BREVET DE TECHNICIEN SUPÉRIEUR
SERVICES INFORMATIQUES AUX ORGANISATIONS
Option : Solutions logicielles et applications métiers

U7 – CYBERSÉCURITÉ DES SERVICES INFORMATIQUES

SESSION 2025

_____

Durée : 4 heures
Coefficient : 4

_____

Matériel autorisé :
Aucun matériel ni document est autorisé.

Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Le sujet comporte 20 pages, numérotées de 1/20 à 20/20.

Cas Kirassur

Ce sujet comporte 20 pages dont un dossier documentaire de 12 pages.
La candidate ou le candidat est invité(e) à vérifier qu'il est en possession d'un sujet complet.

Barème

DOSSIER A	Prise en compte d'évènements redoutés	20 points
DOSSIER B	Étude de l'ouverture du système d'information aux partenaires extérieurs	35 points
DOSSIER C	Journalisation des accès et gestion des alertes	25 points
	TOTAL	80 points

Dossier documentaire

Présentation du contexte : Kirassur et sa filiale Tecassur

Société d'assurance créée en 1829, Kirassur accompagne aujourd'hui plus de 20 000 assurés répartis sur l'ensemble de la France et autant à l'étranger. Ses clients sont aussi bien des particuliers que des entreprises. Elle a réalisé en 2021 un chiffre d'affaires de 300 millions d'euros (dont 50 % à l'étranger) avec 126 collaborateurs sur le territoire.

Sa longue expérience de l'assurance permet à Kirassur de concevoir et de commercialiser des produits d'assurance très performants et particulièrement adaptés à ses différents clients. Son activité repose en grande partie sur le numérique, notamment grâce à son site internet qui permet la souscription de contrats en ligne, ainsi que la déclaration et le suivi des sinistres[1].

Afin de proposer à ses clients de meilleurs tarifs, des services de qualité et une prise en charge plus rapide en cas de sinistre, Kirassur a sélectionné des garagistes pour constituer un réseau de professionnels agréés.

Pour stimuler sa croissance, Kirassur envisage aujourd'hui de développer des partenariats avec des sociétés pour commercialiser ses produits. Cette stratégie découle d'un constat sur l'évolution de la consommation qui crée un besoin d'assurance pouvant être commercialisé par des non-spécialistes. Ainsi, les assureurs sont aujourd'hui amenés à travailler avec des partenaires issus de trois univers : la distribution d'assurance (courtiers, agents généraux, distributeurs en ligne, comparateurs), l'économie collaborative (plateforme de partage de véhicules, de logements, de services) et le commerce (physique ou en ligne).

Pour gagner en réactivité et en rapidité, Kirassur veut passer au tout numérique et ouvrir l'accès à tous ses produits via des interfaces de programmation (*Application Programming Interface - API*). À terme, elle souhaite lancer une plateforme d'assurance à la demande (*Insurance as a Service*). Les offres, au format *API*, seront directement intégrables par les partenaires qui les adresseront via des adresses réticulaires (*URL*) sécurisées dans des applications *Web*.

Kirassur a créé une filiale nommée Tecassur qui développe en interne ou intègre des solutions informatiques externes pour répondre en premier lieu à ses besoins.

Tecassur a en charge la maintenance corrective et évolutive des applications principales qu'elle a développées. La sécurité étant un enjeu important, l'équipe de maintenance organise régulièrement des ateliers de gestion des risques, notamment pour éprouver la résilience des applications aux fraudes constatées dans le secteur des assurances.

Une équipe de Tecassur est, par ailleurs, chargée d'étudier les principes qui seront mis en œuvre pour ouvrir les applications existantes aux partenaires via des interfaces *API*. Cette étude débouchera soit sur le développement d'une solution interne, soit sur l'achat d'une solution externe.

L'application métier ToutAssur permettant la gestion des sinistres doit être mise à jour pour renforcer la sécurité et répondre aux exigences réglementaires : elle devra intégrer la journalisation des accès et la génération d'alertes.

Titulaire d'un BTS SIO option SLAM et nouvellement embauché(e) par Tecassur, vous participez à ces trois activités.

[1] Sinistre : dommages ou pertes subis par un assuré (Le Robert)

Dossier A – Prise en compte d'évènements redoutés

Un sinistre de type « Bris de glace » peut être déclaré directement par les garagistes agréés par Kirassur, pour le compte d'un assuré. La réparation est alors réglée au garagiste par l'assureur.

Étant donné qu'à plusieurs reprises des compagnies d'assurance ont fait l'objet de fausses déclarations de sinistres « Bris de glace » de la part de garagistes, cet événement redouté doit être détecté par Kirassur et des contre-mesures doivent être prises pour l'éviter.

Mission A1 : Contrer l'évènement redouté

Une nouvelle règle sera appliquée. Si le nombre de sinistres de type « Bris de glace » déclarés pour un même contrat, sur une période de référence, est supérieur à une limite fixée pour cette garantie, alors tout nouveau sinistre déclaré pour ce contrat aura un statut « À vérifier ». Le gestionnaire, responsable du suivi du sinistre, devra contrôler que cette déclaration correspond bien à un sinistre subi par l'assuré du contrat.

Kirassur a étudié deux solutions pour réaliser le contrôle afin d'éviter cette fausse déclaration :

a) réaliser le contrôle décrit précédemment au niveau du code de l'application ;

b) réaliser le contrôle au niveau de la base de données en utilisant un déclencheur (*trigger*).

La deuxième solution (b) a été choisie pour contrer l'évènement redouté.

Question A1.1
Justifier le choix retenu d'utiliser un déclencheur (*trigger*) pour réaliser le contrôle.

Une première version du déclencheur (*trigger*), présentée dans le dossier documentaire, a été réalisée et vous devez la compléter pour qu'elle réponde au besoin.

Question A1.2
a) Expliquer ce que permet de faire la requête située lignes 15 à 20.

b) Écrire le code du traitement décrit lignes 22 et 23 permettant de placer le sinistre dans l'état « À vérifier ».

c) Modifier le déclencheur (*trigger*) pour que le contrôle ne soit réalisé que pour les garanties « Bris de glace ». Indiquer les numéros des lignes concernées par la modification.

Mission A2 : Détecter l'évènement suspect

Une application de gestion des sinistres est utilisée par les gestionnaires pour visualiser les sinistres à traiter. Elle propose, pour le gestionnaire connecté, la liste des sinistres restant à traiter dans l'ordre croissant des dates de leur déclaration. Cette interface, présentée dans le dossier documentaire, s'appuie sur une requête.

L'affichage doit être modifié pour que seuls les sinistres de type « Bris de glace » s'affichent, en plaçant en premier ceux qui ont été considérés comme étant à vérifier. Par ailleurs, il serait souhaitable que le nom de l'assuré apparaisse sur l'écran.

Question A2.1
Modifier la requête associée à l'interface de la liste des sinistres à traiter par un gestionnaire pour qu'elle réponde aux attentes.

Dossier B – Étude de l'ouverture du système d'information aux partenaires extérieurs

Kirassur veut donner la possibilité à des sociétés partenaires de commercialiser ses produits d'assurance. Pour cela, il a été décidé de mettre en place une interface *API* pour interconnecter les systèmes d'information.

Tecassur est chargé du développement d'un prototype d'application permettant de valider les choix principaux. Celui-ci ne préjuge pas des choix technologiques utilisés par la future solution dont le développement pourra être éventuellement externalisé.

Vous avez en charge notamment la documentation des choix implémentés par le prototype pour constituer la base du cahier des charges du futur système.

Mission B1 : Étude des échanges entre les sociétés partenaires, les assurés et Kirassur

Il a été décidé un schéma général d'échange basé sur l'utilisation d'une clé *API* permanente pour les sociétés partenaires et d'un jeton d'accès temporaire pour les assurés.

La clé *API* donne le droit d'accès aux interfaces *API* pour les sociétés partenaires. Les échanges montrent qu'elle est associée aux adresses *URL*. L'équipe préconise le choix du protocole *HTTPS* pour protéger les échanges.

Question B1.1
Justifier le choix de l'équipe :
a) en expliquant l'évènement redouté lors de l'utilisation du protocole *HTTP* ;
b) en expliquant comment le protocole *HTTPS* protège de cet évènement redouté.

Deux systèmes d'authentification sont proposés, l'un pour les sociétés partenaires, l'autre pour les assurés pris en charge par les sociétés partenaires.

Question B1.2
Expliquer pourquoi, malgré la protection qu'offre le protocole *HTTPS*, le système d'authentification ne renvoie pas à l'assuré la clé *API* de la société partenaire gérant son contrat.

Les assurés utiliseront un jeton d'authentification pour utiliser les services de Kirassur.

Les jetons, dès qu'ils sont générés, sont stockés dans la base de données.

Question B1.3
Expliquer si les jetons stockés dans la base de données donnent l'accès à des informations à caractère personnel.

Dans le prototype, la base de données est manipulée par une classe dédiée appelée DAO (*data access object*).

Il est nécessaire de s'assurer que la classe DAO a été développée dans le respect des bonnes pratiques de sécurité.

Question B1.4
Identifier en l'expliquant la bonne pratique de sécurité implémentée par la méthode *tokenEnregistrer* mais non respectée par la méthode *tokenCharger*.

Il a été décidé de limiter dans le temps la validité d'un jeton. Par exemple, la validité d'un jeton associé à une page pour consulter un contrat est de 10 minutes (soit 600 secondes).

Le jeton est renouvelé périodiquement, le code de renouvellement n'est pas écrit à ce stade dans le prototype.

De même, le code pour vérifier que cette limite n'est pas atteinte n'est pas encore écrit.

Question B1.5

a) Justifier le contrôle de durée de validité.

b) Identifier les instructions de la classe DAO qui calculent la limite de validité (date heure) ainsi que la table et le champ qui stockent cette valeur en base de données.

Mission B2 – Accès illicite à des données

Lors d'un atelier d'analyse de risque, un membre de l'équipe a exposé l'évènement redouté suivant : une entreprise partenaire en utilisant son code *API* essaie de consulter l'ensemble des contrats, même ceux dont il n'est pas à l'origine, en incrémentant le numéro de contrat à chaque accès à l'interface *API*.

Le test de cet évènement redouté a permis d'obtenir le fichier journal du serveur *Web*, donné dans la partie documentaire, montrant la suite de tentatives d'accès.

Question B2.1

Vérifier si le code existant permet d'accéder de façon illégitime aux contrats, si oui expliquer ce qu'il faudrait faire pour l'éviter (sans coder).

Il faut aussi envisager les moyens permettant de se protéger au niveau de l'infrastructure réseau.

Question B2.2

Donner un moyen pour invalider, au niveau d'un réseau, une adresse *IP* externe ayant un comportement anormal.

Si, malgré la sécurité mise en œuvre, un tel incident venait à se produire, Tecassur doit mettre en place des procédures au regard de la réglementation.

Question B2.3

Identifier les obligations que Tecassur doit respecter, dans le cadre du RGPD, suite à une violation des données personnelles associées aux contrats.

Dossier C – Journalisation des accès et gestion des alertes

L'application de gestion des dossiers, actuellement développée, utilise un système de journalisation (*log*) mis en place principalement pour faciliter les tests en développement.

Cette application est construite à l'aide d'une architecture logicielle de type Modèle-Vue-Contrôleur (*MVC*) avec une partie modèle utilisant des classes métier et des gestionnaires de classe pour la manipulation de ces objets.

Votre équipe a été missionnée pour reprendre une partie des éléments de l'application actuelle afin d'intégrer les contraintes liées à la journalisation des actions des utilisateurs connectés.

Mission C1 : Journalisation des accès à la base de données et aux actions

Pour répondre au besoin de journalisation des accès aux fonctionnalités applicatives, l'équipe de développement a fait le choix d'enregistrer dans une table Action de la base de données l'ensemble des actions exécutables par les gestionnaires de sinistres sur l'application.

Une action considérée comme sensible permet à un utilisateur mal intentionné d'en tirer un profit illégitime en manipulant des données personnelles d'un assuré ou d'un tiers. Les actions identifiées comme sensibles sont indiquées par l'attribut estSensible de type booléen dans la table Action.

On vous demande de tracer les actions sensibles dans un nouveau fichier de journalisation, dont la description se trouve dans le dossier documentaire.

Question C1.1

a) Compléter le code de la méthode *logAction* de la classe ActionManager.

b) Compléter le code de la méthode *enregistrerActionSensible* du contrôleur pour journaliser l'action demandée si elle est sensible.

Mission C2 : Gestion des actions sensibles et modélisation d'un système d'alerte

En utilisant l'application de gestion des sinistres ToutAssur, les gestionnaires ont accès aux informations sur les assurés et leurs dossiers. Le risque d'extraction de ces données par un gestionnaire malveillant conduit Kirassur à prévoir un système de supervision des actions sensibles avec des remontées d'alertes. Elle souhaite profiter de ce développement pour renforcer l'actuel système de gestion des habilitations assez complexe à utiliser.

Les gestionnaires de sinistres et leur responsable

Deux types d'utilisateurs sont amenés à utiliser l'application : les gestionnaires et les responsables d'équipe. Chaque gestionnaire est encadré par un responsable.

Les gestionnaires gèrent les sinistres des assurés. Ils sont les principaux interlocuteurs des assurés.

Les responsables accèdent à des fonctionnalités supplémentaires dans l'application ToutAssur, comme répartir les sinistres entre les agents gestionnaires. Dans l'évolution de l'application, ils devront en complément veiller aux actions sensibles. Pour ce faire, ils auront accès à une nouvelle page affichant le relevé des alertes d'abus.

Gestion des habilitations ponctuelles et profils

Les autorisations d'accès aux pages de l'application pour les utilisateurs connectés sont actuellement gérées par des habilitations uniques qui concernent chacune un utilisateur et une action pour une période définie. La trace des habilitations prolongées ou reconduites n'est pas conservée ; une habilitation reconduite écrase la précédente.

On souhaite faciliter la gestion de ces droits en mettant en place des profils de responsabilités qui englobent un ensemble d'actions autorisées. Le système d'habilitation unitaire sera conservé pour délivrer des habilitations ponctuelles en plus des profils, afin de confier des responsabilités à des gestionnaires montant en compétence ou afin de déléguer des actions à d'autres responsables en cas de vacances.

L'attribution d'un profil à un utilisateur sera, comme pour les attributions d'habilitations unitaire, bornée sur une période et sans historisation. Un profil donné pourra concerner plusieurs utilisateurs.

Par exemple :

- Yanis Elaouari se voit attribuer le profil « payeur » du 1 juillet 2022 au 30 juin 2026, qui englobe les actions « valider un dossier », « mettre en paiement », « saisir coordonnées bancaires », « modifier coordonnées bancaires », « vérifier les virements ».
- Lors du congé de sa responsable du 21 juillet 2022 au 8 aout 2022, il reçoit une habilitation ponctuelle sur l'action « valider les mises en paiement ».

Un profil n'est pas lié à un type d'utilisateur (gestionnaire, responsable).

Les actions sensibles

Les actions sensibles nécessitent une supervision des responsables. Chaque action sensible effectuée fera l'objet d'une journalisation dans la base de données avec un identifiant et un horodatage.

Si les journaux montrent qu'une action sensible est réalisée par le même utilisateur au-delà du nombre de fois acceptable sur un intervalle de temps défini, une alerte sera enregistrée. On a donc besoin de stocker pour chaque action sensible, ce qui permettra de déclencher une alerte, c'est-à-dire, un intervalle de surveillance (exprimé en nombre de jours) et un nombre maximum d'appels à cette action sur cet intervalle.

Par exemple : l'action « Modifier les coordonnées bancaires d'un tiers » déclenchera une alerte à la 3ème occurrence survenue dans un intervalle de 5 jours.

Les alertes

Dans la nouvelle version de l'application ToutAssur, une page « Alertes » listera l'ensemble des alertes à traiter. Pour chaque alerte, l'utilisateur, l'action, la date et l'heure de déclenchement seront affichés. Il sera également nécessaire de mémoriser l'état actuel de chaque alerte (« nouvelle », « affectée », « résolue » ou « classée ») pour avoir une vue globale des alertes à traiter.

Enfin, à partir de la page « Alertes », un bouton « Traiter » permettra à un responsable de prendre en charge une alerte. L'alerte passera alors dans l'état « affectée » et l'application mémorisera le responsable ayant pris en charge l'alerte ainsi que la date et l'heure d'affectation.

La schématisation des données nécessaires à l'évolution de l'application de gestion des habilitations a été débutée. Vous devez la finaliser.

A noter que les sinistres et les assurés ne sont pas concernés par ces évolutions.

Question C2.1

Finaliser, dans le formalisme de votre choix, la schématisation des données pour intégrer l'ensemble des évolutions.

Documents communs à tous les dossiers

Document 1 : Schéma de la base de données (extrait)

Schéma relationnel de la base de données sous forme graphique :

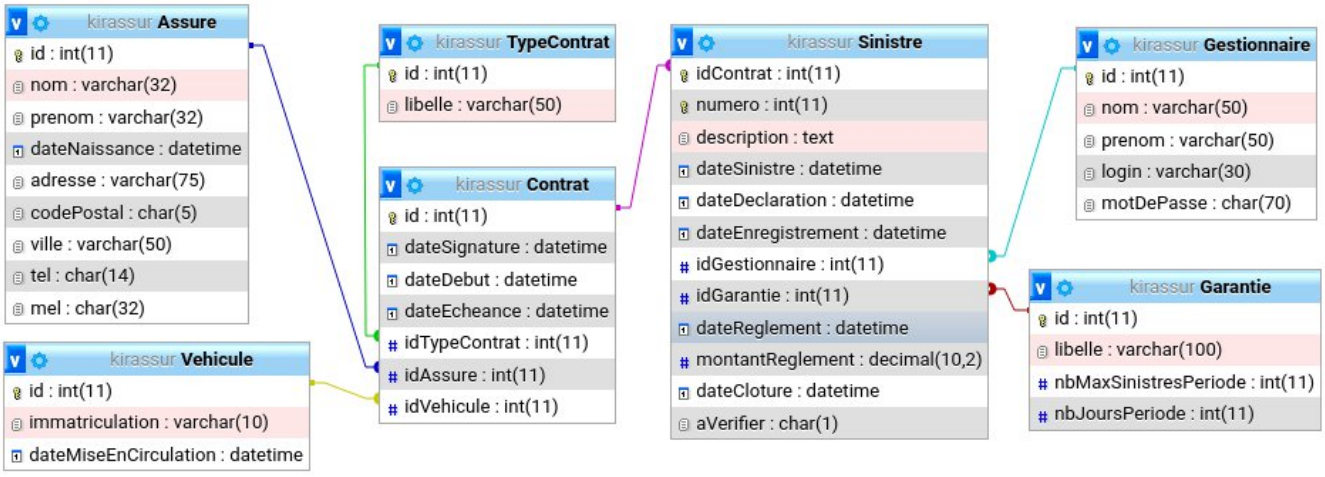


Schéma relationnel de la base de données sous forme textuelle :

Assure (id, nom, prenom, dateNaissance, adresse, codePostal, ville, tel, mel)
id : clé primaire
TypeContrat (id, libelle)
id : clé primaire
Vehicule (id, immatriculation, dateMiseEnCirculation)
id : clé primaire
Contrat (id, dateSignature, dateDebut, dateEcheance, idTypeContrat, idAssure, idVehicule)
id : clé primaire
idTypeContrat : clé étrangère en référence à id de TypeContrat
idAssure : clé étrangère en référence à id d'Assure
idVehicule : clé étrangère en référence à id de Vehicule
Gestionnaire (id, nom, prenom, login, motDePasse)
id : clé primaire
Garantie (id, libelle, nbMaxSinistresPeriode, nbJoursPeriode)
id : clé primaire
Sinistre (idContrat, numero, description, dateSinistre, dateDeclaration, dateEnregistrement, idGestionnaire, idGarantie, dateReglement, montantReglement, dateCloture, aVerifier)
idContrat, numero : clé primaire
idContrat : clé étrangère en référence à id de Contrat
idGestionnaire : clé étrangère en référence à id de Gestionnaire
idGarantie : clé étrangère en référence à id de Garantie

Remarques :

- l'attribut aVerifier de la relation Sinistre contient O si le sinistre est à vérifier, N sinon.

- extrait des données de la table Garantie :

id	libelle	nbMaxSinistresPeriode	nbJoursPeriode
1	Responsabilité civile	NULL	NULL
2	Vol	NULL	NULL
3	Bris de glace	2	90
4	Dégâts matériels	NULL	NULL

Si 2 sinistres de type « Bris de glace » ont été déclarés pour un même contrat, sur une période de 90 jours alors tout nouveau sinistre déclaré sur le contrat et portant sur une garantie « Bris de glace » fera l'objet d'un contrôle. Les attributs nbMaxSinistresPeriode et nbJoursPeriode ne sont pas valorisés pour les autres garanties car aucun contrôle n'est réalisé pour celles-ci.

Documents dossier A

Document A1 : Déclencheur (trigger) utilisé pour gérer les sinistres à contrôler

Ce déclencheur est associé à une base *MySQL*.

```
1  CREATE TRIGGER controleSinistre
2  BEFORE INSERT ON Sinistre FOR EACH ROW
3  BEGIN
4    DECLARE nbSinDeclare int;
5    DECLARE dateDebPeriode datetime;
6    DECLARE nbJours int, nbMaxSinistres int;
7    DECLARE libelleGarantie varchar(100);
8    -- Récupération les données correspondant à l'idGarantie du sinistre
9    SELECT libelle, nbMaxSinistresPeriode, nbJoursPeriode
10   INTO libelleGarantie, nbMaxSinistres, nbJours FROM Garantie
11   WHERE id = NEW.idGarantie;
12   -- Calcul de la date de début de période de contrôle
13   SET dateDebPeriode = SUBDATE(NEW.dateSinistre, INTERVAL nbJours DAY);
14 -- Requête à expliquer (question A1.2 a) --
15   SELECT count(*) into nbSinDeclare
16   FROM Sinistre
17   JOIN Garantie ON Garantie.id = Sinistre.idGarantie
18   WHERE Sinistre.idContrat = NEW.idContrat
19   AND libelle = "Bris de glace" AND dateSinistre >= dateDebPeriode
20   AND dateSinistre <= NEW.dateSinistre ;
21 -- Traitement à ajouter (question A1.2 b) --
22   -- si le nombre de sinistres trouvés sur la période est supérieur au nombre maximum
23   -- autorisé sur la période, alors le sinistre est mis dans l'état "à vérifier"
24 END
```

Remarque :

Instruction permettant de modifier la valeur d'un attribut du sinistre en cours d'ajout :

SET NEW.nomAttribut = nouvelle valeur ;

Document A2 : Interface de la liste des sinistres à traiter par un gestionnaire (exemple)

Agence Kirassur

Gestionnaire connecté : jules.bourdin@kirassur.com

SINISTRES À TRAITER | GESTION DES DOSSIERS

liste des dossiers sinistre à traiter

numéro sinistre	description	date sinistre	date declaration	date enregistrement	Garantie	à verifier	description detaillée du dossier
256203-1	bris de pare-brise - vendalisme	10/01/2022	10/01/2022	10/01/2022	Bris de Glace	X	description du dossier
158625-4	bris de pare-brise - vendalisme	11/01/2022	11/01/2022	12/01/2022	Bris de Glace		description du dossier
258901-2	refus de priorité	05/03/2022	08/03/2022	10/03/2022	Dégâts matériels		description du dossier
256203-3	chute de tuiles : bris de pare-brise	21/03/2022	22/03/2022	22/03/2022	Bris de Glace	X	description du dossier
256203-3	projectile jeté : bris de pare-brise	05/04/2022	05/04/2022	06/04/2022	Bris de Glace	X	description du dossier
258901-1	collision avec poids lourd sur autoroute	08/05/2022	18/05/2022	18/05/2022	Dégâts matériels	X	description du dossier
256203-4	bris de pare-brise	08/05/2022	18/05/2022	18/05/2022	Bris de Glace	X	description du dossier

Copyright 2019 Kirassur - Toute reproduction interdite Mentions Légales

Document A3 : Requête associée au document A2

```
SELECT Contrat.id, Sinistre.numero, Sinistre.description, Sinistre.dateSinistre, Sinistre.da-
        teDeclaration, Sinistre.dateEnregistrement, Garantie.libelle, Sinistre.aVerifier
FROM Contrat
INNER JOIN Sinistre on Contrat.id = Sinistre.idContrat
INNER JOIN Garantie on Garantie.id = Sinistre.idGarantie
WHERE Sinistre.dateReglement is null
AND Sinistre.idGestionnaire = :idGestionnaireConnecte
ORDER BY Sinistre.dateDeclaration
```

Où idGestionnaireConnecte est une variable qui contient l'identifiant du gestionnaire connecté.

Document A4 : Documentation de la fonction SUBDATE de MySQL

SUBDATE(date, INTERVAL expr unit) ou SUBDATE(date, expr)

Où *date* est la valeur de la date à modifier, *expr* est une expression numérique, *unit* une unité de temps parmi (MICROSECOND, SECOND, MINUTE, HOUR, DAY, WEEK, MONTH, QUARTER, YEAR).

Exemple première forme : SELECT SUBDATE('2021-05-30', 15);

Retourne : '2021-05-15'

Exemple seconde forme : SELECT SUBDATE('2021-05-30', INTERVAL 1 YEAR);

Retourne : '2020-05-30 '

Documents dossier B

Document B1 : Authentification des sociétés partenaires à l'aide d'une clé API

Une interface de programmation (*API* : *application programming interface*) permet de rendre disponibles les données ou les fonctionnalités d'une application existante afin que d'autres applications les utilisent.

La plupart de ces interfaces requièrent une clé (*API key*). Cette clé permet à l'interface *API* d'authentifier les logiciels tiers qui utilisent les fonctionnalités exposées.

Si les données et les fonctionnalités manipulées sont protégées, il faut absolument garantir l'identité du partenaire commanditaire et contrôler que ses actions sont permises.

Document B2 : Authentification des assurés à l'aide de jetons d'authentification

Un jeton d'authentification (*token*), appelé parfois jeton de sécurité, est une séquence aléatoire de caractères générés par Kirassur, qui possède une durée de vie limitée.

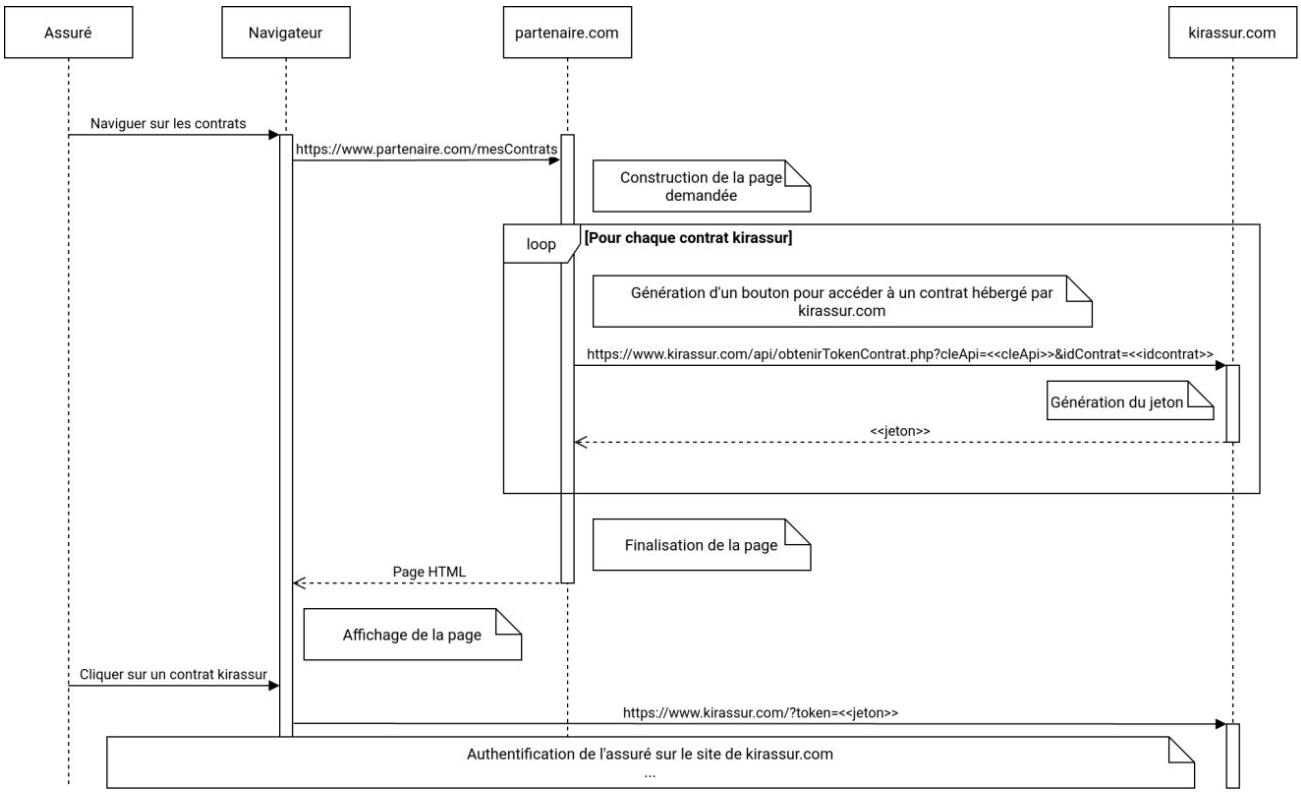
Un jeton est sécurisé par les trois propriétés suivantes :

- valeur unique : il est improbable que 2 jetons aient la même valeur ;
- non prévisible : la connaissance de la valeur d'un jeton ou d'une suite de jetons ne permet pas d'anticiper la valeur d'un autre jeton (*token*) ;
- durée limitée dans le temps : après un horaire déterminé, il n'est plus utilisable.

En général, un jeton est associé à une action. Seule une requête détentrice d'un jeton valide peut réaliser l'action ou la suite d'actions associées.

Document B3 : Connexion d'un assuré via une société partenaire (utilisation d'une clé API puis d'un jeton)

Les sociétés partenaires s'authentifient à l'aide d'une clé *API*. Les assurés reçoivent un jeton qui leur permet d'accéder à leur contrat après une seconde phase d'authentification (non implémentée dans le prototype).



Document B4 : Code PHP de www.KIRASSUR.com/api/obtenirTokenContrat.php hébergé par KIRASSUR (extrait) :

Exemple d'adresse *URL* correcte déclenchant cette page : https://www.KIRASSUR.com/api/obtenirTokenContrat.php?cleAPI=ABCD-ABCDEFGHIJ-ABCD&idContrat=10001

```
1. <?php
2. require("./autoload/autoload.php") ;
3. // filtrage des données externes : remplace html_special_char
4. $cleAPI = filter_input(INPUT_GET,"cleAPI", FILTER_SANITIZE_STRING);
5. $idContrat = filter_input(INPUT_GET,"idContrat", FILTER_SANITIZE_STRING);
6. $dao = new DAO();
7. if($dao->cleAPIVerifierValidite($cleAPI)){
8.   //Génération du token
9.   $octetsAleatoires = openssl_random_pseudo_bytes(9) ;//Création de la séquence aléatoire de 9 octets à la base du token
10.  $token = bin2hex($octetsAleatoires) ; //Conversion de $octetsAleatoires en hexadécimal pour être codable dans l'URL
11.  $dao->tokenEnregistrer($token, $idContrat, 1, 600) ;
12.  die($token) ; //arrête la connexion en retournant le token généré dans le flux HTTP
13. } else {
14.  die() ; //arrête immédiatement la connexion.
15. }
```

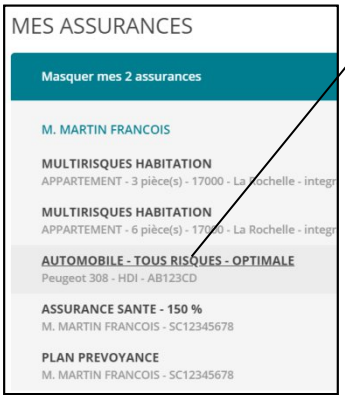
autoload/autoload.php : script qui charge toutes les classes et bibliothèques nécessaires.

dao : objet permettant de manipuler la base de données. Il est généré par le fichier « *autoload.php* ».

Document B5 : Code HTML lié au bouton d'accès à un contrat pour un assuré

Exemple de vue affichée lorsque l'assuré François Martin clique sur le bouton associé à son dossier sur le site de la société partenaire qui lui a vendu ce contrat :

IHM de l'assuré :



Extrait du code HTML du bouton « AUTOMOBILE – TOUS RISQUES – OPTIMALE » :

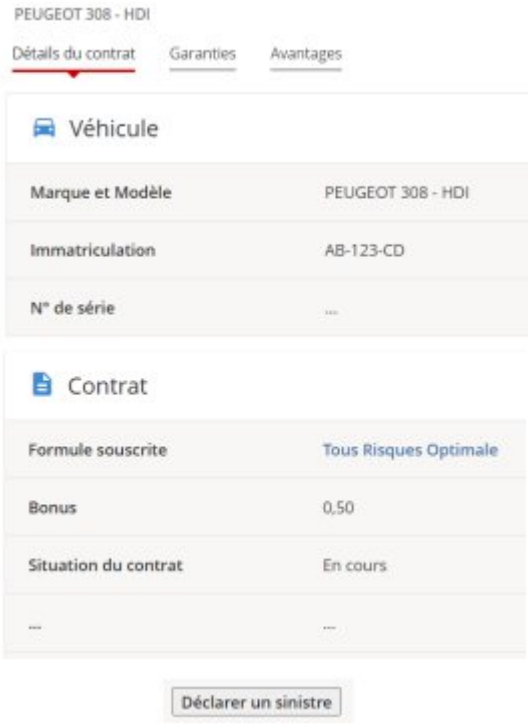
```
<div id="KIRASSUR-container-bouton">
   <a href="https://www.KIRASSUR.com/index.php?token=ae29cf450b6786bf87">
    <button>
        AUTOMOBILE – TOUS RISQUES – OPTIMALE
    </button>
  </a>
</div>
```

La valeur « ae29cf450b6786bf87 » est un jeton.

L'adresse https://www.KIRASSUR.com/index.php?token=ae29cf450b6786bf87 permet de charger l'application gérant le dossier et les sinistres associés à ce contrat en cliquant sur le bouton « AUTOMOBILE – TOUS RISQUES – OPTIMALE »

Document B6 : Extrait du code PHP exécuté par la racine de « www.KIRASSUR.com »

```
1. <?php
2. require("./autoload/autoload.php");
3. [...]
4. // vérifie l'existence d'une valeur 'token' dans $_GET
5. if (filter_has_var(INPUT_GET,"token")) {
6.    // filtrage de la variable externe $_GET["token"]
7.    $a_token = filter_input(INPUT_GET,"token", FILTER_SANITIZE_STRING);
8.  $enrToken = $dao->tokenCharger($a_token);
9.    if (!empty($enrToken))
10. switch ($enrToken['utilisation']) {
11.   case 1 :
12.     $vue->afficherAccueilDossier();
13.     break;
14.   case 2 :
15.     [...]
16.     break;
17.  }
18. }
```



Précisions :

- **$vue**

Objet d'affichage des vues.

- **$vue->afficherAccueilDossier() ;**

Méthode générant les vues d'accueil d'un dossier. Par exemple, elle peut fournir le corps de cette page :

Document B7 : Extrait de la classe « DAO.class.php »

```
1.  <?php
2.  class DAO
3.  {
4.    private PDOStatement $pdo = null;
5.    //Constructeur de la classe DAO
6.    function __construct()
7.    {
8.    // Instanciation du pilote de manipulation de la base de données.
9.    // ENV::CONNEXION est une constante venant d'un fichier de configuration
10.     $this->pdo = new PDO(ENV::CONNEXION);
11.   }
12.   function cleAPIVerifierValidite(string $cleAPI): bool
13.   {
14.    $requete = $this->pdo->prepare("
15.      SELECT id
16.      FROM Partenaire
17.      WHERE cleAPIValide = :param1");
18.    $requete->bindParam("param1", $cleAPI, PDO::PARAM_STR);
```

```
19.    $resultat = $requete->execute();
20.    $partenaire = $requete->fetch(PDO::FETCH_ASSOC);
21.    return $partenaire != null
22.   }
```

```
23.  function tokenCharger(string $token_val): ?array
24.  {
25.   $requete = $this->pdo->prepare("
26.     SELECT valeur, idContrat, utilisation , dateHeureLimite
27.     FROM Jeton WHERE valeur = '$token_val' ");
28.   $requete->execute();
29.   $token = $requete->fetch(PDO::FETCH_ASSOC);
30.   if (!empty($token))
31.     if (!verifierDate($token['dateHeureLimite']))
32.       $token = null;
33.   return $token;
34.  }
35.  function tokenEnregistrer(
36.     string $token,
37.     string $idContrat,
38.     int $usage,
39.     int $dureeEnSec
40.   ): bool {
41.   $dateHeureFin = date_create(date('Y-m-d H:i:s')); //Crée une date au format Année-Mois-
     Jour Heure:Minutes:seconde à partir de la date du jour
42.   //Ajoute $dureeEnSec à $dateHeureFin
43.   date_add($dateHeureFin,
          date_interval_create_from_date_string($dureeEnSec . ' seconds'));
44.   //Formate la date pour son insertion en base de données
45.   $dateHeureFinBdd = date_format($dateHeureFin, 'Y-m-d H:i:s');
46.   $requete = $this->pdo->prepare("
47.     INSERT INTO Jeton (`valeur`, `idContrat`, `utilisation `,
48.                                  `dateHeureLimite`)
49.     VALUES (:param1, :param2, :param3, '$dateHeureFinBdd')");
50.   $requete->bindParam("param1", $token, PDO::PARAM_STR);
51.   $requete->bindParam("param2", $idContrat, PDO::PARAM_STR);
52.   $requete->bindParam("param3", $usage, PDO::PARAM_INT);
53.   $resultat = $requete->execute();
54.   return $resultat;
55.  }
56. }
```

public PDO::prepare (string $requeteSQL): PDOStatement

Prépare une requête *SQL* prête à être exécutée par la méthode PDOStatement::execute().

public PDOStatement::bindParam (string|int $parameter, mixed &variable): bool

Lie un paramètre (par exemple :param1 dans la requête préparée) à une variable. Elle ne sera évaluée et positionnée dans la requête qu'au moment de l'appel à la méthode PDOStatement::execute(). Retour : *true* en cas de succès ou *false* si une erreur survient.

public PDOStatement::execute (): bool

Exécute une requête préparée. Retour : *true* en cas de succès ou *false* si une erreur survient.

public PDOStatement::fetch (int $fetchStyle): mixed

Récupère une ligne depuis un jeu de résultats associé à l'objet PDO *$fetchStyle* détermine la façon dont PDO retourne la ligne. Dans notre cas, PDO::FETCH_ASSOC indique que le résultat prendra la forme d'un tableau associatif.

En cas de succès, la méthode retourne une variable contenant un enregistrement.

En cas d'échec, la méthode retourne *false*.

Document B8 : Base de données de KIRASSUR (extrait concernant les jetons)

Remarque : la base de données des contrats partenaires est différente de celle des contrats Kirassur.

Schéma relationnel de la base de données sous forme textuelle :

Jeton (valeur, idContrat, utilisation, dateHeureLimite)

valeur : clé primaire la valeur du jeton (ou *token*) en string

idContrat : clé étrangère en référence à id de Contrat

Précisions :

- valeur : la valeur du jeton (ou *token*) sous forme de chaîne de caractères ;
- dateHeureLimite(date/heure) : indique le jour et l'horaire d'invalidité du jeton ;
- utilisation (entier) : indique l'usage prévu pour ce jeton (1 : connexion à une prestation intégrée, 2 : connexion par un courriel d'invitation…).

Partenaire (id, titre, adresse, cleAPIValide)

id : clé primaire

Contrat (id, dateSignature, dateDebut, dateEcheance, idAssure, idPartenaire)

id : clé primaire

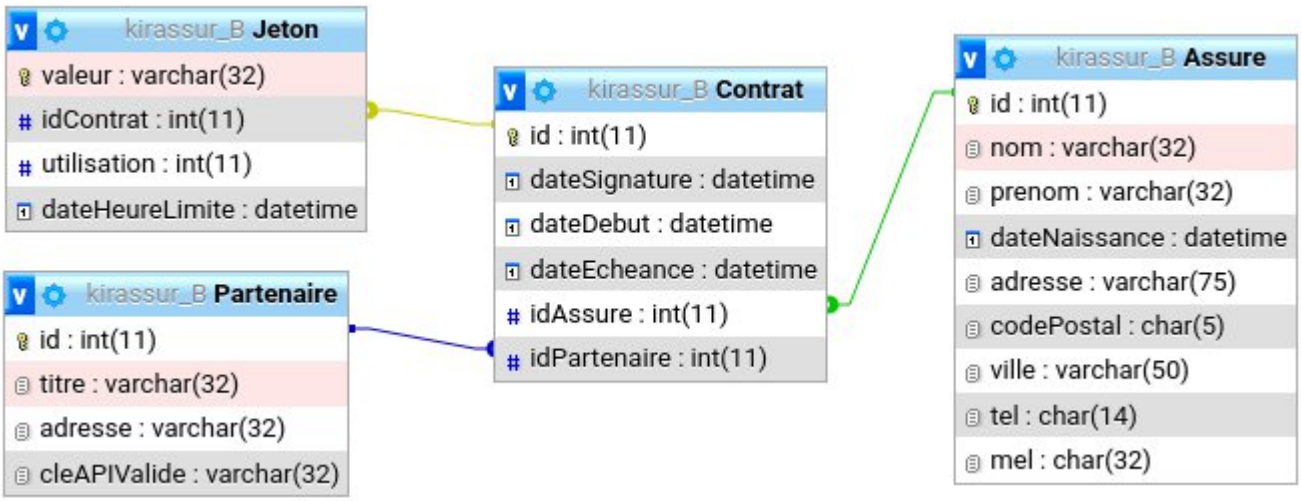
idAssure : clé étrangère en référence à id de Assure

idPartenaire : clé étrangère en référence à id de Partenaire(attribut pouvant être *null* si le contrat n'est pas en sous-traitance)

Assure (id, nom, prenom, dateNaissance, adresse, codePostal, ville, tel, mel)

id : Clé primaire

Schéma relationnel de la base de données sous forme graphique :



Document B9 : Contenu de la base de données (extraits concernant les jetons)

Table : Jeton

valeur	idContrat	utilisation	dateHeureLimite
A2E4BC5F908414FC23	10001	1	2022-05-02 04:30:02
F1234790CFAC23F591	10002	1	2022-05-02 04:30:03
…			
0192848569010589CF	10161	1	2022-05-02 04:32:59
CE1AF34069540FCA2	10162	1	2022-05-02 04:33:00

Table : Contrat

id	dateSignature	dateDebut	dateEcheance	idAssure	idPartenaire
10001	17/03/2017	17/03/2017	31/12/2022	17012	1
10002	17/03/2017	01/04/2017	31/12/2022	18082	2
…					
10161	31/03/2017	01/05/2017	30/04/2023	17081	3
10162	01/04/2017	05/04/2017	31/12/2022	8901	1

Table : Partenaire

id	titre	adresse	cleAPIValide
1	lassureurpascher.com	…	ABCD-ABCDEFGHIJ-ABCD
2	Manif	…	1N73-LL1G3NC315-7H34
3	Le Crédit Bisontin	…	B1L1-TY704D4P70-CH4N

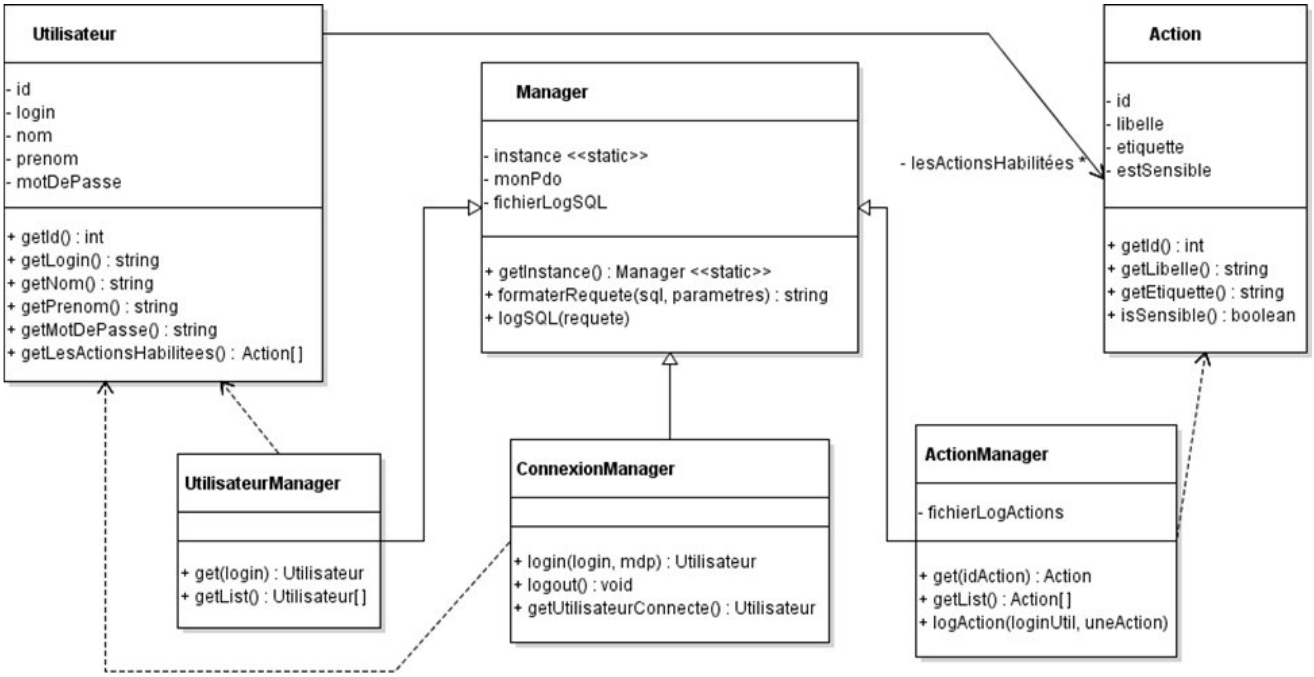
Document B10 : Extrait du fichier journal du serveur Web après simulation de l'incident

192.0.2.3-[02/May/2022:03:30:02 +0100] "GET /index.php?cleAPI=ABCD-ABCDEFGHIJ-ABCD&idContrat=10001"-200- "curl/7.72.0"
192.0.2.3-[02/May/2022:03:30:02 +0100] "GET /index.php?token=A2E4BC5F908414FC23"-200-"curl/7.72.0"
[318 entrées plus loin]
192.0.2.3-[02/May/2022:03:32:59 +0100] "GET /index.php?cleAPI=ABCD-ABCDEFGHIJ-ABCD&idContrat=10161"-200- "curl/7.72.0"
192.0.2.3-[02/May/2022:03:32:59 +0100] "GET /index.php?token=0192848569010589CF"-200- "curl/7.72.0"
192.0.2.3-[02/May/2022:03:33:00 +0100] "GET /index.php?cleAPI=ABCD-ABCDEFGHIJ-ABCD&idContrat=10162"-200- "curl/7.72.0"
192.0.2.3-[02/May/2022:03:33:00 +0100] "GET /index.php?token=CE1AF34069540FCA2"-200- "curl/7.72.0"

Remarque : 192.0.2.3 est l'adresse *IP* externe à l'origine de la requête *HTTP*

Documents dossier C

Document C1 : Extrait du diagramme des classes de la partie Modèle de l'application



Document C2 : Extrait de la classe Manager

```
1.  <?php
2.  require_once("modele/MakeLog.php"); // classe technique de journalisation. Elle fournit la méthode ajouterLog(string $param) qui ajoute la chaine passée en paramètre en fin de fichier MakeLog
3.  class Manager // classe technique permettant d'accéder au SGBD
4.  {
5.    private static Manager $instance ; // (singleton)
6.    private PDOStatement $monPdo;
7.    private MakeLog $fichierLogSQL; // objet fichier de log des requêtes
8.
9.    // Constructeur qui instancie l'ensemble des attributs de la classe
10. private function __construct() {…}
11.
12. // retourne l'instance du Manager
13. public static function getInstance(): Manager {…}
14.
15. // reconstruit la requête préparée sous forme d'une
16. // chaîne de caractères pour être journalisée
17. public function formaterRequete(string $sql, array $parametres = []) {…}
18.
19. // enregistre une requête dans fichierLogSQL
20. public function logSQL(string $requete)
21. {
22.   $horodatage = date('Y-m-d H:i:s', $_SERVER['REQUEST_TIME'])
23.   $this->fichierLogSQL->ajouterLog($horodatage." ".$requete);
24. }
```

```
25. ...
26. }
```

Document C3 : Documentation PHP du tableau $_SERVER

$_SERVER est un tableau contenant des informations créées par le serveur *Web*.

Extraits des indices possibles :

- SERVER_ADDR : L'adresse *IP* du serveur sous lequel le script courant est en train d'être exécuté.
- SERVER_NAME : Le nom du serveur hôte qui exécute le script suivant. Si le script est exécuté sur un hôte virtuel, ce sera la valeur définie pour cet hôte virtuel.
- REQUEST_TIME : Le temps *Unix* du début de la requête.
- QUERY_STRING : La chaîne de requête, si elle existe, qui est utilisée pour accéder à la page.
- REMOTE_ADDR : L'adresse *IP* du client qui demande la page courante.
- REMOTE_HOST : Le nom de l'hôte qui lit le script courant. La résolution *DNS* inverse est basée sur la valeur de REMOTE_ADDR.

Document C4 : Extrait de la classe ActionManager

```
1.  <?php
2.  require_once("modele/Manager.php");
3.  require_once("modele/Action.php");
4.  class ActionManager extends Manager
5.  {
6.    private MakeLog $fichierLogActions; // objet fichier des actions sensibles
7.
8.   // Constructeur qui instancie l'attribut $fichierLogActions
9.   // permettant d'accéder au fichier log
10.  private function __construct() { ... }
11.
12.  public function get(int $idAction): Action
13.  { // instancie l'objet action dont l'id est passée en paramètre
14.    $sql = 'SELECT * FROM Action WHERE id = :param ';
15.    $requete = $this->getInstance()->prepare($sql);
16.    $requete->bindParam('param', $idAction, PDO::PARAM_INT);
17.    $requeteComplete = $this->formaterRequete($sql, [$idAction]);
18.    $this->logSQL($requeteComplete);
19.    $resultat = $requete->execute();
20.    $donnees = $requete->fetch(PDO::FETCH_ASSOC);
21.    return new Action($donnees['id'], $donnees['libelle'],
           $donnees['etiquette'], $donnees['estSensible']);
22. }
23.
24.   // enregistre une action dans fichierLogActions
25.  public function logAction(string $loginUtil, Action $uneAction)
26.  {
27.     // code à compléter
28.  }
29. ...
30. }
```

Document C5 : Format du fichier actionsAAAAMMJJ0000.log que l'on souhaite obtenir

Date heure - adresse IP - login de l’utilisateur effectuant l’action - libellé de l’action journalisée
Date heure - adresse IP - login de l’utilisateur effectuant l’action - libellé de l’action journalisée

Document C6 : Méthode enregistrerActionSensible(Action $uneAction) du contrôleur

```
<?php
require_once("modele/ConnexionManager.php");
require_once("modele/ActionManager.php");

function enregistrerActionSensible(Action $uneAction)
{

  $connexionManager = new ConnexionManager();
  $utilisateurConnecte = $connexionManager->getUtilisateurConnecte();
  // renvoie l'objet utilisateur connecté, null sinon

  $actionManager = new ActionManager();

// code à compléter :
// vérifier le statut de l'action passée en paramètre et, si elle est sensible, appeler la méthode logAction du manager en lui transmettant le login de l'utilisateur connecté (ou 'non connecté') et l'action.
}
```

Document C7 : Schémas conceptuels en cours de modification

Schéma entité-association

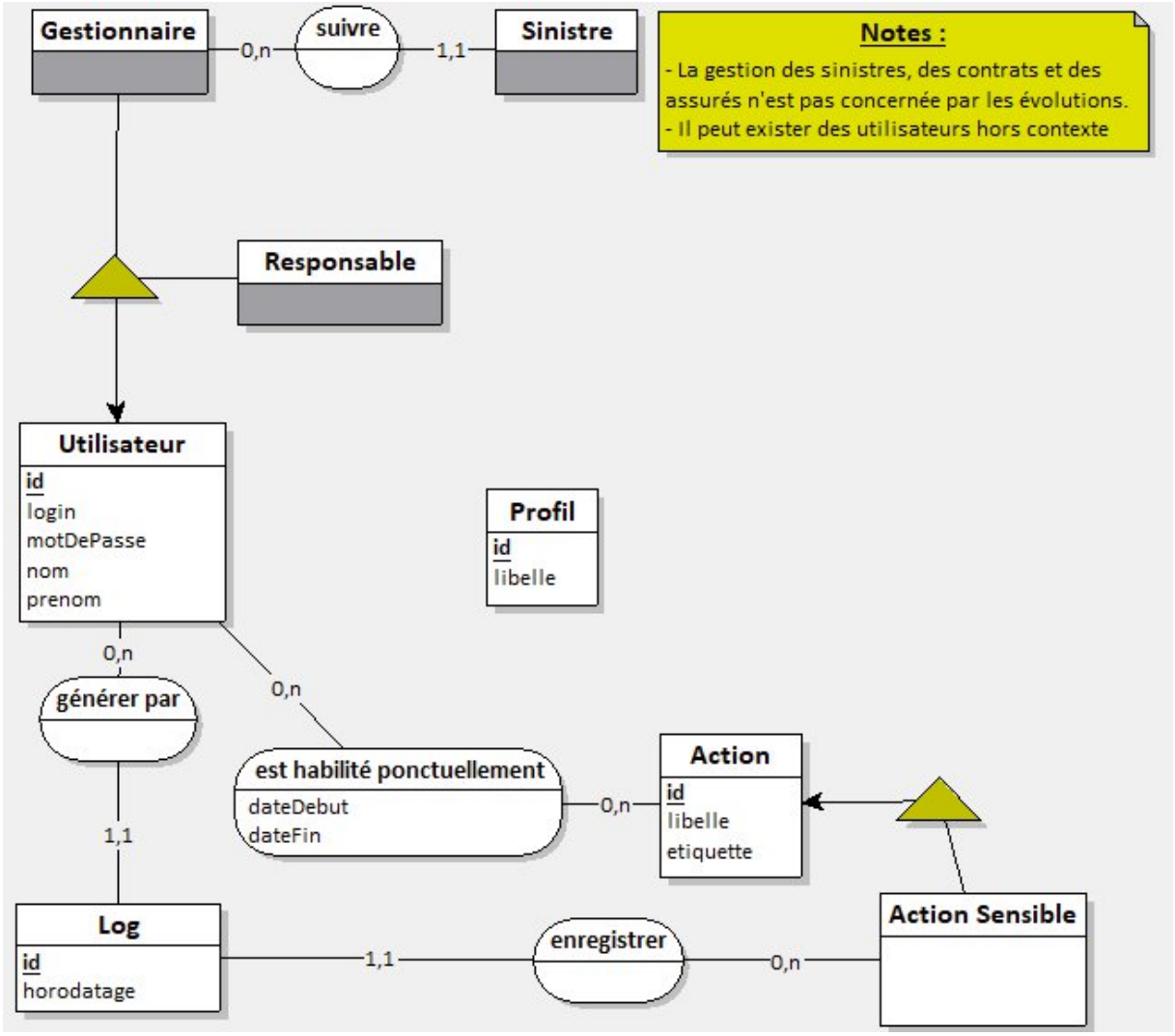
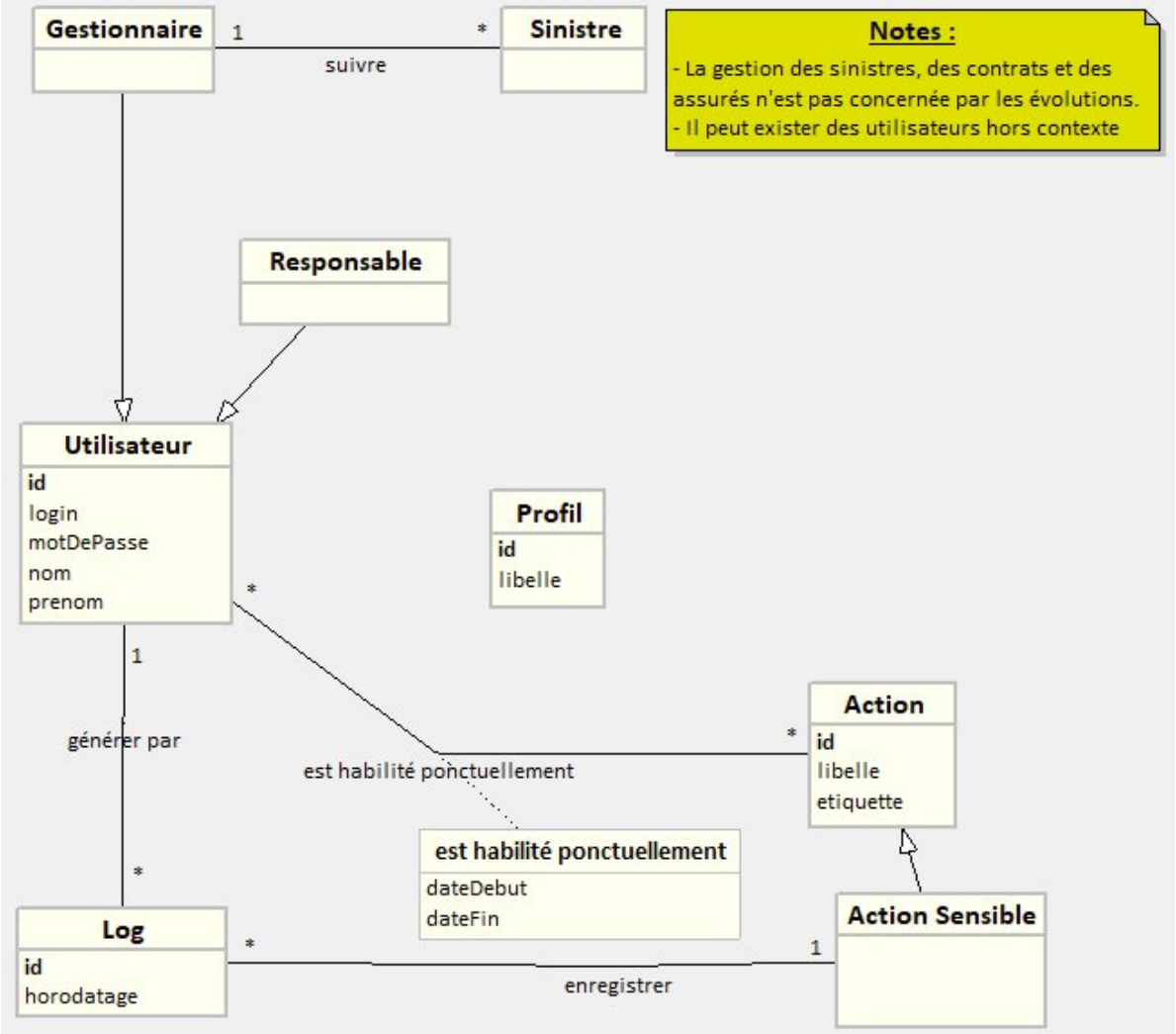


Diagramme de classes


Gestionnaire
1
suivre
*
Sinistre
Notes :
- La gestion des sinistres, des contrats et des assurés n'est pas concernée par les évolutions.
- Il peut exister des utilisateurs hors contexte
Responsable
Utilisateur
id
login
motDePasse
nom
prenom
Profil
id
libelle
*
1
générer par
est habilité ponctuellement
*
Action
id
libelle
etiquette
est habilité ponctuellement
dateDebut
dateFin
*
Log
id
horodatage
*
1
enregistrer
Action Sensible