
  DECLARE getNumAgCli VARCHAR(10);
  SELECT numAgence INTO getNumAgSal FROM technicien WHERE new.numMatr = numMatr;
  SELECT numAgence INTO getNumAgCli FROM client WHERE new.numClient = numClient;
  IF getNumAgSal != getNumAgCli THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Le client et le salarié ne possèdent pas le même numéro d agence';
  END IF;
END

```

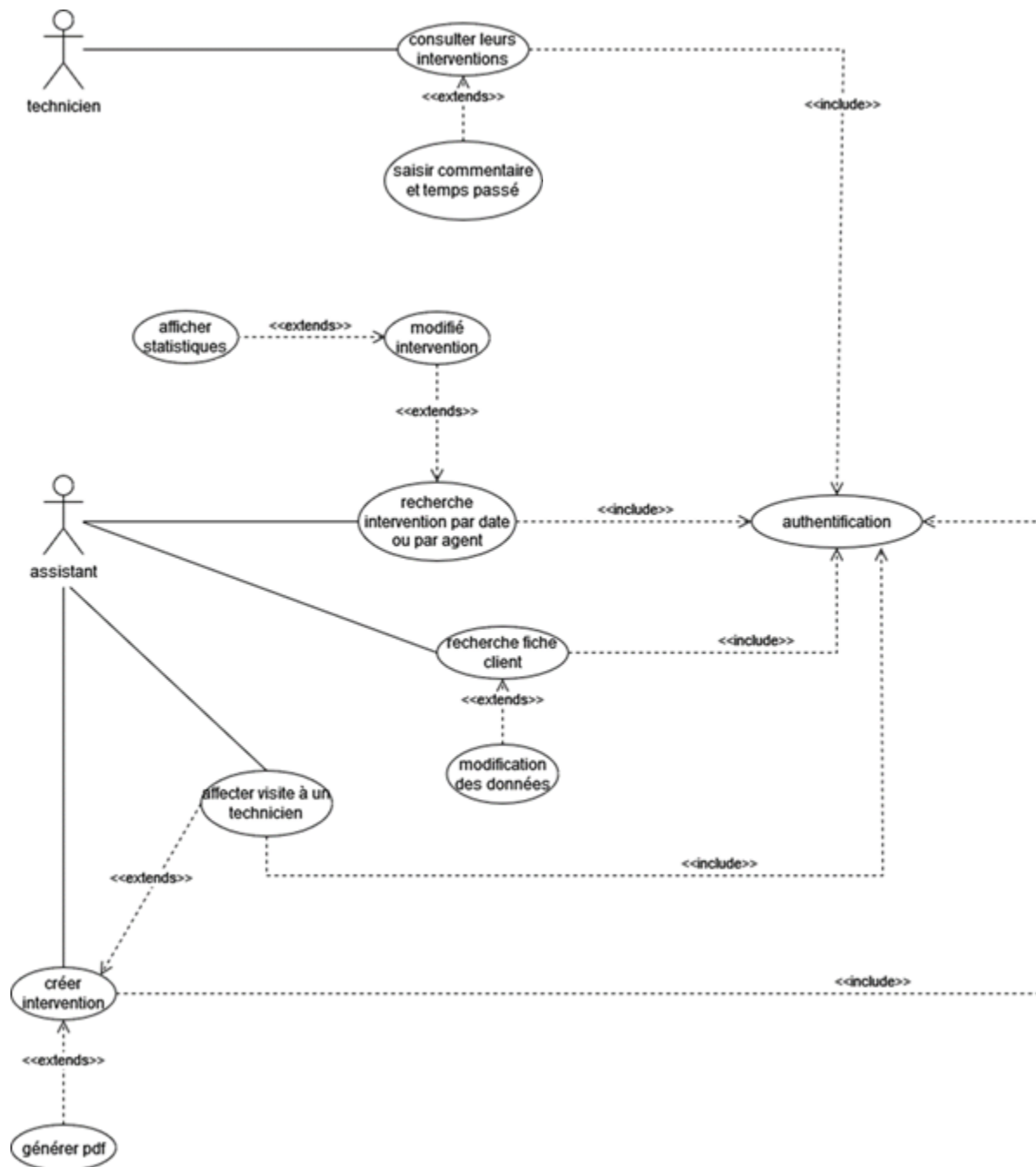
Créateur

root@localhost

Technologies utilisées

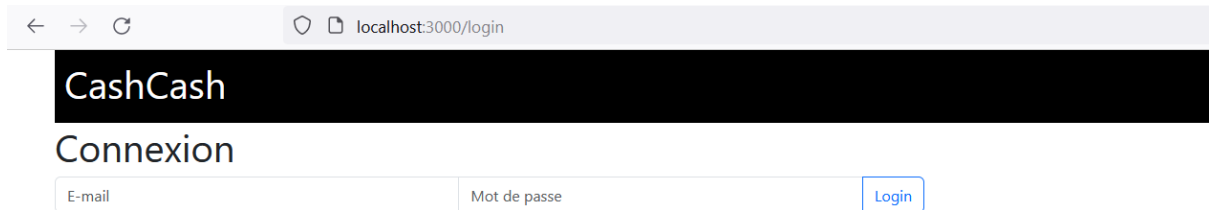
Pour ce projet on a décidé de travailler sur le logiciel node.js, avec le Framework express. La raison c'est le fait que c'est toujours un outil populaire et grandement utilisé qui facilitera le travail des futurs développeurs. De plus, l'outil est moderne et est fonctionnel pour des systèmes d'exploitation récents. On utilise un plug-in sur vs code nommé LiveShare qui permet à des collaborateurs de modifier le code en direct.

Diagramme de cas d'utilisation



Présentation des fonctionnalités

1) Page de connexion et bouton déconnexion

A screenshot of a web browser showing the login page of an application named 'CashCash'. The browser's address bar shows 'localhost:3000/login'. The page has a dark header with the 'CashCash' logo. Below the header, the word 'Connexion' is displayed. There are two input fields: one for 'E-mail' and one for 'Mot de passe' (Password). A blue 'Login' button is positioned to the right of the password field.

CashCash

Connexion

E-mail Mot de passe Login

login.pug

```
extends layout

block content
  h1 Connexion
  form(action="", method="post")
    div(class="input-group mb-3" style="width:50%")
      input#username(type="text" class="form-control" name="email",
placeholder="E-mail" , required aria-label="Fiche client" aria-
describedby="basic-addon2")
      input#password(type="password" class="form-control" name="mdp",
placeholder="Mot de passe" , required aria-label="Fiche client" aria-
describedby="basic-addon2")
      input(type="submit", value="Login" class="btn btn-outline-primary")
      //img(src="images/weed.jpg")
  if messageErr
    div(class="alert alert-danger" role="alert")= messageErr
```

login.js

```
// Ajout des modules nécessaires
const express = require('express');
const session = require('express-session');
const path = require('path');
const bcrypt = require('bcrypt');
const router = express.Router();
const connection = require('../connect');

// Gérer les requêtes GET à la racine '/'
router.get('/', function(req, res, next) {
  // Rendre la vue 'login' pour afficher le formulaire de connexion
  res.render('login');
});

// Gérer les requêtes POST à la racine '/'
router.post('/', function(req, res) {
  // Extraire les données du formulaire de connexion
  let email = req.body.email;
  let mdp = req.body.mdp;

  // Vérifier si l'email et le mot de passe ont été fournis
  if (email && mdp) {

    // Requête SQL pour vérifier l'authentification de l'utilisateur
    connection.query('SELECT * FROM salarie where salarie.email = ?', [email],
function(error, data) {
  if (error) throw error;

  if (data.length > 0) {
    bcrypt.compare(mdp, data[0].mdp, function(err, result) {
      if (result){
        // Enregistrer les informations de session et rediriger vers la page d'accueil
        req.session.loggedin = true;
        req.session.email = email;
        req.session.typeSal = data[0].typeSal;
        req.session.numSal = data[0].numMatr;
        res.redirect('/');
      } else {
        // En cas d'échec d'authentification, afficher un message d'erreur
        res.render('login',{messageErr : "le mot de passe ou l'email n'est pas
correcte"});
      }
      res.end();
    });
  } else {
    res.render('login',{messageErr : "l'utilisateur n'existe pas"});
    res.end();
  }
});
} else {
  // Si l'email ou le mot de passe n'ont pas été fournis, afficher un message d'erreur
  res.render('login',{messageErr : "veuillez entrer un email et un mot de passe"});
  res.end();
}
});

// Exporter l'objet router pour le rendre disponible pour une utilisation dans d'autres
fichiers
module.exports = router;
```

			numMatr	nomSalarie	prenomSalarie	adresseSalarie	dateEmbauche	mdp	typeSal	email
<input type="checkbox"/>	Éditer	Copier	Supprimer	S0001	Pitt	Brad	32 rue du Chemin Templeuve 59342	2004-12-03	\$2b\$10\$/q\$O9wQCqGVxunFIS3zOKYJlIbPLtIFUuAGPlyH1....	1 bradpitt@cc.com
<input type="checkbox"/>	Éditer	Copier	Supprimer	S0002	Looses	Erwani	24 rue de la Voie Roubaix 59424	2015-08-24	\$2b\$10\$/x\$uONVo1FQu0qqgMaFaGOqEGWUxANUXAnH.hL8qGK...	0 erwaniloses@ag1.fr
<input type="checkbox"/>	Éditer	Copier	Supprimer	S0003	Smith	John	45 rue de la Paix Roubaix 59402	2007-09-20	\$2b\$10\$/k/q\$O9wQCqGVxunFIS3zOKYJlIbPLtIFUuAGPlyH1....	1 johnsmith@cc.com
<input type="checkbox"/>	Éditer	Copier	Supprimer	S0004	Davis	Emily	67 avenue du Soleil Lille 59002	2018-03-12	\$2b\$10\$/x\$uONVo1FQu0qqgMaFaGOqEGWUxANUXAnH.hL8qGK...	0 davisemily@ag2.fr
<input type="checkbox"/>	Éditer	Copier	Supprimer	S0005	Garcia	Maria	10 Place de la Liberté Lille 59000	2019-05-08	\$2b\$10\$/x\$uONVo1FQu0qqgMaFaGOqEGWUxANUXAnH.hL8qGK...	0 mariagarcia@ag2.fr
<input type="checkbox"/>	Éditer	Copier	Supprimer	S0006	Chen	Wei	55 Boulevard des Arts Roubaix 59410	2016-11-30	\$2b\$10\$/k/q\$O9wQCqGVxunFIS3zOKYJlIbPLtIFUuAGPlyH1....	1 weichen@cc.com
<input type="checkbox"/>	Éditer	Copier	Supprimer	S0007	Kim	Ji-hoon	18 Rue de la République Lille 59000	2020-02-17	\$2b\$10\$/x\$uONVo1FQu0qqgMaFaGOqEGWUxANUXAnH.hL8qGK...	0 jihoonkim@ag3.fr

CashCash

Connexion

le mot de passe ou l'email n'est pas correcte

CashCash

Connexion


l'utilisateur n'existe pas

```
router.get('/logout', (req, res) => {
  if (req.session) {
    req.session.destroy(err => {
      if (err) {
        res.status(400).send('Unable to log out')
      } else {
        res.redirect('/login');
      }
    });
  } else {
    res.end()
  }
})
```

2) Les fonctionnalités pour l’assistant téléphonique


CashCash

bradpitt@cc.com




Affecter une visite
Ajouter une visite à un technicien

Cliquer




Recherche fiche
Rechercher une fiche client

Cliquer



Créer intervention
Créer une intervention

Cliquer



Recherche d'intervention
Par date ou agent

Cliquer

Déconnexion

a- Affecter une visite à un technicien

→ ↺ localhost:3000/affecterVisite 80 % ☆

CashCash

Affecter visite

Numéro intervention	Date intervention	Heure intervention	Numéro client
I0072	12/03/2020	10:10:10	CL0002
I0073	12/12/2020	10:10:10	CL0002
I0074	12/01/2020	10:10:10	CL0002
I0076	10/02/2023	10:00:00	CL0002
I0077	13/03/2024	13:00:00	CL0004
I0078	13/03/2024	13:00:00	CL0004
I0079	30/11/2023	19:00:00	CL0004
I0081	12/03/2024	10:00:00	CL0002
I0082	12/03/2024	12:00:00	CL0004

affecterVisite.js

```
// Ajout des modules nécessaires
var express = require('express');
var router = express.Router();
const session = require('express-session');
const db = require('../connect'); // Contient la logique de connexion à la base de données

// Défini une route pour gérer les requêtes GET à la racine '/'
router.get('/', function(req, res, next) {
  // Vérifier si l'utilisateur est connecté et a les variables de session requises
  if (req.session.email && req.session.typeSal) {
    // Vérifier si l'utilisateur a le rôle requis (typeSal)
    if (req.session.typeSal === 1) {

      // Afficher les interventions sans Techniciens
      const sql = "SELECT * FROM intervention WHERE numMatr IS NULL";

      db.query(sql, (err, data) => {
        // Gérer les éventuelles erreurs survenues lors de la requête à la base de données
        if (err) {
          throw err;
        }

        // Rendre la vue 'affecterVisite' et transmettre les données récupérées
        res.render('affecterVisite', { interventionData: data });

      });
    } else {
      // Si l'utilisateur n'a pas le rôle correct, le rediriger vers la racine '/'
      res.redirect("/");
    }
  } else {
    // Si l'utilisateur n'est pas connecté ou n'a pas les variables de session requises, le
    // rediriger vers la racine '/'
    res.redirect("/");
  }
});

// Exporter l'objet router pour le rendre disponible pour une utilisation dans d'autres
// fichiers
module.exports = router;
```


affecterVisite.pug

```
extends layout

block content
  h1 Affecter visite
  if interventionData
    table(class="table table-sm table-dark")
      thead
        tr
          th(scope="col") Numéro intervention
          th(scope="col") Date intervention
          th(scope="col") Heure intervention
          th(scope="col") Numéro client
          th(scope="col")
      for item in interventionData
        tr
          td(scope="row")= item.numIntervention
          td(scope="row")
            script.
              function formatDate(dateString) {
                var options = { day: '2-digit', month: '2-digit', year: 'numeric' };
                var date = new Date(dateString);
                return date.toLocaleDateString('fr-FR', options);
              }
              var formattedDate = formatDate("#{item.dateIntervention}");
              document.write(formattedDate);
          td(scope="row")= item.heureIntervention
          td(scope="row")= item.numClient
          td(scope="row")
            a(href="/ajouterTech?idInter="+item.numIntervention+"&idCli="+item.numClient)
Affecter visite avec un technicien

if messageErr
  div(class="alert alert-danger" role="alert")= messageErr
  br
  a(href="/ajouterTech")
  a(href="..") retour au menu
```

→ ↻ localhost:3000/ajouterTech?idInter=I0072&idCli=CL0002

CashCash

I0072

Numéro Technicien	
S0004	<button>Affecter la visite avec ce technicien</button>
S0005	<button>Affecter la visite avec ce technicien</button>

ajouterTech.js

```
// Ajout des modules nécessaires
const express = require('express');
const router = express.Router();
const db = require('../connect');
const session = require('express-session');

// Gérer les requêtes GET à la racine '/'
router.get('/', (req, res, next) => {

    //console.log(req.session);

    // Vérifier si l'utilisateur est connecté, a la variable de session 'typeSal' et a le rôle (typeSal)
    // égal à 1
    if (req.session.email && req.session.typeSal && req.session.typeSal === 1) {
        // Extraire les identifiants du client et de l'intervention des paramètres de requête
        const idCli = req.query.idCli;
        const idInter = req.query.idInter;

        // Requête SQL pour sélectionner les techniciens disponibles pour l'intervention associée à un
        // client
        const sql = "SELECT DISTINCT technicien.* FROM technicien, client, intervention WHERE
        technicien.numAgence = client.numAgence AND client.numClient = intervention.numClient AND
        intervention.numClient = ?";

        // Exécuter la requête dans la base de données
        db.query(sql, idCli, (err, data) => {
            if (err) {
                throw err;
            }

            //console.log(data);

            // Rendre la vue 'ajouterTech' avec les données récupérées
            res.render('ajouterTech', { techData: data, idCli: idCli, idInter: idInter, numMatr:
data[0].numMatr });
        });
    } else {
        // Si l'utilisateur n'a pas les autorisations nécessaires, le rediriger vers la racine '/'
        res.redirect("/");
    }
});

// Gérer les requêtes POST à la racine '/'
router.post('/', (req, res, next) => {
    // Extraire les données à mettre à jour à partir du corps de la requête
    const updateData = [
        req.body.numTech,
        req.body.numInter
    ];

    // Requête SQL pour mettre à jour le technicien associé à une intervention
    const query = "UPDATE intervention SET numMatr = ? WHERE numIntervention = ?";

    // Exécuter la requête dans la base de données
    db.query(query, updateData, (err, data) => {
        if (err) {
            // En cas d'erreur, rendre la vue 'ajouterTech' avec un message d'erreur
            res.render('ajouterTech', { messageErr: "Erreur" });
        } else {
            // En cas de succès, rendre la vue 'ajouterTech' avec un message de succès et le numéro
            // d'intervention
            res.render('ajouterTech', { messageSucc: "Modification réussie !", numInter: req.body.numInter
});
        }
    });
});

// Exporter l'objet router pour le rendre disponible pour une utilisation dans d'autres fichiers
module.exports = router;
```

ajouterTech.pug

```
extends layout

block content
  if idInter
    h1= idInter
  if techData
    table(class="table table-sm table-dark")
      thead
        tr
          th(scope="col") Numéro Technicien
          th(scope="col")
      for item in techData
        tr
          td(scope="row")= item.numMatr
          td(scope="row")
            form(action="" method="post")
              input(type="hidden" value=item.numMatr name="numTech")
              input(type="hidden" value=idInter name="numInter")
              input(type="submit", value="Affecter la visite avec ce
technicien" class="btn btn-outline-primary btn-sm" id="confirm")

  if messageSucc
    p &nbsp;
    div(class="alert alert-success" role="alert")= messageSucc

    p &nbsp;
    a(href="/genererPDF?numInter="+numInter class="btn btn-outline-primary btn-sm"
style="width:fit-content") Générer un pdf

    p &nbsp;
    a(href=`ajouterTech/annulerTech/${numInter}`, class="btn btn-outline-danger btn-sm"
) Annuler l'affectation
    a(href="..") retour au menu
```

[→](#) [↺](#) [🛡️](#) [📄](#) localhost:3000/ajouterTech?idInter=I0076&idCli=CL0002

CashCash

Modification réussie !

Générer un pdf

Annuler l'affectation

[retour au menu](#)

Annuler l'affectation

ajouterTech.js

```
// Gérer les requêtes GET pour annuler l'association d'un technicien à une intervention
router.get('/annulerTech/:numInter', (req, res, next) => {
  // Vérifier si l'utilisateur est connecté, a la variable de session 'typeSal' et a le
  rôle (typeSal) égal à 1
  if (req.session.email && req.session.typeSal && req.session.typeSal === 1) {
    // Extraire le numéro d'intervention des paramètres de requête
    const idInter = req.params.numInter;

    // Requête SQL pour annuler l'association d'un technicien à une intervention
    const query = "UPDATE intervention SET numMatr = NULL WHERE numIntervention = ?;";

    // Exécuter la requête dans la base de données
    db.query(query, [idInter], (err, data) => {
      // Afficher les données récupérées dans la console à des fins de débogage
      console.log(data);
      console.log(req.params);

      if (err) {
        throw err;
      } else {
        // Rediriger vers la page 'affecterVisite' après l'annulation de l'association
        res.redirect("/affecterVisite");
      }
    });
  } else {
    // Si l'utilisateur n'a pas les autorisations nécessaires, le rediriger vers la racine
    res.redirect("/");
  }
});
```

Générer un pdf

genererPdf.js

```
// Ajout des modules nécessaires
var express = require('express');
var router = express.Router();
const db = require('../connect');
const session = require('express-session');
const puppeteer = require('puppeteer');

// Fonction pour récupérer les données d'une intervention en fonction de son numéro
const fetchInterventionData = (numInter) => {
  const interventionQuery = "SELECT * FROM intervention WHERE numIntervention = ?";
  return new Promise((resolve, reject) => {
    db.query(interventionQuery, [numInter], (err, data) => {
      err ? reject(err) : resolve(data);
    });
  });
};

// Fonction pour récupérer les numéros de série des matériels associés à une intervention
const fetchReparerData = (numInter) => {
  const reparerQuery = "SELECT numSerie FROM reparer WHERE numIntervention = ?";
  return new Promise((resolve, reject) => {
    db.query(reparerQuery, [numInter], (err, data) => {
      err ? reject(err) : resolve(data);
    });
  });
};

// Gérer les requêtes GET à la racine '/'
router.get('/', async (req, res) => {
  // Vérifier si l'utilisateur est connecté, a la variable de session 'typeSal' et a le rôle (typeSal) égal à 1
  if (req.session.email && req.session.typeSal && req.session.typeSal === 1) {
    // Extraire le numéro d'intervention des paramètres de requête
    const numInter = req.query.numInter;

    try {
      // Récupérer les données de l'intervention en utilisant la fonction
      fetchInterventionData
      const interventionData = await fetchInterventionData(numInter);

      // Récupérer les numéros de série des matériels associés à l'intervention en
      utilisant la fonction fetchReparerData
      const reparerData = await fetchReparerData(numInter);

      //console.log(interventionData);
      // console.log(reparerData);

      // Lancer le navigateur Puppeteer
      const browser = await puppeteer.launch();
      const page = await browser.newPage();

      // Utiliser les données pour générer le contenu PDF
      const pdfContent = `
        <style>
          body {
            font-family: 'Helvetica Neue', Helvetica, Arial, sans-serif;
            line-height: 1.6;
            font-size: 14px;
            color: #333;
            margin: 0;
            padding: 0;
          }
          p {

```

```

        margin: 0 0 10px;
    }

    h2 {
        font-size: 24px;
        margin-bottom: 20px;
    }

    ul {
        padding-left: 20px;
        list-style-type: disc;
    }

    li {
        margin-bottom: 5px;
    }
</style>
<p>Numéro technicien: ${interventionData[0].numMatr}</p>
<p>Date intervention: ${interventionData[0].dateIntervention}</p>
<p>Heure intervention: ${interventionData[0].heureIntervention}</p>
<p>Numéro client: ${interventionData[0].numClient}</p>
<p>Numéro intervention: ${interventionData[0].numIntervention}</p>

<h2>Matériaux à réparer:</h2>
<ul>
    ${reparerData.map(item => `<li>${item.numSerie}</li>`).join('')}
</ul>
`;

// Définir le contenu de la page PDF et générer le PDF
await page.setContent(pdfContent);
await page.pdf({ path: 'intervention' + interventionData[0].numIntervention +
'.pdf', format: 'A4' });

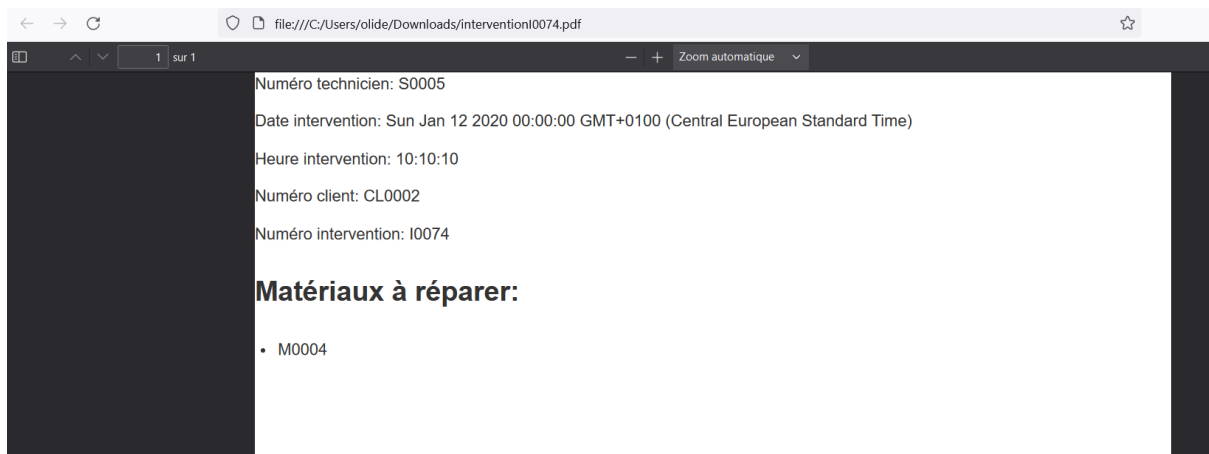
// Fermer le navigateur Puppeteer
await browser.close();

// Télécharger le PDF généré
res.download('intervention' + interventionData[0].numIntervention + '.pdf');
} catch (error) {
    // En cas d'erreur, afficher l'erreur dans la console et renvoyer une réponse
    // avec un statut d'erreur
    console.error(error);
    res.status(500).send('Error generating PDF');
}
} else {
    // Si l'utilisateur n'a pas les autorisations nécessaires, le rediriger vers la
    // racine '/'
    res.redirect("/");
}
});

// Exporter l'objet router pour le rendre disponible pour une utilisation dans d'autres
// fichiers
module.exports = router;

```

Résultat:

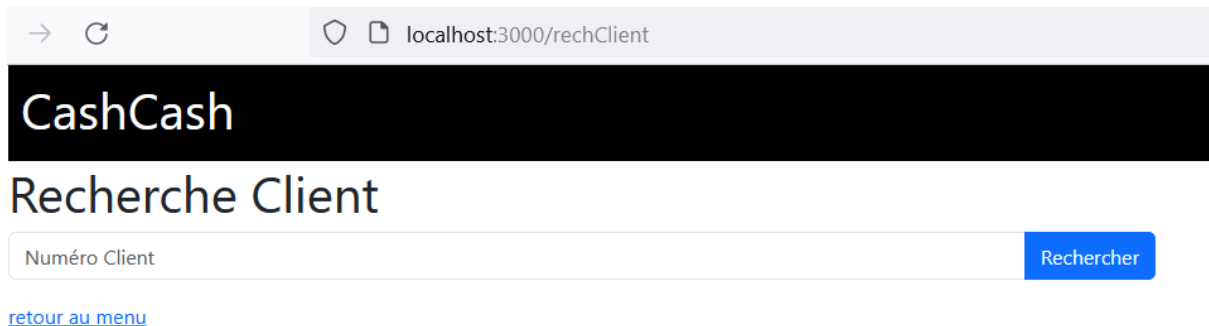


Numéro technicien: S0005
Date intervention: Sun Jan 12 2020 00:00:00 GMT+0100 (Central European Standard Time)
Heure intervention: 10:10:10
Numéro client: CL0002
Numéro intervention: I0074

Matériaux à réparer:

- M0004

b- Recherche d'un client

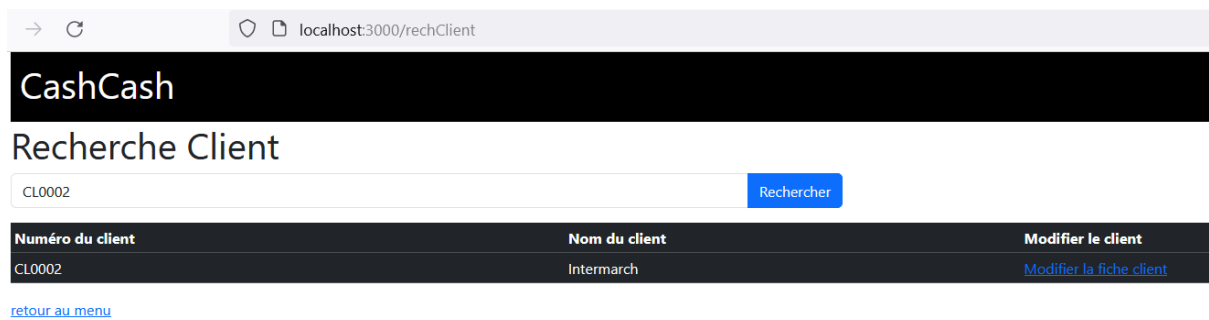


CashCash

Recherche Client

Numéro Client

[retour au menu](#)



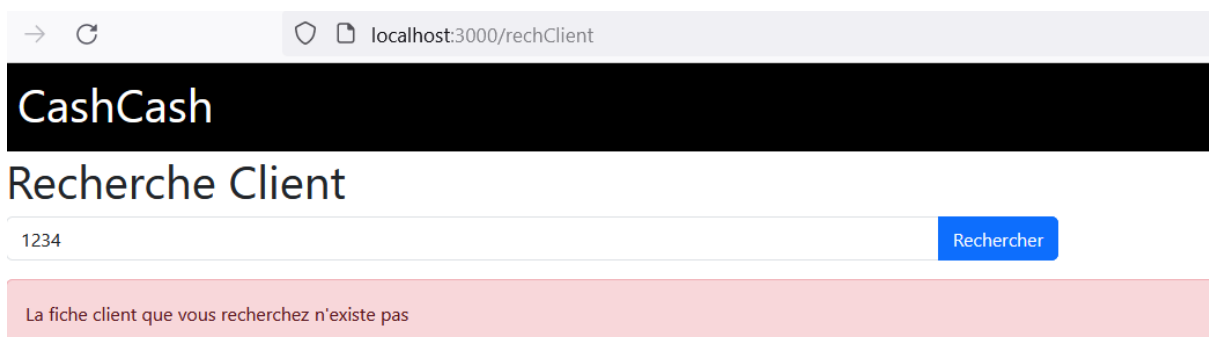
CashCash

Recherche Client

CL0002

Numéro du client	Nom du client	Modifier le client
CL0002	Intermarch	Modifier la fiche client

[retour au menu](#)



CashCash

Recherche Client

1234

La fiche client que vous recherchez n'existe pas

rechClient.js

```
// Ajout des modules nécessaires
var express = require('express');
var router = express.Router();
const db = require('../connect');
const session = require('express-session');

// Gérer les requêtes GET à la racine '/'
router.get('/', function(req, res, next) {
  // Vérifier si l'utilisateur est connecté et a la variable de session 'typeSal'
  if (req.session.email && req.session.typeSal) {
    // Vérifier si le typeSal de l'utilisateur est égal à 1 (1 = assistant téléphonique)
    if(req.session.typeSal == 1){
      // Rendre la vue 'rechClient' pour permettre à l'utilisateur de rechercher des
clients
      res.render('rechClient');
    } else {
      // Rediriger vers la racine '/' si l'utilisateur n'a pas les autorisations
nécessaires
      res.redirect("/");
    }
  } else {
    // Rediriger vers la racine '/' si l'utilisateur n'est pas connecté
    console.log("utilisateur non connecté")
    res.redirect("/");
  }
});

// Gérer les requêtes POST à la racine '/'
router.post('/', (req, res) => {
  // Extraire le numéro du client à rechercher à partir du formulaire
  const numCli = req.body.numCli;
  // Requête SQL pour récupérer les informations du client en fonction de son numéro
  const sql = "SELECT * FROM client WHERE client.numClient = ?";

  db.query(sql, numCli, (err, data) => {
    if (err) throw err;
    console.log(data);
    // Vérifier si des données ont été trouvées pour le numéro de client donné
    if (data.length == 0){
      // Afficher un message si la fiche client recherchée n'existe pas
      res.render('rechClient', { message : "La fiche client que vous recherchez n'existe
pas" });
    } else {
      // Afficher les données du client trouvé
      res.render('rechClient', { cliData: data });
    }
  });
});

// Exporter l'objet router pour le rendre disponible pour une utilisation dans d'autres
fichiers
module.exports = router;
```


rechClient.pug

```
extends layout

block content
  h1 Recherche Client

  form(action="", method="post")
    div(class="input-group mb-3" style="width:50%")
      input(type="text" class="form-control" placeholder="Numéro Client" aria-label="Fiche client" aria-describedby="basic-addon2" name="numCli")
      button(type="submit" class="btn btn-primary") Rechercher

  if cliData
    table(class="table table-sm table-dark")
      thead
        tr
          th(scope="col") Numéro du client
          th(scope="col") Nom du client
          th(scope="col") Modifier le client
      for item in cliData
        tr
          td(scope="row")= item.numClient
          td(scope="row")= item.nomClient
          td(scope="row")
            a(href="/modifClient?id="+item.numClient) Modifier la fiche client
      else
        if message
          div(class="alert alert-danger" role="alert")= message
    a(href="..") retour au menu
```

→ ↻ localhost:3000/modifClient?id=CL0002

CashCash

Intermarch	test
123123121	12345
45 rue de la voie Verte Lille	0101010102
0101010102	intermarché59@gmail.com
8.50	01:20:46
A0002	Modifier les données

[retour au recherche client](#)
[retour au menu](#)

modifClient.pug

```
extends layout

block content
  h1= id

  if userData
    form(action="", method="post")
      table
        for item in userData
          tr
            td
              input(type="text" value=item.nomClient placeholder="Nom du client" name="nom")
            td
              input(type="text" value=item.raisonSociale placeholder="Raison sociale" name="raisonSociale")
          tr
            td
              input(type="text" value=item.SIREN placeholder="SIREN" name="siren")
            td
              input(type="text" value=item.APE placeholder="APE" name="ape")
          tr
            td
              input(type="text" value=item.adresseClient placeholder="Adresse du client" name="adresse")
            td
              input(type="text" value=item.telephoneClient placeholder="Numéro de téléphone" name="telephone")
          tr
            td
              input(type="text" value=item.telecopieClient placeholder="Numéro de télécopie" name="telecopie")
            td
              input(type="text" value=item.emailClient placeholder="Email" name="email")
          tr
            td
              input(type="text" value=item.distanceKilo placeholder="Distance Kilométrique" name="distanceKilo")
            td
              input(type="text" value=item.tempsDeplacement placeholder="Temps de déplacement" name="tempsDeplacement")
          tr
            td
              input(type="text" value=item.numAgence placeholder="Numéro de l'agence" name="numeroAgence")
            td
              button(type="submit" class="wider btn btn-primary btn-sm")
                Modifier les données

  if messageSucc
    div(class="alert alert-success" role="alert")= messageSucc
    a(href="/modifClient?id="+id) vous avez fait un erreur? retour à la page de modification

  if messageErr
    div(class="alert alert-danger" role="alert")= messageErr
    a(href="/modifClient?id="+id) retourner à la page de modification
  br
  a(href="rechClient") retour au recherche client
  br
  a(href="..") retour au menu
```

modifClient.js

```
// Ajout des modules nécessaires
var express = require('express');
var router = express.Router();
const db = require('../connect');
const session = require('express-session');

// REGEX pour les validations
var numericPattern = /^[0-9]+$/;
var emailPattern = /^((([^\<>()[]\@\\.,;:\s@"]+)(\.[^\<>()[]\@\\.,;:\s@"]+)*)|(\\".+\"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\]|(([a-zA-Z-0-9]+\.)+[a-zA-Z]{2,})))$/;
var decimalPattern = /^\\d{1,3}(\\.\\d{1,2})?$/;
var timePattern = /^(0[0-9]|1[0-9]|2[0-3]):[0-5][0-9]:[0-5][0-9]$/;

// Gérer les requêtes GET à la racine '/'
router.get('/', function(req, res, next) {
  // Vérifier si l'utilisateur est connecté, a la variable de session 'typeSal' et a le rôle (typeSal) égal à 1
  if (req.session.email && req.session.typeSal && req.session.typeSal === 1) {
    // Extraire l'identifiant du client des paramètres de requête
    const id = req.query.id;
    // Requête SQL pour récupérer les données du client en fonction de son identifiant
    const sql = "SELECT * FROM client WHERE client.numClient = ?";
    db.query(sql, id, (err, data) => {
      if (err) throw err;
      console.log(data);
      // Rendre la vue 'modifClient' en passant les données du client en tant que
      // variables
      res.render('modifClient', { userData: data });
    });
  } else {
    // Rediriger vers la racine '/' si l'utilisateur n'a pas les autorisations
    // nécessaires
    res.redirect("/");
  }
});

// Gérer les requêtes POST à la racine '/'
router.post('/', (req, res) => {
  // Validation des champs du formulaire avec des expressions régulières

  if (req.body.raisonSociale.length > 30) {
    res.render('modifClient', { messageErr: "Trop long!", id: req.query.id });
  } else if ((numericPattern.test(req.body.siren) == false) || (req.body.siren.length != 9)) {
    res.render('modifClient', { messageErr: "SIREN incorrecte", id: req.query.id });
  } else if ((numericPattern.test(req.body.ape) == false) || (req.body.ape.length != 5)) {
    res.render('modifClient', { messageErr: "APE incorrecte", id: req.query.id });
  } else if ((numericPattern.test(req.body.telephone) == false) || (req.body.telephone.length != 10)) {
    res.render('modifClient', { messageErr: "Téléphone incorrecte", id: req.query.id });
  } else if ((numericPattern.test(req.body.telecopie) == false) || (req.body.telecopie.length != 10)) {
    res.render('modifClient', { messageErr: "Télécopie incorrecte", id: req.query.id });
  } else if ((emailPattern.test(req.body.email) == false) || (req.body.email.length > 30)) {
    res.render('modifClient', { messageErr: "Email incorrecte", id: req.query.id });
  } else if (req.body.adresse.length > 50) {
    res.render('modifClient', { messageErr: "Adresse incorrecte", id: req.query.id });
  } else if (req.body.nom.length > 10) {
    res.render('modifClient', { messageErr: "Nom trop long", id: req.query.id });
  } else if (decimalPattern.test(req.body.distanceKilo) == false) {
    res.render('modifClient', { messageErr: "Distance kilométrique incorrecte", id: req.query.id });
  }
});
```

```

    } else if (timePattern.test(req.body.tempsDeplacement) == false) {
        res.render('modifClient', { messageErr: "Temps déplacement incorrecte", id:
req.query.id })
    } else if (req.body.numeroAgence.length > 5) {
        res.render('modifClient', { messageErr: "Numero agence non valide", id:
req.query.id })
    } else {
        // Si les champs sont valides, exécuter la mise à jour des données du client dans
la base de données
        const query = "UPDATE client SET raisonSociale = ?, SIREN = ?, APE = ?,
adresseClient = ?, telephoneClient = ?, telecopieClient = ?, emailClient = ?, nomClient =
?, distanceKilo = ?, tempsDeplacement = ?, numAgence = ? WHERE numClient = ?";

        const updateData = [
            req.body.raisonSociale,
            req.body.siren,
            req.body.appe,
            req.body.adresse,
            req.body.telephone,
            req.body.telecopie,
            req.body.email,
            req.body.nom,
            req.body.distanceKilo,
            req.body.tempsDeplacement,
            req.body.numeroAgence,
            req.query.id
        ];

        db.query(query, updateData, (err, data) => {
            // En cas d'erreur, afficher un message d'erreur spécifique
            if (err) {
                res.render('modifClient', { messageErr: "Le numéro d'agence n'existe pas",
id: req.query.id })
            } else {
                // Si la mise à jour réussit, afficher un message de succès
                res.render('modifClient', { messageSucc: "Modification réussie!", id:
req.query.id });
            }
        });
    }
});
});
module.exports = router;

```

→ ↺ localhost:3000/modifClient?id=CL0002

CashCash

Intermarch	test
123123121	1234
45 rue de la voie Verte Lille	0101010102
0101010102	intermarché59@gmail.com
8.50	01:20:46
A0002	Modifier les données

[retour au recherche client](#)

[retour au menu](#)

→ ↺

localhost:3000/modifClient?id=CL0002

CashCash

CL0002

APE incorrecte

[retourner à la page de modification](#)
[retour au recherche client](#)
[retour au menu](#)

*

→ ↺

localhost:3000/modifClient?id=CL0002

CashCash

Intermarch	test
123123121	12347
45 rue de la voie Verte Lille	0101010102
0101010102	intermarché59@gmail.com
8.50	01:20:46
A0002	Modifier les données

[retour au recherche client](#)
[retour au menu](#)

→ ↺

localhost:3000/modifClient?id=CL0002

CashCash

CL0002

Modification réussie!

[vous avez fait un erreur? retour à la page de modification](#)
[retour au recherche client](#)
[retour au menu](#)

c- Création d'une intervention

→ ↻ localhost:3000/creerInter

CashCash

Créer une intervention

Date intervention:

Heure intervention:

Numéro client

[retour au menu](#)

creerInter.pug

```
extends layout

block content
  h1 Créer une intervention

  form(action="", method="post")
    table
      tr
        td
          label Date intervention:
          input(type="date" placeholder="Date intervention" name="dateInter"
required)
      tr
        td
          label Heure intervention:
          input(type="time" placeholder="Heure intervention" step="1"
name="heureInter" required)
      tr
        td
          input(type="text" placeholder="Numéro client" name="numCli" required)
      tr
        td
          button(type="submit" class="wider btn btn-primary btn-sm") Créer une
intervention

    if messageSucc
      div(class="alert alert-success" role="alert")= messageSucc
      a(href="/ajouterMat?numCli=${numCli}&numInter=${numInter}" class="btn btn-secondary
btn-sm") Ajouter le matériel à réparer

    if messageErr
      div(class="alert alert-danger" role="alert")= messageErr

    br
    a(href="..") retour au menu
```

creerInter.js

```
// Ajouts des modules nécessaires
var express = require('express');
var router = express.Router();
const db = require('../connect');
const session = require('express-session');

// Gérer les requêtes GET à la racine '/'
router.get('/', function(req, res, next) {
  // Vérifier si l'utilisateur est connecté, a la variable de session 'typeSal' et a le rôle (typeSal) égal à 1
  if (req.session.email && req.session.typeSal && req.session.typeSal === 1) {
    // Rendre la vue 'creerInter' si les conditions sont remplies
    res.render('creerInter');
  } else {
    // Rediriger vers la racine '/' si les conditions ne sont pas remplies
    res.redirect("/");
  }
});

// Gérer les requêtes POST à la racine '/'
router.post('/', (req, res) => {
  // Requête SQL pour insérer une nouvelle intervention dans la base de données
  const insertQuery = "INSERT INTO intervention (numIntervention, dateIntervention, heureIntervention, numClient, numMatr) SELECT CONCAT('I', LPAD(COALESCE((SELECT COUNT(numIntervention) + 1 FROM intervention), 1), 4, '0')),?, ?, ?, NULL;";
  // Requête SQL pour récupérer la dernière intervention insérée
  const selectQuery = "SELECT * FROM intervention ORDER BY numIntervention DESC LIMIT 1;";
  // Données à insérer dans la requête d'insertion
  const insertData = [
    req.body.dateInter,
    req.body.heureInter,
    req.body.numCli
  ];

  // Exécuter la requête d'insertion dans la base de données
  db.query(insertQuery, insertData, (err, result) => {
    if (err) {
      // En cas d'erreur, rendre la vue 'creerInter' avec un message d'erreur
      res.render('creerInter', { messageErr: "Le numéro de client n'existe pas" });
    } else {
      // Si l'insertion est réussie, récupérer la dernière intervention insérée
      db.query(selectQuery, (err, results) => {
        if (err) {
          throw err;
        } else {
          // Rediriger vers la page 'ajouterMat' avec les paramètres nécessaires
          res.redirect("/ajouterMat?numCli=" + req.body.numCli + "&numInter=" + results[0].numIntervention);
        }
      });
    }
  });
});

module.exports = router;
```

→ ↻ localhost:3000/ajouterMat?numCli=CL0002&numInter=I0075

CashCash

CL0002

☐ M0004

Ajouter les matériaux

Annuler l'intervention

ajouterMat.pug

```
extends layout

block content
  if numCli
    h2= numCli
  if matData
    table
      form(action="", method="post")
        for item in matData
          tr
            td
              input(type="checkbox", name=item.numSerie)
              label(for=item.numSerie)= item.numSerie
          tr
            td
              input(type="submit", value="Ajouter les matériaux", class="btn btn-outline-primary btn-sm", id="confirm")
        if messageSucc
          p &nbsp;
          div(class="alert alert-success" role="alert")= messageSucc

          p &nbsp;
          a(href="/ajouterMat/cancel/${numInter}" class="btn btn-outline-danger btn-sm" style="width:fit-content") Annuler l'intervention

        if messageErr
          div(class="alert alert-danger" role="alert")= messageErr

        if messageSucc
          br

          a(href="..") retour au menu
```


ajouterMat.js

```
// Ajout des modules nécessaires
var express = require('express');
var router = express.Router();
const db = require('../connect');
const session = require('express-session');

// Gérer les requêtes GET à la racine '/'
router.get('/', async (req, res) => {
  // Vérifier si l'utilisateur est connecté, a la variable de session 'typeSal' et a le
  rôle (typeSal) égal à 1
  if (req.session.email && req.session.typeSal && req.session.typeSal === 1) {
    // Extraire les numéros de client et d'intervention à partir des paramètres de
    requête
    const numCli = req.query.numCli;
    const numInter = req.query.numInter;

    // Requête SELECT pour récupérer les données des matériaux en fonction du client et
    du contrat non nul
    const selectQuery = "SELECT * FROM materiel WHERE numClient = ? and numContrat is
    not null;";

    // Exécuter la requête dans la base de données
    db.query(selectQuery, numCli, (err, data) => {
      if (err) {
        throw err;
      }

      // Si aucune donnée n'est récupérée, rediriger vers une page d'annulation,
      sinon, rendre la vue 'ajouterMat'
      if (data.length === 0) {
        res.redirect(`ajouterMat/cancel/${numInter}`);
      } else {
        res.render('ajouterMat', { matData: data, numInter: numInter, numCli: numCli
      });
    }
  }
});

} else {
  // Si l'utilisateur n'a pas les autorisations nécessaires, le rediriger vers la
  racine '/'
  res.redirect("/");
}
});

// Gérer les requêtes POST à la racine '/'
router.post('/', (req, res) => {
  // Extraire les matériaux sélectionnés à partir du corps de la requête
  const selectedMaterials = Object.keys(req.body);
  const numCli = req.query.numCli;
  const numInter = req.query.numInter;

  // Si aucun matériau n'est sélectionné, récupérer les données du matériel et rediriger
  avec un message d'erreur
  if (selectedMaterials.length === 0) {
    const selectQuery = "SELECT * FROM materiel WHERE numClient = ? and numContrat is
    not null;";

    db.query(selectQuery, numCli, (err, data) => {
      if (err) {
        throw err;
      }

      console.log(data.length);
    });
  }
});
```

```

        if (data.length === 0) {
            res.redirect(`ajouterMat/cancel/${numInter}`);
        } else {
            res.render('ajouterMat', { matData: data, numInter: numInter, messageErr:
"Aucun matériau sélectionné" });
        }
    });

    } else {
        // Si des matériaux sont sélectionnés, insérer les données dans la table 'reparer'
        const insertQuery = "INSERT INTO reparer (numSerie, numIntervention) VALUES ?";

        // Mapper les matériaux sélectionnés pour les valeurs d'insertion
        const values = selectedMaterials.map(numSerie => [numSerie, numInter]);

        // Exécuter la requête d'insertion dans la base de données
        db.query(insertQuery, [values], (err, result) => {
            if (err) {
                throw err;
            }

            // Afficher le nombre de lignes insérées dans la console et rendre la vue avec
un message de succès
            //console.log(`Inserted ${result.affectedRows} rows into reparer table.`);
            res.render('ajouterMat', { messageSucc: "Ajout réussi !", numInter: numInter,
numCli: numCli });
        });
    }
});

// Gérer les requêtes GET pour annuler une opération
router.get('/cancel/:numInter', (req, res) => {
    // Vérifier si l'utilisateur est connecté, a la variable de session 'typeSal' et a le
rôle (typeSal) égal à 1
    if (req.session.email && req.session.typeSal && req.session.typeSal === 1) {
        // Extraire le numéro d'intervention des paramètres de requête
        const numInter = req.params.numInter;

        // Requête DELETE pour supprimer les entrées correspondant au numéro d'intervention
dans les tables 'reparer' et 'intervention'
        const deleteReparerQuery = "DELETE FROM reparer WHERE numIntervention = ?";
        db.query(deleteReparerQuery, [numInter], (errReparer, resultReparer) => {
            if (errReparer) {
                throw errReparer;
            }

            // Afficher le nombre de lignes supprimées dans la console

            console.log(`Deleted ${resultReparer.affectedRows} rows from reparer where
numIntervention = ${numInter}.`);

            // Requête DELETE pour supprimer l'intervention
            const deleteInterventionQuery = "DELETE FROM intervention WHERE numIntervention
= ?";
            db.query(deleteInterventionQuery, [numInter], (errIntervention,
resultIntervention) => {
                if (errIntervention) {
                    throw errIntervention;
                }

                // Afficher le nombre de lignes supprimées dans la console
                //console.log(`Deleted ${resultIntervention.affectedRows} row from
intervention where numIntervention = ${numInter}.`);

                // Rediriger vers la création d'une nouvelle intervention
                res.redirect("/creerInter");
            }
        });
    }
});

```

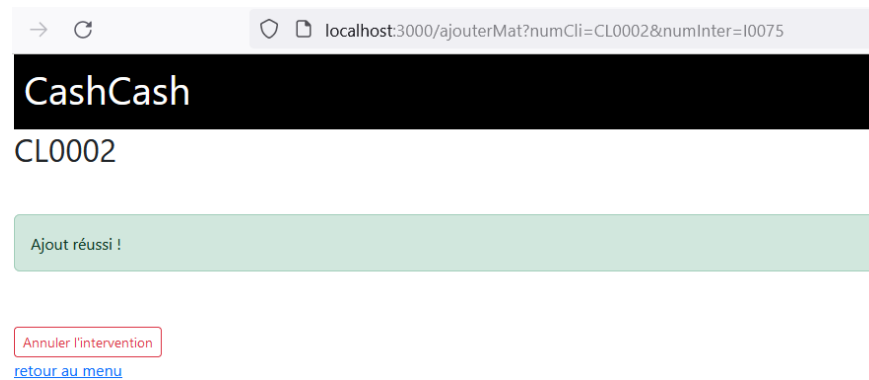
```

    });
  });
} else {

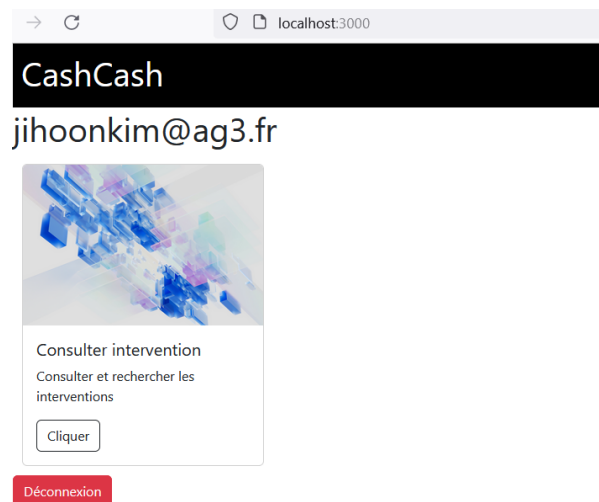
    // Si l'utilisateur n'a pas les autorisations nécessaires, le rediriger vers la
    racine '/'
    res.redirect("/");
  }
});

// Exporter l'objet router pour le rendre disponible pour une utilisation dans d'autres
fichiers
module.exports = router;

```



3) Les fonctionnalités pour le technicien



→ ↺ localhost:3000/consultInter 80 % ☆

CashCash

Consulter les interventions

Numéro de l'intervention	Date intervention	Heure intervention	Numéro du client	Temps déplacement	Distance Kilométrique	
I0059	Sat Oct 10 2020 00:00:00 GMT+0200 (Central European Summer Time)	10:10:10	CL0004	01:15:30	7.30	Valider l'intervention
I0055	Tue Dec 12 2023 00:00:00 GMT+0100 (Central European Standard Time)	10:00:00	CL0004	01:15:30	7.30	Valider l'intervention

[retour au menu](#)

consulterInter.pug

```
extends layout

block content
  h1 Consulter les interventions
  if interData
    table(class="table table-sm table-dark")
      thead
        tr
          th(scope="col") Numéro de l'intervention
          th(scope="col") Date intervention
          th(scope="col") Heure intervention
          th(scope="col") Numéro du client
          th(scope="col") Temps déplacement
          th(scope="col") Distance Kilométrique
          th(scope="col")
      for item in interData
        tr
          td(scope="row")= item.numIntervention
          td(scope="row")= item.dateIntervention
          td(scope="row")= item.heureIntervention
          td(scope="row")= item.numClient
          td(scope="row")= item.tempsDeplacement
          td(scope="row")= item.distanceKilo
          td(scope="row")
            a(href="/validerInter?numInter="+item.numIntervention) Valider
  l'intervention
  else
    if message
      div(class="alert alert-danger" role="alert")= message
    a(href="..") retour au menu
```

consulterInter.js

```
// Ajout des modules nécessaires
var express = require('express');
var router = express.Router();
const db = require('../connect');
const session = require('express-session');

// Gérer les requêtes GET à la racine '/'
router.get('/', function(req, res, next) {
  // Vérifier si l'utilisateur a une session avec l'adresse email enregistrée
  if (req.session.email) {
    // Vérifier si l'utilisateur a le rôle (typeSal) différent de 1
    if (req.session.typeSal == 0) {

      //console.log(req.session);

      // Requête SQL pour récupérer les interventions associées à un technicien spécifique
      const sql = "SELECT DISTINCT intervention.numIntervention,
intervention.dateIntervention, intervention.heureIntervention, intervention.numClient,
client.tempsDeplacement,client.distanceKilo FROM intervention, reparer, client WHERE
intervention.numMatr = ? AND intervention.numClient = client.numClient AND
intervention.numIntervention = reparer.numIntervention AND reparer.durreReparationMat is
null ORDER BY intervention.dateIntervention, intervention.heureIntervention,
client.distanceKilo ASC";

      // Exécuter la requête dans la base de données avec le numéro du technicien issu de
      la session
      db.query(sql, req.session.numSal, (err, data) => {
        if (err) throw err;

        //console.log(data);

        // Si aucune donnée n'est récupérée, rendre la vue 'consultInter' avec un message
        approprié, sinon, rendre la vue avec les données récupérées
        if (data.length == 0) {
          res.render('consultInter', { message: "Vous n'avez pas d'intervention" });
        } else {
          res.render('consultInter', { interData: data });
        }
      });
    } else {
      // Si l'utilisateur a le rôle (typeSal) égal à 1, le rediriger vers la racine '/'
      res.redirect("/");
    }
  } else {
    // Si l'utilisateur n'a pas de session ou n'a pas l'adresse email enregistrée, le
    rediriger vers la racine '/'
    res.redirect("/");
  }
});

// Exporter l'objet router pour le rendre disponible pour une utilisation dans d'autres
fichiers
module.exports = router;
```

→ ↻ localhost:3000/validerInter?numInter=10059 80 % ☆ 📄 📄 📄 📄

CashCash

Valider les interventions

Numéro de série	Durée de l'intervention	Commentaire	
M0006	<input type="text" value="--:--:--"/>	<input type="text" value="Commentaire"/>	<input type="button" value="Valider l'intervention"/>
M0007	<input type="text" value="--:--:--"/>	<input type="text" value="Commentaire"/>	<input type="button" value="Valider l'intervention"/>

[retour en arrière](#)
[retour au menu](#)

validerInter.pug

```
extends layout

block content
  h1 Valider les interventions
  if repareData
    table(class="table table-sm table-dark")
      thead
        tr
          th(scope="col") Numéro de série
          th(scope="col") Durée de l'intervention
          th(scope="col") Commentaire
          th(scope="col")
      for item in repareData
        tr
          form(action="", method="post")
            td(scope="row")= item.numSerie
            td(scope="row")
              input(type="time" step="1" name="dureeRep" required)
            td(scope="row")
              input(type="text" placeholder="Commentaire"
name="commentaire")
            td(scope="row")
              input(type="hidden" value= item.numSerie name="numSerie")
              input(type="hidden" value= item.numIntervention
name="numInter")
              button(type="submit" class="wider btn btn-primary btn-sm")
Valider l'intervention

    if message
      div(class="alert alert-success" role="alert")= message
      a(href=`/validerInter/annulerIntervention/${numMat}/${numInter}` class="btn btn-
outline-danger btn-sm" style="width:fit-content") Annuler l'intervention
    br
    a(href="/consultInter") retour en arrière
    br
    a(href="..") retour au menu
```

validerInter.js

```
// Ajout des modules nécessaires
var express = require('express');
var router = express.Router();
const db = require('../connect');
const session = require('express-session');

// Gérer les requêtes GET à la racine '/'
router.get('/', function(req, res, next) {
  // Vérifier si l'utilisateur est connecté
  if (req.session.email){
    // Vérifier si le typeSal de l'utilisateur n'est pas égal à 1 (1 = assistant téléphonique)
    if(req.session.typeSal !== 1){
      // Extraire le numéro d'intervention à partir de la requête
      const id = req.query.numInter;
      // Requête SQL pour récupérer les réparations associées à une intervention spécifique qui n'ont pas encore de durée de réparation
      const sql = "SELECT * FROM reparer WHERE reparer.numIntervention = ? and reparer.durreReparationMat is null";
      db.query(sql, id, (err, data) => {
        if (err) throw err;
        // Rendre la vue 'validerInter' avec les données des réparations à valider
        res.render('validerInter', {repareData: data});
      });
    } else {
      // Rediriger vers la racine '/' si l'utilisateur n'a pas les autorisations nécessaires
      res.redirect("/");
    }
  } else {
    // Rediriger vers la racine '/' si l'utilisateur n'est pas connecté
    res.redirect("/");
  }
});

// Gérer les requêtes POST à la racine '/'
router.post('/', (req, res) => {
  // Requête SQL pour mettre à jour la durée de réparation et le commentaire d'une réparation spécifique
  const query = "UPDATE reparer SET durreReparationMat = ?, commentaire = ? WHERE numSerie = ? AND numIntervention = ?";

  const updateData = [
    req.body.dureeRep,
    req.body.commentaire,
    req.body.numSerie,
    req.body.numInter
  ];

  db.query(query, updateData, (err, data) => {
    if (err){
      // Afficher une erreur si la mise à jour échoue
      throw err;
    } else {
      // Rendre la vue 'validerInter' avec un message de succès et les informations de l'intervention mise à jour
      res.render('validerInter', {message : "Modification réussie!", numInter : req.body.numInter, numMat : req.body.numSerie});
    }
  });
});

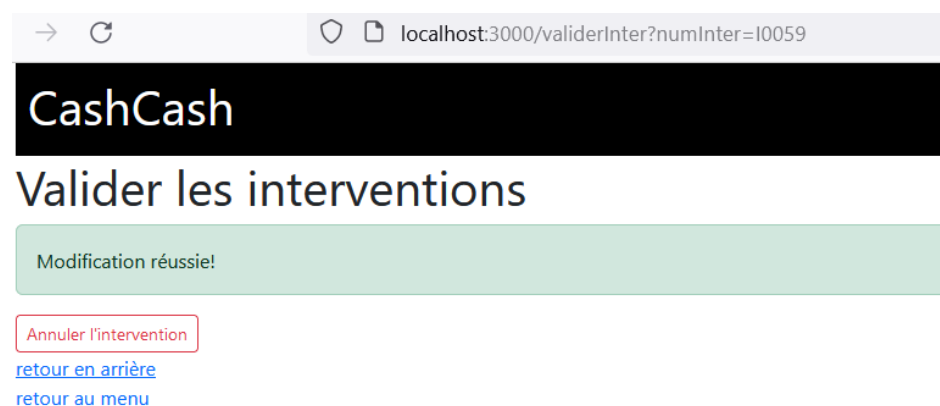
// Gérer les requêtes GET pour annuler une intervention
router.get('/annulerIntervention/:numSerie/:numIntervention', (req, res, next) => {
```

```

// Vérifier si l'utilisateur est connecté
if (req.session.email){
  // Vérifier si le typeSal de l'utilisateur n'est pas égal à 1 (1 = assistant
  téléphonique)
  if(req.session.typeSal !== 1){
    // Extraire les numéros de série et d'intervention à partir de la requête
    const numSerie = req.params.numSerie;
    const numIntervention = req.params.numIntervention;
    // Requête SQL pour annuler la durée de réparation et le commentaire d'une
    réparation spécifique
    const query = "UPDATE reparer SET durreReparationMat = NULL, commentaire = NULL
    WHERE numSerie = ? AND numIntervention = ?";
    db.query(query, [numSerie, numIntervention], (err, data) => {
      if (err) {
        // Afficher une erreur si la mise à jour échoue
        throw err;
      } else {
        // Rediriger vers la vue 'validerInter' avec le numéro de
        l'intervention annulée
        res.redirect("/validerInter?numInter="+numIntervention);
      }
    });
  } else {
    // Rediriger vers la racine '/' si l'utilisateur n'a pas les autorisations
    nécessaires
    res.redirect("/");
  }
} else {
  // Rediriger vers la racine '/' si l'utilisateur n'est pas connecté
  res.redirect("/");
}
});

// Exporter l'objet router pour le rendre disponible pour une utilisation dans d'autres
fichiers
module.exports = router;

```



Commentaires

Je suis satisfait du résultat du travail. Malgré le fait qu'il y eu des difficultés de la répartition et l'organisation du travail dans le groupe.