UFR Sciences et Techniques - **L3 Informatique (IFA)**

Lundi 18 décembre 2023 – de 16h à 18h

Enseignante : V.MARC
Nature de l'épreuve : écrite
UE : I.S.I 1ère session (Partie V.MARC)
Durée conseillée : 1 heure et 40 minutes

DISPOSITIFS ELECTRONIQUES NON AUTORISES
DOCUMENTS NON AUTORISES

Rappel du contexte et des consignes

Les questions de cet examen s'inscrivent dans le contexte du Système d'Information de gestion de jeux sérieux en ligne (« Esc@peWeb ») sur lequel vous avez travaillé.

Il est fortement conseillé de **lire rapidement l'ensemble du sujet et des annexes** avant de commencer à répondre aux questions. Le barème sur 20 points (ramené à la fin sur 16 points) est donné à titre indicatif, et pourra être sujet à des modifications.

Conception et développement d'un système d'information de gestion de jeux sérieux en ligne

1. Vous avez suivi des enseignements d'« Ingénierie des Systèmes d'Information ».
   a) En vous aidant de la définition d'un « Système d'Information » (S.I.), expliquez pourquoi le système informatique « Esc@peWeb » est bien un « Système d'Information » (S.I.).
   b) Puis rappelez ce qu'est l'« Ingénierie ».

/1,5pt

2. « **User story** » et « **Backlog produit** » sont 2 notions différentes mais liées. Expliquez en 4 ou 5 phrases maximum ces 2 notions puis le(s) lien(s) entre une « User Story » et le « Backlog produit » dans le cadre d'une approche AGILE.

/1 pt

Ajout d'une fonctionnalité au cahier des charges de l'application Web « Esc@peWeb »

Le client souhaite l'ajout d'une nouvelle fonctionnalité dans l'espace privé de l'application Web.
Pour s'assurer que tous les membres de l'équipe de développement, dont vous faites partie, travaillent bien sur une base de données identique, le Product Owner impose **une base de données simplifiée** permettant de développer une petite application Web de test pour présenter cette nouvelle fonctionnalité au client.

Description de la nouvelle fonctionnalité :
Un organisateur ('O') connecté a la possibilité de visualiser et dupliquer (copier) les scénarios des autres organisateurs. Quand un organisateur connecté à son espace privé clique dans le tableau récapitulatif de tous les scénarios sur l'icône permettant de copier le scénario d'un autre organisateur, on souhaite qu'une demande d'autorisation pour cette demande soit enregistrée. Ensuite, l'organisateur auteur du scénario pourra donner, ou non, son accord pour la copie demandée. Si l'organisateur auteur du scénario est d'accord, son scénario est copié (et ses étapes aussi) et la copie est attribuée à l'organisateur à l'origine de la demande de copie. Une fois la copie autorisée par l'organisateur auteur, l'organisateur demandeur pourra faire autant de copies du scénario qu'il le souhaitera !

A noter : dans cette application simplifiée de test, on ne gérera pas d'indice pour les étapes des scénarios.

Dictionnaire des données et multiplicités pour cette nouvelle fonctionnalité :

Compte utilisateur : pseudo + mot de passe + nom + prénom + adresse e-mail + statut ('O' / 'A')
Scénario : identifiant + intitulé + description + chemin de l'image + état ('A' / 'D ') + code
Etape : identifiant + intitulé + chemin de l'image + question + réponse + ordre + code

Un organisateur peut ajouter de 0 au minimum à n scénarios maximum.
Un scénario est ajouté par un et un seul compte organisateur.
Un scénario peut rassembler de 0 à n étape(s).
Une étape concerne un et un seul scénario maximum.
Un organisateur peut demander la copie de 0 minimum à n scénarios maximum.
Un scénario peut faire l'objet de 0 au minimum à n demande(s) de copie maximum de la part d'autres organisateurs (*à noter :* on conservera **la date de la demande de copie**, on prévoira **la future réponse (autorisation)**, initialement à *NULL*, de l'auteur du scénario (Acceptée ('A') / Refusée ('R')) et **un compteur**, initialement à 0, **du nombre des copies** faites par le demandeur depuis l'autorisation de copie acceptée).

3. a) Rappelez à quoi sert une « **user story** » AGILE et donnez Donnez le template (la phrase-type) d'une « **user story** » AGILE.

   b) Puis proposez l'énoncé (en utilisant le template / la phrase type) d'au moins 2 « **user stories** » en rapport avec la nouvelle fonctionnalité de copie d'un scénario.

/1,5pt

Modélisation de la base de données

4. Donnez le diagramme UML de classes correspondant à cette nouvelle fonctionnalité.
   On vous demande de faire figurer sur ce diagramme les associations et les classes utiles pour **cette** fonctionnalité, c'est à dire **UNIQUEMENT**, les classes :
- de gestion des comptes des utilisateurs,
- de gestion des scénarios, des étapes et de la copie des scénarios.

*N'oubliez pas de préciser **un titre** pour votre diagramme UML de classes !*

/1,5pts

5. Après avoir vérifié que vous avez bien **nommé vos associations** et précisé **les multiplicités des associations** qui existent entre vos classes, donnez le schéma **relationnel** correspondant à votre diagramme UML de classes.

/1pt

Création des tables de la base de données MariaDB

Pour développer l'application de test, vous devez créer **4 tables** dans votre base de données :

- `t_compte_cpt`
- `t_scenario_sce`
- `t_etape_eta`
- `t_copie_cop`

6. Donnez le code SQL permettant de créer la table « `t_copie_cop` », sans oublier la(les) contrainte(s) de clé **(cf Annexe 1).**

Pour information, les colonnes de cette table sont : `cop_date_demande, cop_autorisation, cop_compteur, sce_id, cpt_pseudo`.

/1,5pt

7. Une de vos collègues prévoit d'exécuter, dans l'onglet SQL de phpmyadmin, le code SQL suivant :

```
INSERT INTO t_copie_cop (sce_id, cpt_pseudo, cop_date_demande, cop_autorisation, cop_compteur) VALUES (2,'mdurand',CURDATE(), NULL, 0)
INSERT INTO t_copie_cop (cop_autorisation, cop_compteur, cop_date_demande, sce_id, cpt_pseudo) VALUES (NULL, 0,CURDATE(),2,'mdurand');
```

Avant d'exécuter son code SQL, elle vous demande de le vérifier et de lui donner votre avis. Que lui répondez-vous ? **Justifiez et expliquez.**

/1,5pt

Requêtes de recherche et de manipulation des données de la base de données MariaDB

8. Donnez **la description textuelle** de la requête SQL suivante :

```
SELECT sce_intitule, SUM(cop_compteur) AS n
FROM t_copie_cop JOIN t_scenario_sce USING (sce_id)
GROUP BY sce_id ORDER BY n DESC LIMIT 2;
```

9. Donnez les **requêtes SQL** permettant
   a) de donner les intitulés et les identifiants des scénarios qui n'ont pas encore d'étape **et** qui n'ont pas encore fait l'objet de demande de copie pour le moment,

   b) de supprimer le scénario d'identifiant « 17 ».

/2,5pts

Code SQL/PSM pour manipuler les données de la base MariaDB

Conseil : pour les questions suivantes, cf Annexe 4 et Annexe 5.

Suite à l'acceptation d'une demande de copie d'un scénario (autorisation de copie acceptée ('A') par l'auteur du scénario original), on souhaite effectuer la copie d'un scénario et de ses étapes grâce

- à **un trigger** SQL/PSM nommé *copier* (code SQL/PSM **à déterminer**),
- à **une procédure** SQL/PSM nommée *dupliquer_scenario()* (code SQL/PSM **à déterminer**),
- et à **une procédure** SQL/PSM nommée *copier_etapes_scenario()* (procédure fournie et **à appeler uniquement**).

Copie du scénario « 2 » = récupération des données + Insertion d'un nouveau scénario « 8 »

2	Les légumes	A	DfgT67UOpO	legumes.png	dd29	
8	Les légumes	A	a7f2c06c81	legumes.png	tuxie	

Image 1: Copie par "tuxie" du scénario "2" créé par l'organisateur "dd29" qui a donné l'autorisation de le copier

4	Couleur	A	GtPkUjBS	Quelle est la couleur d'un petit pois ?	verte	1	2
5	Légume et fruit !	A	577KyT45	Quel légume est biologiquement un fruit ?	tomate	2	2
12	Couleur	A	GtPkUjBS	Quelle est la couleur d'un petit pois ?	verte	1	8
13	Légume et fruit !	A	577KyT45	Quel légume est biologiquement un fruit ?	tomate	2	8

Image 2: Copie des étapes du scénario "2" (Code SQL/PSM fourni)

10. a) Donnez **la** requête SQL qui **met à jour** dans la table « `t_copie_cop` » le compteur de copies (à mettre à 1) et l'autorisation de copier le scénario « **3** » qui est

acceptée ('A') pour l'organisateur de pseudo « **mdurand** » qui en a fait la demande.
b) Donnez le code SQL/PSM d'une procédure *dupliquer_scenario()* qui **insère** dans la table « `t_scenario_sce` » une nouvelle ligne dont l'identifiant est auto-incrémenté qui est une copie des informations générales du scénario à copier (son intitulé, son état et le nom de son image). La procédure doit prendre **en paramètres 3 valeurs** : le pseudo de l'organisateur qui fait la copie, le code du scénario à copier (10 caractères) et un nouveau code de scénario (10 caractères) pour la nouvelle ligne à ajouter dans la table des scénarios (cf description page 2).

c) Puis donnez le code d'un **trigger** (déclencheur) nommé *copier* qui **AVANT** la modification d'une ligne de la table « `t_copie_cop` » (passage de *NULL* à 'A' pour la colonne autorisation **ou** incrémentation du compteur de copies) :
- **génère** un nouveau code de scénario aléatoire de 10 caractères (cf Annexe 4),
- **récupère** le code du scénario concerné par la demande de copie, à partir de son identifiant,
- **appelle** la procédure SQL/PSM créée en 10b) qui copie la ligne du scénario,
- puis **appelle** la procédure SQL/PSM *copier_etapes_scenario()* que l'on vous fournit (cf ci-dessous) qui copie les étapes du scénario original, c'est à dire qui crée de nouvelles lignes dans la table « `t_etape_eta` », et les associe au nouveau scénario créé dans la table « `t_scenario_sce` » par la procédure de la question 10b).

Voici un extrait du code SQL/PSM de la procédure *copier_etapes_scenario()* :

```
DELIMITER $$
CREATE PROCEDURE copier_etapes_scenario (IN CODE1 CHAR(10),IN CODE_COPIE CHAR(10))
BEGIN
-- …
END$$
DELIMITER ;
```

A noter : on demande juste **d'utiliser** la procédure *copier_etapes_scenario()* dans le trigger, on ne demande pas le code SQL/PSM de cette procédure !

/4pts

Code HTML / PHP / SQL de la petite application Web de test

Pour travailler sur la nouvelle fonctionnalité, vous commencez à développer avec CodeIgniter une petite application Web de test qui permet à un organisateur connecté de lister tous **ses** scénarios qui font l'objet d'une demande de copie en attente (autorisation à *NULL* pour le moment).

Voici les éléments que vous avez déjà mis en place :

1er élément : URL de la page Web affichant les demandes de copie en attente pour les scénarios de l'utilisateur connecté
https://obiwan.univ-brest.fr/~escapeweb/index.php/**scenario/lister_demandes**

2ème élément : nom du fichier de la **view** créé pour afficher la liste des scénarios de l'utilisateur connecté concernés par une demande de copie en attente

affichage_demandes.php

3ème élément : corps de la fonction membre du « **Model** » permettant de récupérer, pour un utilisateur particulier dont le pseudo est passé en paramètre, les intitulés des scénarios, les dates de demande de copie associées et les utilisateurs à l'origine des demandes.

```
public function check_demandes($login)
{
//...
}
```

4ème élément : quand on veut préciser dans quelle table on recherche la valeur d'une colonne spécifique, on peut préfixer le nom de la colonne par le nom de la table.

```
SELECT t_scenario_sce.cpt_pseudo AS auteur
FROM t_scenario_sce
WHERE sce_id=3;
```

11. En vous aidant de l'**Annexe 2 et de l'Annexe 3** et en vous basant **OBLIGATOIREMENT sur les éléments déjà mis en place**,

a) **donnez le code PHP** de la fonction membre de la classe du « **Model** » pour pouvoir, dans la base de données MariaDB, récupérer les intitulés des scénarios (+ date de demande de copie + pseudo de l'organisateur demandant la copie) de l'utilisateur connecté qui font l'objet d'une demande de copie en attente (autorisation à *NULL* pour le moment dans la table de gestion des copies).

b) **Donnez aussi le code PHP** de la fonction membre du « **Controller** » utilisant la fonction de votre réponse à la question précédente 11)a) et chargeant 3 « **Views** », une pour le haut de la page, une pour l'affichage des scénarios, des dates de demande associées et des pseudos des organisateurs demandant l'autorisation de faire des copies et une pour le bas de la page Web.

c) Puis proposez **enfin le code PHP et HTML** du fichier « **View** » qui permet d'afficher l'un sous l'autre, sans mise en forme, les intitulés des scénarios de l'utilisateur connecté qui font l'objet d'une demande de copie en attente (+ date de demande de copie + pseudo de l'organisateur ayant fait la demande). S'il n'y a pas de résultat, un message « Aucune demande de copie en attente pour vos scénarios ! » doit s'afficher.

/4pts

ANNEXES

Annexe 1 : SQL-DDL (DROP, RENAME, ALTER)

DROP

Suppression d'une table

```
DROP [TEMPORARY] TABLE [IF EXISTS]
  [nomBase.] nomTable1 [,[nomBase2.] nomTable2,...]
  [RESTRICT | CASCADE] ;
```

Exemple	*Description*
`DROP TABLE CLIENT ;`	Supprime la table CLIENT

RENAME

Modification du nom d'une table

```
RENAME TABLE nomTable1 TO nouveau_nomTable1
    [, nomTable2 TO nouveau_nomTable2] … ;
```

Exemple	*Description*
`RENAME TABLE COMMANDE TO PANIER ;`	Donne le nom PANIER à la table COMMANDE

ALTER

Modifications structurelles d'une table (ALTER TABLE)

```
ALTER TABLE [ IF EXISTS ] nomTable
ADD [COLUMN] nomColonne |
DROP [COLUMN] nomColonne |
ALTER [COLUMN] nomColonne SET DEFAULT <valeur par défaut> |
CHANGE [COLUMN] nomColonne nouveauNomColonne <définition de la colonne> |
MODIFY [COLUMN] nomColonne <définition de la colonne> |
ADD CONSTRAINT <contrainte de table> |
DROP CONSTRAINT nomContrainte {RESTRICT | CASCADE};
```

Exemple	*Description*
`ALTER TABLE CLIENT` `DROP COLUMN PRENOM ;`	Modifie la table CLIENT en supprimant la colonne PRENOM.
`ALTER TABLE T MODIFY A INT NOT NULL AUTO_INCREMENT PRIMARY KEY;`	Modifie la table T en ajoutant une contrainte de clé primaire sur A (+ auto_incrémentation).
`ALTER TABLE CLIENT ADD CONSTRAINT FK_CLI_PER FOREIGN KEY (PER_ID) REFERENCES PERSONNE (PER_ID) ;`	Modifie la table CLIENT en ajoutant une contrainte de clé étrangère sur PER_ID.

Annexe 2 : Documentation CodeIgniter → Extrait de « Generating Query Results »

Extrait du manuel utilisateur en ligne de CodeIgniter :
http://www.codeigniter.com/user_guide/database/results.html

There are several ways to generate query results:

getNumRows() : This method returns the number of rows returned by the query.

```
$query = $db->query('SELECT * FROM my_table');

echo $query->getNumRows();
```

getResultArray() : This method returns the query result as a pure array, or an empty array when no result is produced. **Typically you'll use this in a foreach loop**, like this:

```
$query = $db->query('SELECT name, title, email FROM my_table');

foreach ($query->getResultArray() as $row) {
    echo $row['title'];
    echo $row['name'];
    echo $row['email'];
}
```

getRow() : This method returns a single result row. **If your query has more than one row, it returns only the first row.** The result is returned as an object. Here's a usage example:

```
$query = $db->query('SELECT name, title, email FROM my_table');

$row = $query->getRow();
if (isset($row)) {
    echo $row->title;
    echo $row->name;
    echo $row->email;
}
```

Annexe 3 : fichiers CodeIgniter MVC pour afficher les nom/prénom d'un utilisateur dont on connaît le pseudo (ex : « mdurand »)

```
class Compte extends BaseController {
…
public function afficher_compte($login=FALSE){
      if ($login==FALSE) {return redirect()->to('/');}
      else{
        $data['titre'] = 'Détails du compte sélectionné :';
        $data['cpt'] = $this->model->get_compte($login);
        return view('templates/haut', $data)
            . view('affichage_compte')
            . view('templates/bas');
       }
    }
}
```

Extrait du fichier Compte.php

```
class Db_model extends Model{
…
public function get_compte($login){
   $resultat = $this->db->query("SELECT cpt_nom,cpt_prenom FROM t_compte_cpt
   WHERE cpt_login='".$login."';");
   return $resultat->getRow();}
}
```

Extrait du fichier Db_model.php

```
<h1>Espace privé, bienvenue </h1>
<?php
      $session=session();
      echo $session->get('user');
?>
<h1><?php echo $titre;?></h1>
<?php
      echo $cpt->cpt_nom;
      echo(" -- ");
      echo $cpt->cpt_prenom;
?>
```

Fichier affichage_compte.php

→ URL : http://obiwan.univ-brest.fr/~codeigniter/index.php/**compte/afficher_compte/mdurand**

Annexe 4 : Quelques fonctions MariaDB

Fonctions MariaDB de gestion du temps	Description
ADDDATE(date,n)	Ajoute n jours à une date de type DATE ou DATETIME
DATEDIFF(date1,date2)	Détermine un nombre entier de jours entre 2 dates
CURDATE()	Retourne la date courante au format 'AAAA-MM-JJ' (DATE) ou AAAAMMJJ (INT)

Fonctions MariaDB de gestion de chaînes	Description
CONCAT_WS(separateur,c1,c2,...)	Retourne une chaîne constituée de la concaténation des chaînes séparées par le séparateur précisé
CONCAT(c1,c2)	Retourne la chaîne correspondant à la concaténation des deux chaînes spécifiées
RANDOM_BYTES(length)	Retourne une chaîne de length octets générés aléatoirement. Ex : `SELECT RANDOM_BYTES(8);`

Fonctions MariaDB de groupe	Description
COUNT({* | [DISTINCT] expr })	Retourne le nombre de valeurs de l'expression expr parmi les lignes trouvées par un SELECT
MAX([DISTINCT] expr)	Retourne la valeur maximale de expr
SUM([DISTINCT] expr)	Retourne la somme de expr

Annexe 5 : Eléments du langage SQL/PSM

Code SQL/PSM	Description

BEGIN
... déclarations de variables et code
END;
	Bloc SQL/PSM

DECLARE nomVariable typeVariable	Déclaration d'une variable
SET nomVariable = valeur;	Affectation d'une variable
IF condition **THEN** Liste d'instructions 1 ; [**ELSEIF** condition **THEN** liste d'instructions 2 ;] [**ELSE** liste d'instructions 3 ;] **END IF;**	Structure conditionnelle **IF**
CASE expression **WHEN** expression1 **THEN** Liste d'instructions 1 ; ... **ELSE** liste d'instructions n ; **END CASE;**	Structure conditionnelle **CASE**
CREATE [OR REPLACE] PROCEDURE nom_procédure [(noms_paramètres {**IN** | **OUT** |**IN OUT**} types)] Bloc SQL/PSM	Procédure
CREATE [OR REPLACE] FUNCTION nom_fonction [(noms_paramètres types)] **RETURNS** type Bloc SQL/PSM avec **RETURN**(valeurRetour); avant END;	Fonction
CREATE TRIGGER nom_trigger **{BEFORE | AFTER }** **{DELETE | INSERT | UPDATE} ON** table **FOR EACH ROW** Bloc SQL/PSM	Trigger (déclencheur)