Documentsassociés au dossierA

DocumentA1 : Besoins de sécurité pour les récits utilisateurs (user stories) de la première itération de l'application gestion des billets et des visites sur le site Web (extraits)

	Intitulé du récit utilisateur (*user story)*	Disponibilité	Intégrité	Confidentialité	Preuve
1	En tant qu'acheteur, je veux acheter en ligne les billets pour plusieurs personnes afin de pouvoir participer à une visite.				
2	En tant qu'acheteur, je veux télécharger les billets d'une de mes réservations au format *PDF*afin de les transmettre aux personnes pour lesquelles j'ai réalisé la réservation.	**	**	*	-
5	En tant qu'acheteur, je veux consulter les caractéristiques des seniors pour lesquels j'ai acheté un billet sur un mois donné afin de réaliser une campagne publicitaire.	-	*	**	-
15	En tant que visiteur, je veux poster un commentaire afin de donner mon avis sur la qualité de la visite organisée.	*	**	-	*
22	En tant que visiteur, je peux créer, à l'issue de ma visite, un compte afin de retrouver sur le site *Web* les photos, vidéos et expériences effectuées sur le CDV.	*	**	**	*
25	En tant que responsable commercial, je veux consulter les statistiques de temps passé par zone de visite et les activités réalisées par les visiteurs afin de proposer un meilleur service aux visiteurs.				
27	En tant que visiteur, je veux être déconnecté du site lorsque je clique sur le bouton déconnexion afin de ne plus être identifié.	**	*	-	-
30	En tant que guide, je veux modifier mon mot de passe sur l'application mobile en toute sécurité afin de sécuriser mon compte.	*	**	**	*

- : pas de besoin ** : besoin important* *** : besoin très important*

Disponibilité : la fonctionnalité doit être utilisée au moment voulu.
Intégrité : les données doivent être exactes et complètes.
Confidentialité : les informations ne doivent pas être divulguées.
Preuve : les traces de l'activité du système sont opposables en cas de contestation.

Document A2 : Extrait de l'analyse des risques et menaces de l'environnement

Acteurs à l'origine de la malveillance

Acteurs malveillants	Modes opératoires	Probabilité
Attaquant externe (*hacker*)	L'attaquant externe accède à la base de données.	**
	L'attaquant externe surcharge le système.	***
Acheteur	L'acheteur envoie de fausses informations.	**
	L'acheteur surcharge le système.	*
Visiteur	Le visiteur envoie de fausses informations.	*
	Le visiteur surcharge le système.	*

** : faible probabilité* *** : forte* **** : très forte probabilité*

Impacts des événements redoutés

Numéro de l'événement	Événement	Impact pour l'entreprise	Gravité
1	Le système ne répond pas.		
2	Un attaquant accède à la base de données et ajoute des enregistrements dans la table billet.	Perte financière pour l'entreprise.	**
3	Un attaquant accède à la base de données et modifie l'affectation des guides aux visites.		
4	Un acheteur accède aux billets au format *PDF* d'un autre acheteur en modifiant le numéro de réservation dans la barre d'adresse.	Problème de confiance. Désorganisation des visites	*

** : modérée* *** : très élevée*

Scénarios de risques (*abuser story*) et mesures à prévoir

Numéro de l'événement	scénario de risque (*abuser story*)	Mesures à prévoir
2	En tant qu'attaquant externe, je peux ajouter des billets dans la base de données.	2.1 Tout billet doit être obligatoirement associé à une réservation.
		2.2 Interdiction d'avoir deux fois le même codeQR dans la table billet.
		2.3 Lister et contrôler le nombre de billets existants par nom et prénom de visiteur
4	En tant qu'acheteur, je peux imprimer les billets d'un autre acheteur.	

Documents associés au dossier B

Document B1 : Récit utilisateur (user story) n°1 "Achat des billets en ligne"

Titre : Achat des billets en ligne Valeur Métier : 15

Objectif : En tant qu'acheteur, je veux acheter en ligne les billets pour plusieurs personnes afin de pouvoir participer à une visite guidée.

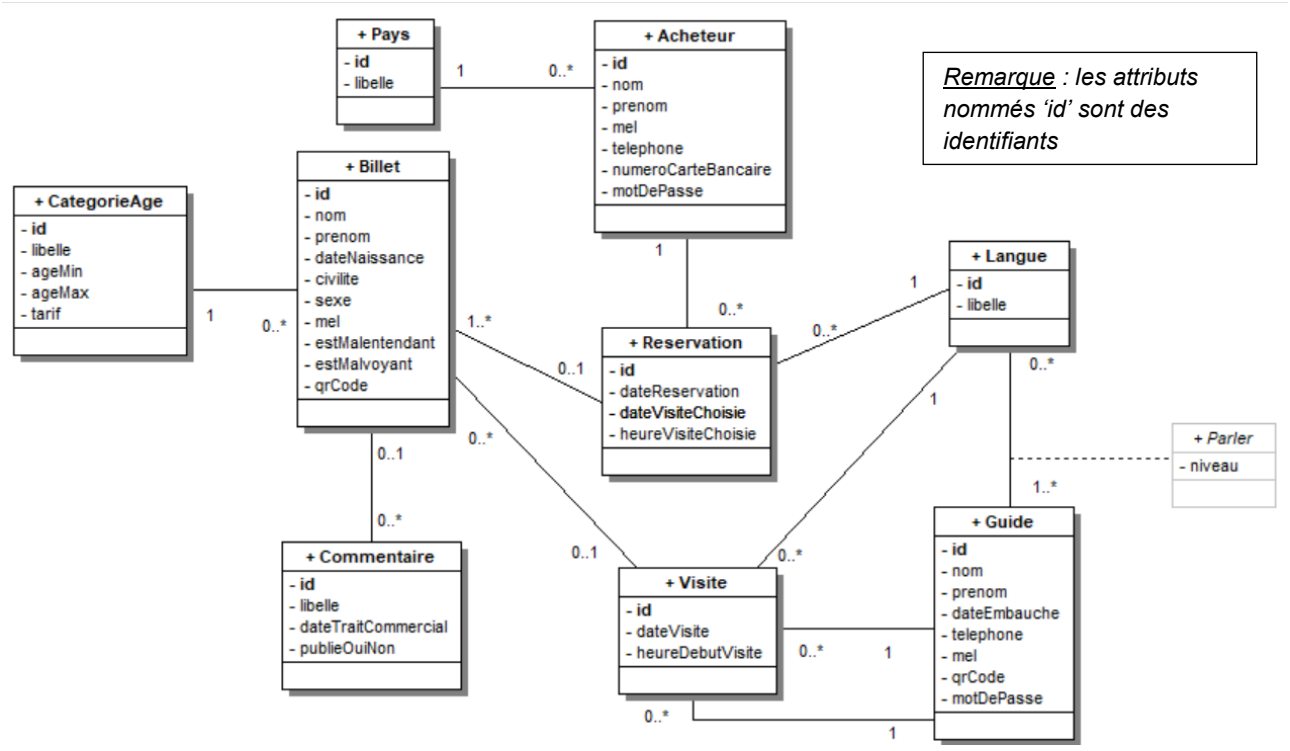
Rattachement : itération (*sprint*) n°1

Critères d'acceptation:

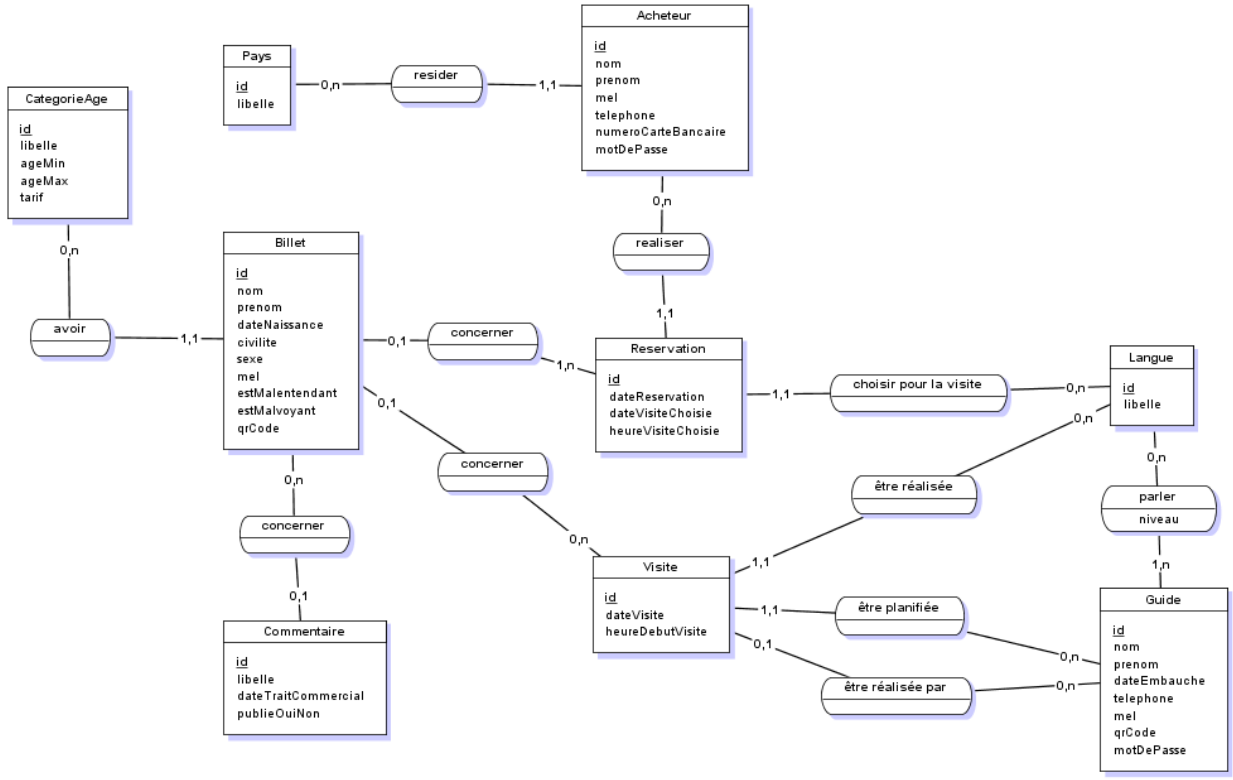
- ➢ Pour réserver des billets sur internet, une personne (appelée acheteur) devra créer un compte en fournissant son adresse mél qui sera utilisée comme identifiant, et un mot de passe.
- ➢ Une fois connecté, l'acheteur obtiendra un premier formulaire lui proposant de renseigner les informations utiles à la réservation de billets : son nom, son prénom, son mél, son numéro de téléphone. Puis,il devra choisir la date et l'heure souhaitées pour la visite, la langue dans laquelle il souhaite que la visite se déroule et le nombre de billets désirés.
- ➢ Ensuite, pour chacun des billets qu'il souhaite réserver, un deuxième formulaire de saisie lui demandera de fournir, pour chaque personne destinatrice d'un billet : son nom, son prénom, sa civilité, son sexe, son adresse mél, si elle est ou non malentendante, si elle est ou non malvoyante et sa date de naissance. La date de naissance est utilisée uniquement pour déterminer la tranche d'âge de la personne ; elle est déterminante pour connaître le tarif du billet. La connaissance du/des handicaps (malentendance, malvoyance) du visiteur permettra de lui préparer un compagnon de visite adapté.
- ➢ Suite à la validation des données saisies, l'acheteur est invité à saisir ses données bancaires. Un paiement sécurisé est effectué.

Document B2 : Représentation conceptuelle de la base de données

Représentation conceptuelle de la base de données avec un diagramme UML :

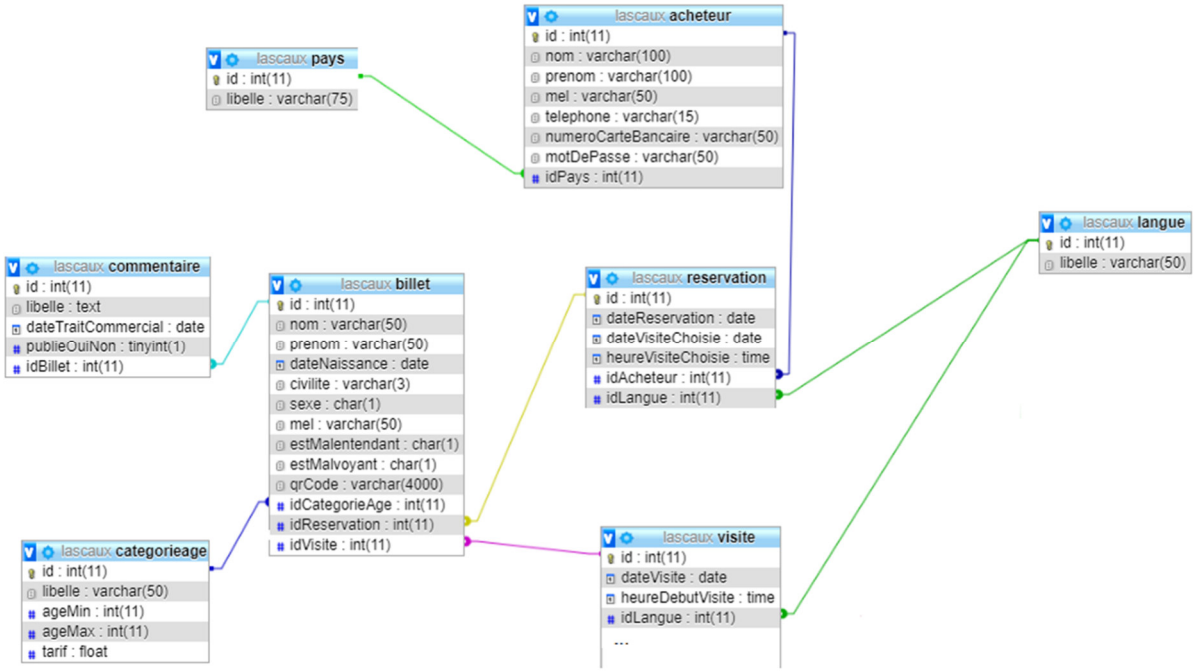


Représentation conceptuelle de la base de données avec un schéma entité-association :



Document B3 : Extrait du schéma relationnel de la base de données utile aux missions 1 et 2

Extrait du schéma relationnel sous forme graphique :



Extrait du schéma relationnel sous forme textuelle :

Pays (id, libelle)
Clé primaire : id

CategorieAge (id, libelle, ageMin , ageMax, tarif)
Clé primaire : id

Langue (id, libelle)
Clé primaire : id

Acheteur (id, nom, prenom, mel, telephone, numeroCarteBancaire, motDePasse, idPays)
Clé primaire : id
Clé étrangère : idPays en référence à id de Pays

Reservation (id, dateReservation, dateVisiteChoisie,heureVisiteChoisie, idAcheteur, idLangue)
Clé primaire : id
Clés étrangères : idAcheteur en référence à id de Acheteur
idLangue en référence à id de Langue

Visite (id, dateVisite, heureDebutVisite, idLangue, ...)
Clé primaire : id
Clés étrangères : idLangue en référence à id de Langue
…

Billet (id, nom, prenom, dateNaissance, civilite, sexe, mel, estMalentendant, estMalvoyant, qrCode, idCategorieAge, idReservation, idVisite)
Clé primaire: id
Clés étrangères : idCategorieAge en référence à id de CategorieAge
idReservation en référence à id de Reservation
idVisite en référence à id de Visite

Commentaire (id, libelle, dateTraitCommercial, publieOuiNon, idBillet)
Clé primaire : id
Clé étrangère : idBillet en référence à id de Billet

Document B4 : Contenu de la table Acheteur

id	nom	prenom	mel	telephone	numeroCarteBancaire	motDePasse	idPays
1	Tesure	Alice	alice.tesure@gmail.com	0600255636	5485889536529889-0625-999	alice60TESURE	1
2	Leduc	Jean	jean.leduc@gmail.com	0617736690	1235882231129889-0322-888	DUDU@458796	1

Document B5 :Mél de Denise Bradord relatif au test du récit utilisateur (user story) n° 5

Bonjour,

Je viens de tester la fonctionnalité décrite dans le récit utilisateur (*user story*) n° 5 qui permet à l'acheteur connecté de visualiser le nom, le prénom et l'adresse mél des séniors pour lesquels il a réalisé des réservations sur un mois et une année sélectionnés.

Afin de tester cette fonctionnalité, j'ai créé deux comptes acheteur : le premier au nom d'Alice Tesure, le deuxième au nom de Jean Leduc. Voici un extrait des caractéristiques de ces deux comptes :

id	nom	prenom	telephone
100	Tesure	Alice	0600255636
101	Leduc	Jean	0617736690

J'ai ensuite saisi deux réservations sur le mois de février 2020 : la première avec le compte d'Alice Tesure et la deuxième avec celui de Jean Leduc :

id	dateReservation	idacheteur
352	2020-02-06	100
353	2020-02-02	101

Puis, j'ai renseigné les caractéristiques des personnes associées aux réservations comme ceci :

id	civilite	nom	prenom	idCategorieAge	idReservation
85	Mme	Gouriton	Alexandra	3	352
86	M.	Foulinos	Romain	4	352
87	M.	Rastic	Jules	4	352
88	Mme	Julouis	Sophie	4	353

Pour rappel, voici les enregistrements existants dans la table CategorieAge :

id	libelle
1	Enfant
2	Adolescent
3	Adulte
4	Senior

Connectée avec le compte d'Alice Tesure, j'ai demandé la liste des caractéristiques des réservations réalisées pour les seniors durant le mois de février 2020 et voici ce que j'ai obtenu :

Réservations réalisées par Alice Tesure pour des seniors durant le mois de février 2020

Nom	Prénom	Mel
Foulinos	Romain	foul.rom@gmail.com
Rastic	Jules	rastic.jules@gmail.com
Julouis	Sophie	julouis.sophie@gmail.com

Comme tu peux le constater, Alice Tesure visualise une réservation réalisée par un autre acheteur, alors qu'elle ne devrait pouvoir consulter que ses propres réservations.

Les données affichées proviennent de la requête SQL paramétrée suivante qui est responsable de ce problème de confidentialité :

```
SELECT *
FROM Acheteur AS A
INNER JOIN Reservation AS R
INNER JOIN Billet AS B ON B.idReservation = R.id
INNER JOIN CategorieAge AS C ON B.idCategorieAge = C.id
INNER JOIN Langue AS L ON R.idLangue = L.id
WHERE A.id = :par_idAcheteur
AND C.libelle = 'senior'
AND YEAR(dateReservation) = :par_annee
AND MONTH(dateReservation) = :par_mois
```

Peux-tu corriger cette requête pour régler le problème de confidentialité que je t'ai soumis ?

Par ailleurs, peux-tu optimiser la requête pour éviter qu'elle ne consomme des ressources inutiles ?

Bon courage !
Cordialement
Denise Bradord.

Document B6 : Syntaxe MySQL (extrait du manuel de référence de MySQL 8.0)

L'instruction CREATE USER permet la création d'un nouveau compte utilisateur sur *MySQL*. Pour chaque compte, cette instruction crée une nouvelle ligne dans la table système *mysql.user*.

Format général de l'instruction CREATE USER :

CREATE USER 'nom_de_compte' IDENTIFIED BY *'password'*;

nom_de_compte respecte la syntaxe suivante 'user_name'@'host_name' où le 'user_name' est le nom de l'utilisateur et le 'host_name' est l'adresse IP de la machine. Un nom de compte composé uniquement d'un nom d'utilisateur est équivalent à 'user_name'@'%'.

Un compte créé pour la première fois n'a pas de privilèges et pas rôle par défaut. Pour attribuer des privilèges ou des rôles, il faudra utiliser l'instruction GRANT.

Format général de l'instruction GRANT :

GRANT priv_type [(column_list)][, priv_type [(column_list)]] …
ON [object_type] priv_level
TO user_or_role [, user_or_role] ...

priv_type contient les privilèges attribués. Les valeurs autorisées sont : ALTER, CREATE VIEW, CREATE, DELETE, DROP, GRANT OPTION, INDEX, INSERT, SELECT, SHOW VIEW, TRIGGER et UPDATE et ALL

ON contient le nom de la base de données suivi du nom de la table : db_name.tbl_name

TO contient le nom du compte

Document B7 : Évolutions demandées par laresponsable de produit (Product Owner) et le délégué à la protection des données (DPO)

Appréciation du niveau linguistique des guides

Lors de la première itération (*sprint*), une employée du service administratif a porté une appréciation sur le niveau de chaque langue parlée par un guide. Voici ci-contre un extrait des appréciations enregistrées dans la table LangueParlee.

idGuide	idLangue	niveau
1	1	Niveau satisfaisant
1	3	Niveau déplorable.Aucun effort pour comprendre !
2	2	Mauvais accent anglais. Difficile à comprendre.
2	3	Niveau satisfaisant
2	4	Doit impérativement se former car il très mauvais.

Au regard des valeurs saisies, la responsable de produit (PO) souhaite que le niveau de langue ne soit plus un texte libre, mais qu'il soit sélectionné dans une liste déroulante dont les valeurs seront gérées via une fonctionnalité de l'application développée dans une prochaine itération (*sprint*).

Authentification et renouvellement des mots de passe

Pour des raisons de sécurité, le compte d'un acheteur doit être verrouillé après quatre échecs de connexion. Par ailleurs, les acheteurs devront changer leur mot de passe tous les quinze jours : le nouveau mot de passe choisi ne devra pas avoir déjà été utilisé par l'acheteur.

Les données sensibles

La connaissance du/des handicaps du visiteur (malentendance, malvoyance) permet de lui préparer un compagnon de visite (CDV) adapté. Lors de la première itération (*sprint*), ces indications ont été enregistrées à tort dans la base de données. Le délégué à la protection des données (DPO) vous demande de retirer ces informations sensibles et de proposer un moyen d'enregistrer pour le visiteur les caractéristiques du CDV à préparer. Plusieurs caractéristiques sont possibles :

- 1 : taille de la police de caractères moyenne ;
- 2 : taille de la police de caractères grande ;
- 3 : volume sonore élevé ;
- 4 : CDV clavier braille.

Document B8 : Fiche de registre établie par le délégué à la protection des données pour le traitement réservation de billet

Description du traitement	
Nom du traitement	Gestion des réservations
N° / RÉF	Ref-1005
Date de création du traitement	02/01/20
Mise à jour du traitement	

Acteurs	Nom	Adresse	Code Postal	Ville	Pays	Téléphone	Adresse mél
Responsable du traitement	Louise DUPONT	1, rue du musée	24290	Montignac	France	05 53 50 99 10	dupont.louise@lascaux.fr
Délégué à la protection des données	Martine DEPEO	1, rue du musée	24290	Montignac	France	05 53 50 99 12	depeo.martine@lascaux.fr
Société du DPO (si celui-ci est externe)							

Finalité(s) du traitement effectué	
Finalité principale	Réservation de billets
Sous-finalité 1	Édition du billet d'entrée
Sous-finalité 2	Planification des visites

Catégories de données personnelles concernées	Description	Durée de conservation
État civil, identité, données d'identification, images…	Pour l'acheteur : nom, prénom, téléphone,adresse mail, mot de passe Pour le visiteur : civilité, nom, prénom, adresse mail	Un an après la date de la dernière réservation
Vie personnelle (habitudes de vie, situation familiale, etc.)		
Informations d'ordre économique et financier (revenus, situation financière, situation fiscale, etc.)		
Données de connexion (adresse IP, logs, etc.)	Pour l'acheteur : logs de connexion	
Données de localisation (déplacements, données GPS, GSM, etc.)		
Numéro de Sécurité Sociale (ou NIR)		

Données sensibles	Description	Durée de conservation
Données révélant l'origine raciale ou ethnique	Aucune	
Données révélant les opinions politiques		
Données révélant les convictions religieuses ou philosophiques		
Données révélant l'appartenance syndicale		
Données génétiques		
Données biométriques aux fins d'identifier une personne physique de manière unique		
Données concernant la santé		
Données concernant la vie sexuelle ou l'orientation sexuelle		
Données relatives à des condamnations pénales ou infractions		

Catégories de personnes concernées	Description	Précisions

Documents associés au dossier C

Document C1 : Exemple de billet produit après une réservation.

Ce billet a été attribué pour un adulte identifiable via le code QR (*QrCode)* figurant sur le billet. Le flashage de ce dernier permettra de déterminer l'identité du visiteur.

Document C2 : Architecture et arborescence de l'application d'identification.

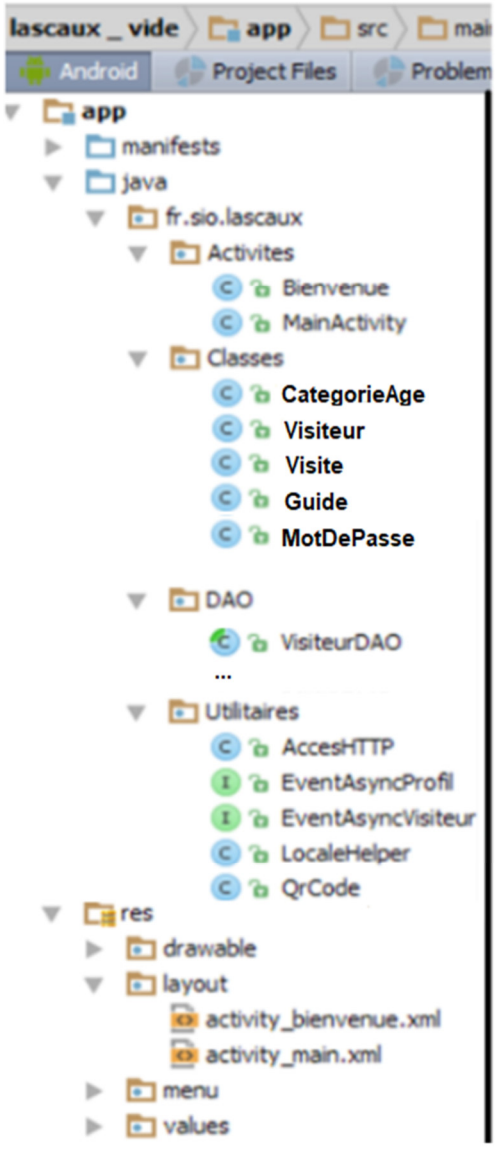
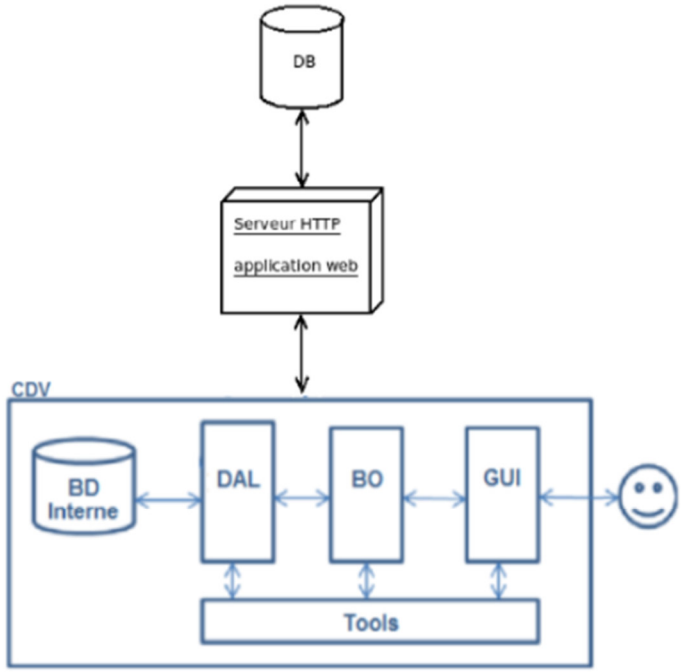


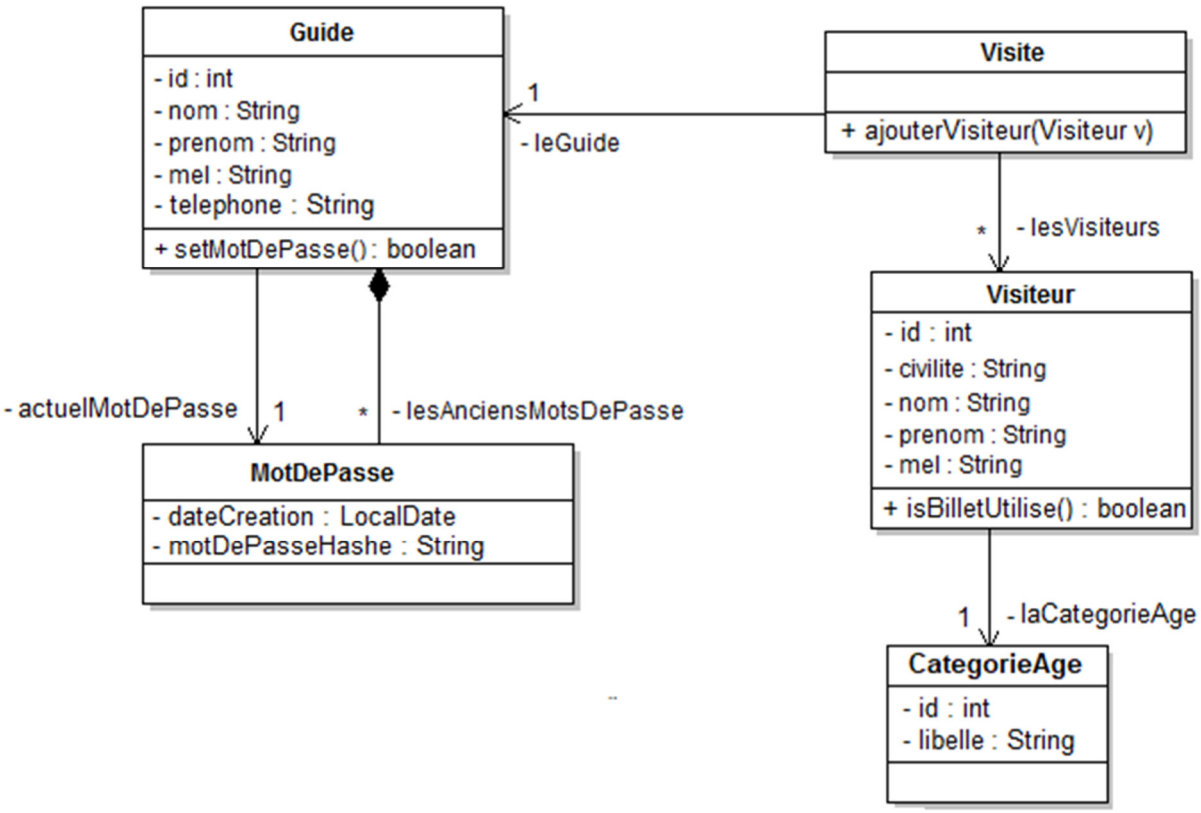
Schéma de principe de l'architecture logicielle de l'application



Les applications seront développées en s'appuyant sur la notion de couches et seront considérées comme multi-couches :

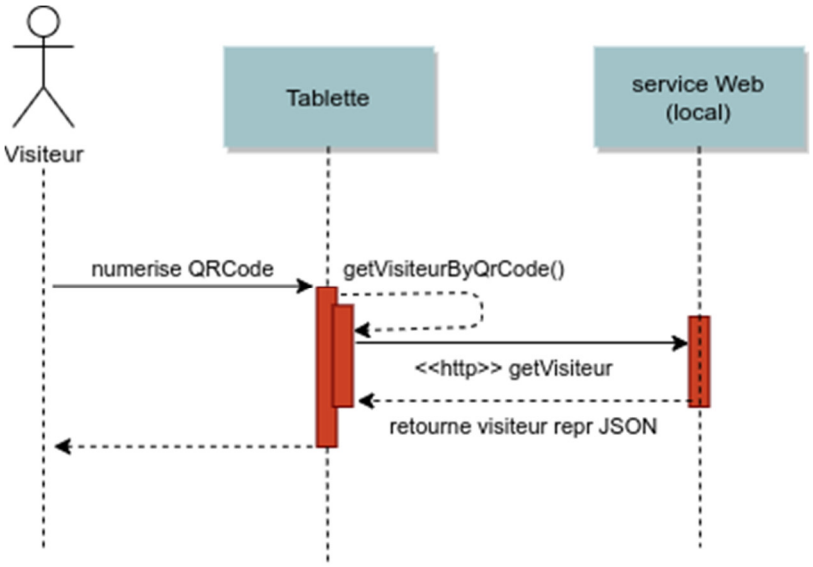
- Les interfaces graphiques (Graphic User Interface GUI) permettront de décrire les interfaces.
- La couche métier (Business Objects BO) sera réalisée par un ensemble de classes correspondant à des objets métiers.
- La couche d'accès aux données(Data Access Layer DAL) assurera l'accès aux données (internes ou externes à l'application).
- La couche outils (Tools) réunira un ensemble de classes techniques utiles à diverses couches.

Document C3 : Diagramme de classes partiel de la partie identification



Remarque : les accesseurs de l'ensemble des attributs ne sont pas représentés sur le diagramme.

Document C4 : Diagramme de séquence de l'identification d'un visiteur par le code QR (QRCode) sur le compagnon de visite (CDV)



Document C5 : Codes partiels de l'application d'identification des visiteurs : appel au service Web.

MainActivity.java

```
/**
* Classe principale de l'application du CDV
*/
public class MainActivity extends Activity {
private Button btnIdentifier;
    private QrCode unQrCode;
    private Visiteur leVisiteur;
    private Visite laVisite;

/**
* Initialise l'activité avec le français comme langue par défaut.
* @param  savedInstanceState l'instance en cours
*/
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Context context = LocaleHelper.setLocale(this, "fr");
        Resources resources = context.getResources();
        setContentView(R.layout.activity_main);
        btnIdentifier = (Button) findViewById(R.id.buttonidentifier);
        btnIdentifier.setOnClickListener(new View.OnClickListener() {

/**
* Gère l'événement clic sur l'activité.
* Récupère la valeur du code QR de la vue
* puis appelle identifierUtilisateur.
* @param v la vue de l'activité
*/
            @Override
            public void onClick(View v) {
String idQrCode;
                unQrCode = new QrCode(); // lit le code QR sur la vue v
                idQrCode = unQrCode.getIdentifiant();
identifierUtilisateur(idQrCode);
            }
        });
    }
```

```
/**
* Construit le Visiteur à partir du code QR scanné
* Initialise le connecteur à la base de données distante
* @param idQrCode le code QR du billet scanné
*/
private void identifierUtilisateur(String idQrCode){
      …
 // partie traitant des visiteurs
/**
      * Recueille les informations nom, prenom, civilite du visiteur et construit leVisiteur
      */
       VisiteurDAO visiteurAcces = new VisiteurDAO(){
          @Override
                public void onTacheTerminee(Visiteur resultat) {
                // une fois la tâche terminée le visiteur est accessible
                this.leVisiteur = resultat;
                if (this.leVisiteur != null) {
                        // la visite 'laVisite'a été initialisée préalablement
                        this.laVisite.ajouterVisiteur(this.leVisiteur);
                // transmission à l'activité Bienvenue du visiteur identifié
                Intent intent = new Intent(getBaseContext(), Bienvenue.class);
                intent.putExtra("monVisiteur", leVisiteur);
                startActivity(intent);
                Toast.makeText(getApplicationContext(), "Chargement des données",
                Toast.LENGTH_LONG).show();
                 }
                else{
                        Toast.makeText(getApplicationContext(), "QrCode incorrect",
                        Toast.LENGTH_LONG).show();
                }
              } // fin de onTacheTerminee
         }; // fin de  new VisiteurDAO()
//appel de la méthode getVisiteurByQrCode dans VisiteurDAO
visiteurAcces.getVisiteurByQrCode(idQrCode);
  }
}
```

VisiteurDAO.java

```
/*** Classe permettant l'accès aux données (data access object)*/
public abstract class VisiteurDAO implements EventAsyncVisiteur {
        private static final String serveur="www.Lascaux.artparietal.fr";
        private static final String chemin="/lascaux/";
        public VisiteurDAO(){ … }//Constructeur
    /**
    * Envoie une requête HTTP pour récupérer les données du visiteur à partir du QrCode scanné
    * @param unQrCode chaîne contenant le codeQR du billet
    */
     public voidgetVisiteurByQrCode(String unQrCode){
        // declenche la requete HTTP http://www.Lascaux.artparietal.fr/lascaux/getVisiteur.php
                AccesHTTP requeteHttp = new AccesHTTP(){
 [...   récupère l'information reçue dans this.ret … ]
                        @Override
                        protected void onPostExecute(Long result) {
                        // une fois le Post réalisé, analyse et récupération des informations json
                                onTacheTerminee(jsonStringToVisiteur(this.ret));
                        }
                };
                requeteHttp.addParam("qrcode",unQrCode);
                requeteHttp.execute("http://" + serveur + chemin + "getVisiteur.php");
        }
  /**
    * Crée un objet visiteur à partir du flux json reçu en paramètre
    * @param jsonString chaîne Json en provenance du serveur
    * @return un visiteur correctement initialisé ou null si échec
    */
        private Visiteur jsonStringToVisiteur(String jsonString){
                 Visiteur unVisiteur = null;
                CategorieAge laCategorieAge,
                 StringnomV,civiliteV, prenomV,melV, libCategorieAgeV;
                 int idV, idCategorieAgeV;
                 try {
                        JSONObject objJson = new JSONObject(jsonString);
                         idV =  Integer.parseInt(objJson.getString("id"));
                         civiliteV = objJson.getString("civilite");
                         nomV = objJson.getString("nom");
                         prenomV = objJson.getString("prenom");
                         melV = objJson.getString("mel");
                         idCategorieAgeV = Integer.parseInt(objJson.getString("idCategorieAge"));
                         libCategorieAgeV = objJson.getString("libCategorieAge");
                         laCategorieAge= new CategorieAge(idCategorieAgeV, libCategorieAgeV);
            unVisiteur = new Visiteur(idV,civiliteV,nomV,prenomV,melV, laCategorieAge);
                 }
                 catch (JSONException e){
                        Log.d("log","pb decodage JSON");
                }
                 return unVisiteur;
    }
}
```

Document C6 : Codes partiels de l'application d'authentification : gestion des mots de passe du guide.

Guide.java

```
/** * Classe Guide*/
publicclassGuide {
private int id;
private String nom;
private String prenom;
private String mel;
private String telephone;
private MotDePasse actuelMotDePasse;
private ArrayList <MotDePasse> lesAnciensMotsDePasse;
    /**
    * Ce constructeur valorise l'ensemble des attributs.
*/
        public Guide(int id, String nom, String prenom, String mel, String telephone,
MotDePasse  actuelMotDePasse, ArrayList <MotDePasse>lesAnciensMotsDePasse ) {
                ...
this.lesAnciensMotsDePasse = lesAnciensMotDePasse;
this.actuelMotDePasse = actuelMotDePasse;
        }
// Méthode à compléter pour être conforme à l'évolution demandée
  /**
    *  Vérifie si le nouveau mot de passe reçu en paramètre est conforme aux spécifications.
    *  Si le nouveau mot de passe est différent du mot de passe actuel, on mémorise l'instance de
    *  l'ancien mot de passe dans l'historique des anciens mots de passe, puis on crée une instance
    *  pour le nouveau mot de passe qui devient l'actuel mot de passe du guide(déjà hashé).
    * @param unMotDePasse le mot de passe hashé
    * @return true (vrai) si la modification a été effectuée ou false (faux) sinon
*/
        public boolean setMotDePasse(String unMotDePasse) {
                boolean modifARealiser = true;
                if(unMotDePasse.equals(this.actuelMotDePasse.getMotDePasseHashe()){
                        modifARealiser = false;
                } else{
this.lesAnciensMotsDePasse.add(this.actuelMotDePasse);
                this.actuelMotDePasse = new MotDePasse( LocalDate.now(), unMotDePasse);
                }
                return modifARealiser;
        }
// Méthode à écrire
  /**
    * Vérifie si le mot de passe est périmé, c'est-à-dire s'il a plus de trois mois
    * @return true (vrai) si le mot de passe a plus de trois mois, false (faux) sinon
    */
        public boolean doitChangerMdP(){...}
}
```

MotDePasse.java

```
/**
* Classe MotDePasse
* mémorise un mot de passe avec sa date de création. Le mot de passe est réputé hashé.
* @version 2.0
*/
public class MotDePasse {
        private LocalDate  dateCreation;
        private String motDePasseHashe;
/**
    * constructeur de la classe
    * @param dateCrea date de début d'utilisation
    * @param motDePasse le mot de passe hashé
*/
        public MotDePasse(LocalDate dateCrea, String motDePasse) {
           this.dateCreation = dateCrea;
           this.motDePasseHashe = motDePasse;
        }
 /**
    * accesseur de la date de création
    * @return la date de la création du mot de passe
*/
        public LocalDate getDateCreation() {
           return dateCreation;
        }
 /**
    * accesseur du mot de passe
    * @return le mot de passe
*/
        public String getMotDePasseHashe() {
           return motDePasseHashe;
        }
}
```

Document C7: Scripts PHP de la phase d'identification du visiteur

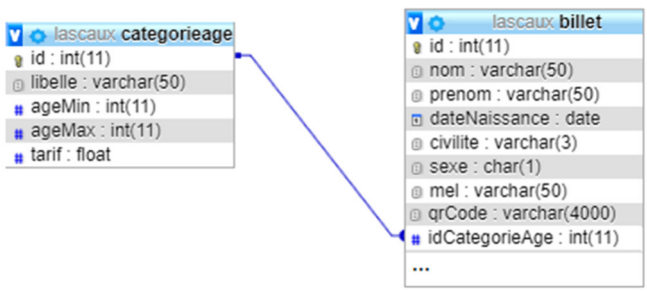
getVisiteur.php

```
<?php
// Le code du script de connexion a été testé et approuvé au niveau de la sécurité
// il initialise la variable globale $pdo
require_once 'connect-db.php';
/**
* Obtenir les informations du visiteur, en représentation JSON, à partir d'une valeur de qrCode d'un billet
* @param string $qrCode
* @global PDO $pdo une référence PDO de connexion à la base de données
* @return string JSON représentation du visiteur ou la cause de l'erreur
*/
function getVisiteurByQrCode($qrCode) {
  global $pdo;
  try {
    $sql = "
      SELECT billet.id as id, civilite, nom, prenom, mel,
                categorieAge.id as idCategorieAge, libelle AS libCategorieAge
      FROM Billet
      JOIN CategorieAge ON idCategorieAge = CategorieAge.id
WHERE qrCode = '" . $qrCode . "'";
$req = $pdo->prepare($sql);
    $req->execute();
// retourne la ligne suivante en tant qu'un tableau indexé par le nom des colonnes
    // ou retourne false si aucune ligne n'est retournée
$ligne = $req->fetch(PDO::FETCH_ASSOC);
return json_encode($ligne);

  } catch (PDOException $e) {
    $erreur = array();
// obtient la raison technique de l'erreur
    $erreur['erreur'] =  $e->getMessage();
    return  json_encode($erreur);
  }
}
header("content-type: application/json; charset=utf-8");
echo getVisiteurByQrCode($_POST['qrcode']);
```

Remarque : La balise fermante php (?>) est volontairement absente en fin de script.

Document C8 : Extrait du schéma relationnel exploité par le script getVisiteur.php



Document C9 : Exemple de création d'une requête préparée

Partons d'une requête de type SELECT simple :

```
SELECT car_name, car_type FROM car WHERE car_id = 3
```

Cette requête peut être découpée en deux parties : une partie fixe et une partie variable.
La partie fixe est celle que l'on va envoyer à la base de données pour préparer l'exécution.
La partie variable est celle qui va changer en fonction de la voiture que l'on veut charger.

```
<?php
  $stmt = $pdo->prepare('SELECT car_name, car_type FROM car WHERE car_id = :id');
  $stmt->bindValue(':id', 3, PDO::PARAM_INT);
  $stmt->execute();

  print_r($stmt->fetchObject());
?>
```

Pour une valeur de type chaîne, le 3ème paramètre de la méthode bindValue est **PDO::PARAM_STR**

Dans notre exemple, l'appel de la méthode bindValue permet d'associer la valeur 3 au paramètre :id de la requête préparée.

Document C10 : Descriptif classe ArrayList

public class ArrayList {
public ArrayList() { … } *// Construit une liste vide*
public boolean add(E e) { … } *// Ajoute un élément dans la liste*
public void clear() { … } *// Supprime tous les éléments de la liste*
public boolean contains(Ee) { … } *// Retourne vrai si l'objet passé en paramètre existe dans la liste*
public E get(int index) { … } *// Retourne l'élément positionné à l'emplacement index*
public int size() { … } *// Retourne le nombre d'éléments contenus dans la liste*
public booleanremove(int index) { … } *// Supprime l'élément positionné à l'emplacement index*
}

L'exemple ci-dessous permet de manipuler une collection de chaînes de caractères. Le principe est le même quel que soit le type des éléments.

```
ArrayList<String> mesChaines;// Déclaration d'une collection de chaînes de caractères
mesChaines = new ArrayList<String>();// Instanciation de la collection
mesChaines.add("un");// Ajout d'une chaîne à la collection
mesChaines.add("deux");
for (String uneChaine : mesChaines) {     // Parcours de la collection
System.out.println(uneChaine);          // Affichage de l'élément courant
}
mesChaines.remove(1);                     // Suppression du 2ème élément (indice 1)
System.out.println(mesChaines.get(0));   // Affichage du 1er élément (indice 0)
System.out.println(mesChaines.size());   // Affichage du nombre d'éléments de la collection
```

Document C11 : Documentation partielle de la classe LocalDate en Java

```
//pour obtenir l'année en cours
LocalDate aujourdhui =  LocalDate.now();
int anneeEnCours = aujourdhui.getYear();
System.out.println(anneeEnCours);
LocalDate uneDate = LocalDate.now();
LocalDate uneAutreDate ;
// pour soustraire un mois à une date
uneAutreDate = uneDate.minusMonths(1);
// pour soustraire une année à une date
uneAutreDate = uneAutre.minusYears(1);
// pour ajouter un mois à une date
uneAutreDate = uneAutreDate.plusMonths(1);
// pour comparer deux dates
if (aujourdhui.isAfter(uneAutreDate)) {
System.out.println("aujourd'hui se situe après l'autre date");
} else {
System.out.println("aujourd'hui se situe avant l'autre date");
}
```

Document C12 : Filtrage des données externes (extrait d'une formation)

Les variables super globales $_POST, $_GET, $_COOKIE, et $_REQUEST détiennent des informations transmises via une requête *HTTP* déclenchée par un utilisateur.

Ces informations peuvent potentiellement provenir d'un utilisateur malintentionné. Il existe deux manières de filtrer : soit on valide le contenu de ces informations car on en connaît le type (URL, mél, tél, date par exemple), soit on nettoie le contenu en supprimant ou transformant ce qui pourrait être malicieux comme des apostrophes ou la présence de balises par exemple.

On utilise donc deux types de filtres :

- les filtres de validation (VALIDATE), qui ne modifient pas les données transmises, mais qui cherchent à vérifier qu'ils correspondent à un format. Si ce n'est pas le cas, ils renvoient FALSE.
- Les filtres de conversion (SANITIZE) qui suppriment ou transforment les caractères ou expressions qui ne sont pas conformes à un format sans forcément s'assurer que le résultat soit effectivement valide (par exemple suppression de balises et encodage de caractères spéciaux).

Quelques exemples :

- pour valider un champ 'mel' contenant une adresse courriel via la méthode POST :
  $mel = filter_input(INPUT_POST, 'mel', FILTER_VALIDATE_EMAIL) ;
- pour nettoyer un champ 'mdp' contenant un mot de passe via la méthode POST
  $mdpSaisi = filter_input(INPUT_POST, 'mdp', FILTER_SANITIZE_STRING) ;
- pour valider un champ 'quantité' contenant une valeur entière via la méthode POST pour laquelle on souhaite vérifier qu'elle est comprise entre 1 et 999:
  $qteCom = filter_input(INPUT_POST, 'quantite', FILTER_VALIDATE_INT, array("options" => array("min_range"=>1, "max_range"=>999))) ;

Pour résumer : les filtres sont l'ultime évolution de la chasse à l'option guillemet magique (magic quote).

Documents associés au dossier D

Document D1 : Liste de propositions émises lors de l'itération zéro (sprint 0)

1. Le compte administrateur de *MySQL* (*root*) sera utilisé par tous les scripts *PHP* de l'application ayant besoin de se connecter à la base de données *MySQL*. Il faudra cependant veiller à changer le mot de passe par défaut de ce compte.
2. L'administrateur *Scrum*(*Scrum Master*) définira l'identifiant et le mot de passe de l'administrateur de l'application en production. Ce dernier aura tous les droits sur le contenu des tables utilisées par l'application. Un fichier contenant l'identifiant et le mot de passe de l'administrateur sera remonté dans le référentiel (*repository*) utilisé par l'équipe *Scrum*.
3. Un référentiel (*repository*) privé sera utilisé pour la gestion de versions (*versioning*) des fichiers contenant le code source des applications.
4. Les développeurs pourront tester l'application en utilisant directement les bases de production.
5. Il est indispensable de vérifier qu'il n'y a pas de vulnérabilités connues sur les dernières versions des bibliothèques logicielles (*librairies*) et des infrastructures logicielles (*frameworks)* libres (*open source*) avant de les utiliser.
6. Les caractéristiques de chaque connexion et déconnexion (*user*, IP, date et heure, etc.) seront conservées dans un fichier des événements(fichier *log*). L'écriture dans ce fichier ne sera effective que du lundi au vendredi.

Document D2 : Risques cyber liés aux prestataires et aux sous-traitants et préconisations de la direction générale de la sécurité intérieure(DGSI)

extrait de DGSI Ingérence économique : https://www.economie.gouv.fr/files/dgsi-special-cybersecurite.pdf

Les entreprises et administrations, indépendamment de leur taille, mission ou secteur d'activité, sont de plus en plus fréquemment amenées à confier à des tiers tout ou partie de la gestion de leurs systèmes d'information. Ces prestations peuvent induire des risques sur l'intégrité, la disponibilité ou la confidentialité desdits systèmes.

Pour mener à bien ces missions, les prestataires disposent, en particulier, de connexions informatiques aux réseaux de l'entreprise cliente. Ces accès peuvent être internes lorsqu'un sous-traitant est physiquement sur le site, ou à distance, notamment dans le cas d'infogérance ou de supervision des réseaux. Indifféremment de la méthode de connexion au réseau utilisée, l'entreprise est exposée à de nouvelles vulnérabilités.

Dans le cadre de ses missions de sécurité économique et de protection du patrimoine, la DGSI a traité plusieurs cas d'atteintes à des systèmes d'information perpétrés par un prestataire, au cours ou à l'issue de sa mission. D'autres cas relèvent d'intrusions dans les systèmes d'information du prestataire en vue d'atteindre l'entreprise ou l'administration ayant sollicité le sous-traitant.

1er exemple : les problèmes liés à un sous-traitant durant une prestation

Une administration a accueilli dans ses locaux les employés du sous-traitant pour un projet de refonte de son système d'information. Les accès informatiques nécessaires pour mener à bien la mission leur ont été octroyés. Après avoir constaté des irrégularités dans le système d'information, l'équipe de sécurité de l'administration cliente a découvert que l'un de ces employés ne respectait pas la charte informatique et utilisait du matériel informatique personnel non déclaré.

Après enquête, il s'est avéré que ce prestataire exfiltrait de l'information sensible et menait des actions informatiques au sein de l'administration elle-même, afin d'augmenter ses privilèges au sein du système d'information de celle-ci.

2ème exemple : malveillance d'un sous-traitant à l'issue d'un contrat de prestation non renouvelé

Une entreprise a subi des attaques provoquant l'indisponibilité de son système d'information, ainsi que la perte de données sauvegardées. Après investigations techniques, il a été démontré que le dysfonctionnement provenait d'un compte administrateur mis à disposition de prestataires.

Il est apparu que l'ancien sous-traitant, dont le contrat était arrivé à terme et n'avait pas été renouvelé, disposait toujours des droits d'accès au réseau du client et au compte administrateur.

Animé par un esprit de vengeance, ce prestataire a exfiltré des données sensibles et mené des actions de sabotage (suppression de données).

3ème exemple : Ingérence à l'encontre d'un prestataire pour compromettre le système d'information de l'entreprise cliente

Une entreprise ayant fait appel à un prestataire pour mener à bien des projets d'ingénierie, a interconnecté une partie de son réseau avec celui du sous-traitant afin de permettre des opérations à distance. Quelques mois après, l'entreprise a été victime d'une compromission importante de son système d'information.

Après enquête, il s'est avéré que le prestataire avait fait l'objet d'une compromission de son système d'information, dont l'objectif était d'accéder au système d'information du client final afin d'en exfiltrer des données sensibles.

Préconisations de la DGSI

Afin de réduire les risques d'atteintes aux systèmes d'information, dont ceux de captation et d'exfiltration de données sensibles, la DGSI recommande d'appliquer les bonnes pratiques suivantes :

<u>En amont d'une prestation :</u>

Bien définir le périmètre et les modalités d'interconnexion :

- Cloisonner au maximum le réseau informatique accueillant les prestataires ;
- Créer des comptes utilisateurs temporaires pour les prestataires et ne leur octroyer que les droits strictement nécessaires aux missions confiées.

Bien définir les modalités contractuelles de la prestation :

- Prévoir contractuellement et de manière précise les missions et obligations des prestataires ;
- Inclure une clause de confidentialité dans les contrats de prestation, couvrant toutes les informations et données relatives à l'entreprise, dont le prestataire pourrait être amené à prendre connaissance ;
- Privilégier le recours à des entreprises ayant leur siège social dans l'Union Européenne afin de ne pas s'exposer à des législations étrangères défavorables et à portée extraterritoriale;
- Assurer le suivi des informations et données détenues par le prestataire ; s'assurer, le cas échéant, que ces dernières sont stockées sur des serveurs situés dans l'Union Européenne ;
- Procéder à toute vérification qui paraîtrait utile pour vérifier le respect de la clause de confidentialité prévue par le contrat ;
- Exiger un niveau de sécurité informatique minimal du prestataire (dispose-t-il d'une politique de sécurité des systèmes d'information ? réalise-t-il des audits de ses systèmes informatiques régulièrement ? Ses salariés sont-ils sensibilisés à la sécurité informatique ?).

<u>Durant la prestation :</u>

- Superviser la sécurité, le maintien en condition opérationnelle et de sécurité des équipements utilisés par le prestataire ;
- Surveiller l'ensemble des activités du prestataire (accès, changement de personnels, journaux d'événements réseau) ;
- Nommer un référent pour assurer le bon déroulement du projet et veiller à ce que les règles informatiques et contractuelles soient appliquées.

Après la prestation :

- Couper l'ensemble des flux réseaux et interconnexions avec le prestataire ;
- Suspendre l'ensemble des accès et comptes utilisateurs utilisés par les sous-traitants avant de les supprimer, le cas échéant ;
- Superviser les équipements du réseau ayant été utilisés par le prestataire afin d'exclure toute malveillance potentielle ;
- Rapporter aux responsables de l'entreprise et de l'administration et, le cas échéant, aux autorités, les incidents constatés.