1. **Création du projet par partie (cf.2)**
2. **PLANIFICATION DU PROJET (BENJAMIN & MATTHIEU)**
   1. Planification des tâches ……………………………………………………………………… 2
   2. Répartition des tâches ……………………………………………………………………… 3
   3. Diagramme de Gantt ……………………………………………………………………… 4
3. **BASE DE DONNEES (BENJAMIN)**
   1. Définir la base de données ……………………………………………………………………… 4
      1. Diagramme entité association …………………………………………………………. 4
   2. Optimisation de la base de données ………………………………………………………… 5
   3. Exécution de la base données (Test) …………………………………………………………. 5
   4. Accessibilité de l’administration ………………………………………………………… 7
   5. Liaison avec le site Web (autonome) …………………………………………………………. 7
4. **PROGRAMMATION (MATTHIEU)**
   1. Réaliser le programme principal ………………………………………………………… 7
   2. Définir les fonctions …………………………………………………………………………………. 8
      1. Fonction « PrisePhoto » ……………………………………………………………………… 8
      2. Fonctions « Serveur/Client » ………………………………………………………… 9
      3. Fonction « AllumerLED » ……………………………………………………………………… 9
      4. Mise à jour de la base de données ………………………………………………………… 10
   3. Mise en service du Raspberry …………………………………………………………. 10
   4. Prise de photo depuis le site Web …………………………………………………………. 11
5. **SITE WEB (CLARA)**
   1. Réalisation de la page de connexion ………………………………………………………… 12
      1. Design et mise en page ……………………………………………………………………… 12
      2. Fonctionnement des utilisateurs authentifié ………………………………… 12
      3. Liaison avec la base de données ………………………………………………………… 12
   2. Réalisation de la page d‘accueil …………………………………………………………
      1. Affichage des photos ………………………………………………………………………
      2. Barre des tâches ………………………………………………………………………………….
      3. Design de la page ………………………………………………………………………
      4. Fonctionnalités des administrateurs …………………………………………….
   3. Relation avec la base de données …………………………………………………………
      1. Mise à jour des mots de passe …………………………………………………………
6. **Réalisation du projet complet sur Raspberry (cf. 11)**
7. **REGROUPEMENTS (BENJAMIN & MATTHIEU)**
8. Base de données et Site Web ………………………………………………………………
9. Installation de la base de données ………………………………………………………………
10. Mise en œuvre du site internet ………………………………………………………………
11. Lancement du test final ………………………………………………………………
12. **Création du projet par partie**
13. **PLANIFICATION DU PROJET** 
    1. **Planification des tâches**

Nous avons tout d’abord listé toutes les tâches que nous avons pu déjà prévisualiser le jour de la réception du matériel.

Une image contenant table

Description générée automatiquement

* 1. **Répartition des tâches**

La répartition des tâches est très importante pour le bon déroulement des tâches. Nous avons donc décider de nous attribuer des rôles selon les parties ou bien même les tâches. Concepteur, Réalisateur et Testeur, seront les rôles répartis sur le groupe afin d’avoir une cohérence entre le temps de travail répartie de chacun.

Concepteur : Personne qui va gérer la tâche et aider le Réalisateur.

Réalisateur : C’est la personne qui va programmer, designer et régler les problèmes.

Testeur : Il s’agit de ce qui vont tester et effectuer un rapport de situation au Réalisateur.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | Planification du projet | Benjamin | Matthieu | Clara |
| 1.1 | Rapport | R | R | R |
| ↳ 1.1.1 | Plan | R | C | C |
| 1.3 | Planification des tâches | T | C | R |
| 1.4 | Répartition des tâches | R | R | R |
| 1.4 | Diagramme de Gantt | R | C | C |
| 1.5 | Phase de test du projet | T | T | T |
| 2 | Base de données |  |  |  |
| 2.1 | Définir la base de données | C | T | R |
| ↳ 2.1.1 | Diagramme entité association | R | T | C |
| 2.2 | Optimisation de la base de données | R | T | C |
| 2.3 | Exécution de la base de données | C | T | R |
| 2.4 | Accessibilité de l’administrateur | C | T | R |
| 2.5 | Liaison avec le site web (autonome) | R | T | C / R |
| 2.6 | Fonctions pour simplifier le code | C / R | T | T |
| 3 | Programmation |  |  |  |
| 3.1 | Réaliser le programme principal | C | C / R | C |
| 3.2 | Définir les fonctions | C | C / R | C |
| ↳ 3.2.1 | Fonction « PrisePhoto » | C | C / R | C |
| ↳ 3.2.2 | Fonction Serveur/Client | T | C / R | T |
| ↳ 3.2.3 | Fonction « AllumerLED » (non reçues) | - | C / R | - |
| ↳ 3.2.4 | Fonction « Mise a jour de la BDD » | C | C / R | C |
| 3.3 | Phase de test | T | T | T |
| 4 | Site Web |  |  |  |
| 4.1 | Réalisation de la page de connexion | C | C | R / T |
| ↳ 4.1.1 | Design et mise en page | C | C | R |
| ↳ 4.1.2 | Fonctionnement des utilisateurs authentifié | T | T | C / R |
| 4.2 | Réalisation de la page d‘accueil | R | C | R / T |
| ↳ 4.2.1 | Affichage des photo | R | T | C / T |
| ↳ 4.2.2 | Barre des taches (en haut) | T | C | R |
| ↳ 4.2.3 | Design de la page | C | C | R |
| ↳ 4.2.4 | Fonctionnalités des administrateurs | C | T | R / T |
| 4.3 | Mise à jour des mots de passes | C | T | R / T |
| 4.4 | Création d'un nouvel utilisateur | C | T | R / T |
| 4.5 | Requête pour l'affichage des images | R | T | C |
| 4.6 | Mise en service du Raspberry | C | R | T |
| ↳ 4.6.1 | Installation apache2, PHP | C | R | T |
| ↳ 4.6.2 | Installation phpMyAdmin, Maria DB | C | R | T |
| ↳ 4.6.3 | Mise en lien base de données et site web | C | R | T |

* 1. **Diagramme de Gantt**

Une fois la répartition des tâches faites, il nous suffit d’estimer le temps passé sur les tâches ainsi que leurs dépendances pour avoir une idée de la « Deadline » du projet. Voici le Gantt que nous avons pu réaliser.

*« Diagramme de Gantt* ***FINAL présent dans l’archive associé*** *»*

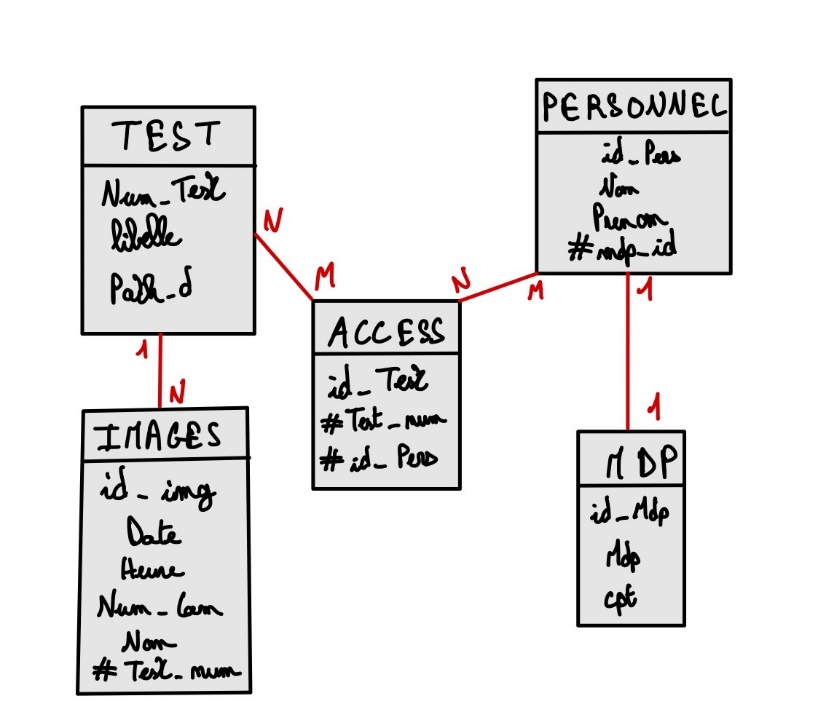
1. **BASE DE DONNEES** 
   1. **Définir la base de données**

Pour définir la base de données, nous avons commencé par définir toutes les informations utiles pour gérer entièrement les données des utilisateurs, des tests et des images. Nous avons alors listé toutes ces données.

Tests : Nom du test, ainsi que son chemin (path).

Personnels : Nom, prénom, mot de passe.

Images : Nom, date, heure, numéro de caméra.

* + 1. **Diagramme entité association**

Durant notre première année de formation, nous avons appris à réaliser un diagramme nous permettant d’initialiser une base de données, celui-ci se nomme « Entité / Association ».

Voici notre diagramme Entité / Association finalisé. Cela représente les tables qui seront dans la base de données avec les données de chacune des tables, ainsi que leurs relations.

**Les relations** sont les traits reliés entre les tables. Ils permettent une certaine liaison entre les tables pour avoir une base de données dynamique et complète.

Le nombre « **1** », « **N** » et « **M** » sont transcrits de part et d'autre des relations. Voici des exemples pour avoir une compréhension plus facile de ces significations.

Exemples : • De la table Images à Test : Il y a un nombre **N** d’images pour **1** test.

• De la table Test et Access : Il y a **N** tests avec **M** accès pour accéder aux tests.

• De la table Access à Personnel : Il y a **N** accès pour plusieurs (**M**) employés.

• De la table Personnel à Mdp : Il y a **1** employé qui a **1** mot de passe.

De nombreux dièses « # » sont devant certains champs des tables. C’est une manière de décrire que le champ est une clé étrangère. C’est-à-dire qu’elle dépend de la clé primaire de la table voisine. Cela permet une certaine relation entre les tables.

* 1. **Optimisation de la base de données**

Après plusieurs utilisations de la base de données, des tests ont été fait pour savoir si la base de données était opérationnelle. On s’est aperçu que certaines relations ne servaient à rien. Nous avons donc refait l’initialisation de la base de données. (*Voir directement le code sql pour la création de la base de données.)*

Le principe était de juste faire 3 tables (Personnel, images et logs). Ces tables sont les seuls nécessaires au bon fonctionnement. Les tables auront toutes les informations utiles pour le site web et les programmes python pour le bon fonctionnement des tâches à venir.

* 1. **Exécution de la base données (Test)**

***Le code SQL final se trouve dans l’archive SAE24\_BMC***

***(Dossier B.B.B.B)***

• Création de la base de données sur PhpMyAdmin (PMA) :

PhpMyAdmin est une application web de gestion de base de données MySQL et MariaDB, réalisée principalement en PHP.

Nous avons alors accédé à PMA grâce à isis (réseau interne de l’IUT) pour le moment, nous devrions tout simplement par la suite installer PMA sur le Raspberry et créer un serveur pour que cela fonctionne de la même manière qu’avec isis.

Pour la création de la base de données tout se passe depuis l’interface graphique de PMA qui est très intuitive. En étant sur isis, nous avons la possibilité d’utiliser une base de données. C’est pour cela que nous avons directement commencé à créer les tables grâce à ces commandes :

Table PERSONNEL :

CREATE TABLE PERSONNEL (

idUser INT PRIMARY KEY NOT NULL AUTO\_INCREMENT,

nom VARCHAR(70) NOT NULL,

prenom VARCHAR(50) NOT NULL,

login VARCHAR(30) NOT NULL,

mdp VARCHAR(30) NOT NULL,

admin BOOLEAN,

cpt\_renouvellement INT NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=10;

Table IMAGES :

CREATE TABLE IMAGES (

idImg INT PRIMARY KEY NOT NULL AUTO\_INCREMENT,

nomImg VARCHAR(70) NOT NULL,

nomTest VARCHAR(70) NOT NULL,

dateImg DATE NOT NULL,

heure TIME NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=10;

Table LOGS :

CREATE TABLE LOGS (

idLog INT PRIMARY KEY NOT NULL AUTO\_INCREMENT,

nomLog VARCHAR(70) NOT NULL,

dateLog DATE NOT NULL,

heureLog TIME NOT NULL,

numUser INT,

numImg INT,

FOREIGN KEY(numUser) REFERENCES PERSONNEL(idUser),

FOREIGN KEY(numImg) REFERENCES IMAGES(idImg)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=10;

Les tests pour la création de la base de données ainsi que les tables sont concluants. Nous avons rencontré quelque difficulté pour la création des clés étrangère mais ce fut très vite réglé.

• Ajouter des données dans les tables :

Si les tables sont bien initialisées alors l’ajout d’enregistrement dans les tables devraient bien se passer. Pour les enregistrements, il est préférable de ne pas mettre des espaces pour éviter toute confusion dans le code, nous en reparlerons plus tard dans la partie du site web. Voici les commandes pour ajouter des enregistrements :

Ajouts des enregistrements :

INSERT INTO PERSONNEL (nom, prenom, login, mdp, admin, cpt\_renouvellement) VALUES

('faretie','benjamin','fb','fb',TRUE,5),

('perre','matthieu','pm','pm',FALSE,30),

('woodman','clara','wc','wc',FALSE,50);

INSERT INTO IMAGES (nomImg, nomTest, dateImg, heure) VALUES

('Photo1','Test1','2022-04-27','19:08:48'),

('Photo2','Test1','2022-05-27','20:08:48'),

('Photo1','Test2','2022-06-27','21:08:48');

INSERT INTO LOGS (nomLog, dateLog, heureLog) VALUES

('Delete Photo 1','2022-07-27','20:00:00'),

('Modification du nom de la Photo 2','2022-08-27','20:00:00'),

('Mot de passe modifier du User 1','2022-09-27','20:00:00');

L’ajout d’enregistrement n’est pas compliqué en matière de code mais il faut être judicieux au niveau des contraintes d’enregistrement. Nous avons essayé de manipuler ces enregistrements avec au début des « nomImg » avec des espaces comme « Image 21 » mais l’espace pose des problèmes dans le code php. Nous en conclurons qu’il ne faut pas mettre d’espaces dans les champs utilisés dans le code php. Pour supprimer un enregistrement, il faut tout simplement utiliser la commande :

DELETE FROM <nom\_table> WHERE <nom\_champ> = <valeur\_champ>

* 1. **Accessibilité de l’administration**

L’accès à la base de données est seulement autorisé aux administrateurs. C’est-à-dire qu’a la création du serveur phpMyAdmin sur le Raspberry, il faut configurer l’identifiant et le mot de passe de l’accès à la base de données. Il sera alors nécessaire de mémoriser les informations de connexions car il y a le risque que tout le monde n’ait plus accès au site à cause d’oubli de mot de passe, alors l’accès à la base de données est plus que nécessaire.

* 1. **Liaison avec le site Web (autonome)**

Le site web étant codé en HTML/CSS et PHP, nous devions nous connecter depuis le code php à la base de données. La commande nécessaire à la connexion :

try {

$bd = new PDO("mysql:host=localhost;dbname=","id","mdp");

$bd->exec('SET NAMES utf8');

}catch (Exception $e){

die("Erreur: Connexion à la base impossible");}

Ce petit bout de code nous permet d’utiliser la base de données définie dans les options de la fonction PDO() (fonction permettant la connexion à la base de données codée en PHP), et si l’accès à la base est refusé cela retourne une erreur indiquant que la connexion à la base de données n‘est pas possible.

Pour conclure la création de la base de données a été un succès et peu de risques ont été découvert, mis à part la perte de mot de passe admin que nous avions déjà anticipé, mais il y a une solution à cela. Il suffirait que l’admin se réfère à l’interface graphique de Phpmyadmin pour ensuite pouvoir retrouver son mot de passe ainsi le modifier. Mais nous en reparlerons dans la partie du site web car le mot de passe sera crypté grâce à la fonction md5() en php. Une fois crypté le mot de passe n’est plus récupérable depuis la base de données, alors nous définirons une méthode pour ce risque dans la partie Web.

1. **PROGRAMMATION**
   1. **Réaliser le programme principal**

***Toutes les fonctions présentées ci-après sont accessibles dans l’archive AE24\_BMC***

***(Dossier Raspberry)***

Ce programme permet au Raspberry de prendre des photos et d’agir comme un serveur de requête. Il respecte donc la condition d’être disponible à n’importe quel moment de la journée, pour cela on automatise le démarrage du fichier. On utilise donc la ressource « Crontab » qui dans notre cas démarre le script au démarrage du Raspberry (Image 1). Pour permettre ensuite de garder le programme ouvert en permanence, on utilise une boucle sans fin (« While True »). Nous disposons également d’un deuxième programme appelant les mêmes ressources pour faire fonctionner la prise de photo depuis le Web.

Une image contenant texte

Description générée automatiquement

Image 1

* 1. **Définir les fonctions**

Plusieurs fonctions sont nécessaires au bon fonctionnement, la 1ere est donc celle qui permet de prendre des photos, la 2nde permet de démarrer un serveur exploitant la fonction de prise de photo, la 3e, elle, teste et reprend une photo en cas de mauvaise luminosité ensuite la fonction nous autorisant à venir communiquer avec le serveur.

Au cours du développement de nouveaux besoins se sont faits, la prise de photo depuis le site internet et la mise a jour de la base de données pour répertorier les images et mettre à jour la date de validité des mots de passe.

* + 1. **Fonction « PrisePhoto »**

Cette fonction est assez simple à mettre en place, il nous suffit de récupérer 2 variables :

* le nom du projet,
* le nom de la photo (non nécessaire selon la provenance de la requête)

Nous avons également besoin de 3 librairies et d’un paquet :

* Le programme FS Webcam
* La libraire os
* La libraire datetime

Pour cela, on utilise le module « FS webcam » qui permet de prendre des photos grâce à des lignes de commandes. Nous devons également pouvoir prendre des photos avec un nombre de caméras variable, ici on suppose que le maximum de caméra connecté est de 4 (4 ports USB sur le Raspberry). Nous avons également besoin de certaines librairies Python : « os » permettant l’envoi de commande depuis le code python au terminal et « datetime » permettant la création du nom de l’image.

*Voie d’amélioration : Créer une boucle au lieu d’un test de condition pour permettre un nombre illimité de caméras connectées et permettre une meilleure lisibilité du code.*

* + 1. **Fonctions « Serveur/Client »**

Après avoir permis l’utilisation de la caméra depuis python, maintenant il nous faut la possibilité de le faire depuis une machine distante. C’est alors que cette fonction prend tout son sens, celle-ci permet donc de communiquer entre un client et le Raspberry. Pour cela, il nous suffit de créer un socket et de récupérer les requêtes émises par le client.

Cette fonction est donc double, une coté serveur (écoute du port) et l’autre du côté du client (envoie sur le port).

Une image contenant table

Description générée automatiquement

* + 1. **Fonction « AllumerLED »**

Cette fonction comme son nom l’indique permet l’allumage de la LED et donc la reprise des photos dans des conditions « meilleures ».

Elle se décompose en 2 principales actions :

* Celle de tester le taux de pixel noir présent sur l’image (50% de pixel noir sur une image de dimensions 1920\*1080px).
* Celle d’allumer la LED et de recapturer les images.

A cause de matériel non fourni, elle n’a pas pu être mise en place. Elle permet quand même de tester le niveau de noir d’une image (non raccordée au programme principal mais le sera avant la présentation).

* + 1. **Mise à jour de la base de données**

Maintenant que nous avons pris en photo et stocké sur le Raspberry l’image, il nous faut donc mettre en lien ces images et projets avec le site. On a donc besoin de 2 fonctions :

* La mise à jour dynamique des mots de passe
* L’ajout automatique des images dans la base de données

La 1ere fonction consiste à envoyer une requête et modifier tous les champs de « cpt\_renouv » (compteur de 90 jours qui permet de vérifier si le mot de passe et encore valable) en enlevant 1 à ces enregistrements par l’appel journalier d’un fichier python via Crontab.

La 2nde fonction, elle, permet d’enregistrer les nouvelles images dans la base de données en envoyant une requête contenant le nom du projet et le chemin absolu de l’image celle-ci est appelé à la suite de la prise de photo.

* 1. **Mise en service du Raspberry**

Maintenant que tout fonctionne, il faut mettre en place le site sur le Raspberry, Nous avons donc besoin de différents paquets pour gérer la partie Web :

* Apache2 (Service Web),
* PHP (Non installé de base dans le paquet Apache2)

et ensuite des paquets de base de données :

* PhpMyAdmin (gestionnaire de base de données avec interface graphique)
* Maria DB (Base de données)

Notre Raspberry disposant maintenant de tous ces paquets, on se rapproche donc de la phase de test globale.

* 1. **Prise de photo depuis le site Web**

Cette fonction permet au site web de communiquer avec les fonctions python cités précédemment à l’image de la fonction Serveur. Ici, le site Web envoie une commande dans le terminal « python3 /home/pi/Downloads/PrisePhotoWeb.py projet » (sujet à évolution) ce qui lance le script qui va donc démarrer la prise de photo par l’appel de la fonction « PrisePhoto » en utilisant les variables passées par le terminal. (Nous utilisons la librairie « sys » pour récupérer nos variables)

1. **SITE WEB**
   1. **Réalisation de la page de connexion**

Dès l’ouverture du site Web via l’adresse A ou 127.0.0, la page de connexion apparaît pour nous permettre d’avoir une certaine restriction selon l’utilisateur qui utilise l’interface du site. Ainsi cela rend la connexion nécessaire à l’accès du reste du site Web.

* + 1. **Design et mise en page**

Pour le design nous avons opté sur un fond d’une couleur similaire au logo Thales pour avoir une unicité des couleurs. Nous avons placé au premier plan au centre de la page le bloc de connexion où il faut renseigner l’identifiant et le mot de passe de l’utilisateur. En cas d’oubli du mot de passe, il faudra systématiquement envoyés un mail à l’administrateur. En cas de mauvais mot de passe ou d’identifiant alors un message d’erreur le signalera…

* + 1. **Fonctionnement des utilisateurs authentifié**

Les utilisateurs authentifiés sont ceux qui ont des identifiant et mot de passe pour se connecter au site, c’est-à-dire qu’ils sont enregistrés dans la base de données. Il sera donc possible de visualiser les photos par la suite, en étant authentifié. Ainsi il n’est pas nécessaire d’être authentifié pour prendre une photo car la fonction est disponible sur la page de connexion.

**Exigences accomplîtes :**

Reference PHOTO\_ATB-Securite-100

L’utilisation de l’interface de l’application PHOTO\_ATB doit être sécurisé par un accès de type login/password.

Reference PHOTO\_ATB-Securite-110

Le déclenchement d’une prise de photo par l’application PHOTO\_ATB ne doit pas nécessiter l’utilisation d’un compte / mot de passe.

Reference PHOTO\_ATB-Securite-130

L’application PHOTO\_ATB doit limiter l’accès aux différentes fonctionnalités en fonction du profil concerné :

* Administrateur,
* Operateur.
  + 1. **Liaison avec la base de données**

La connexion de la base de données depuis le site web est nécessaire au bon fonctionnement de notre site web. Nous nous sommes mis dans la même situation que le Raspberry, en local. Alors l’accès à la base de données se fait tout simplement avec un petit script de 2-3 lignes pour permettre d’utiliser les tables de la base de données.

* 1. **Réalisation de la page d‘accueil**
     1. **Affichage des photos**

**Exigences accomplîtes :**

Reference PHOTO\_ATB-Accès-600

L’application PHOTO\_ATB doit permettre à un utilisateur d’accéder aux photos en les triant par date et par banc.

\*

Reference PHOTO\_ATB-Accès-610

L’application PHOTO\_ATB doit restreindre l’accès de certains utilisateurs à certains bancs en fonction de leur droit.

* + 1. **Barre des taches**
    2. **Design de la page**
    3. **Fonctionnalités des administrateurs**
  1. **Relation avec la base de données**
     1. **Mise à jour des mots de passe**
     2. **Création d'un nouvel utilisateur**
     3. **Requête pour l'affichage des images**

1. **Réalisation du projet complet sur Raspberry**
2. **REGROUPEMENTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Historique / Procédure des Test** | | | | |
| Nombre de Tests : 5 | | | | Validation finale : |
| Phases requissent : • Base de données fonctionnel  • Site Web Accessible sur le Raspberry  • Raspberry opérationnel  • Fonctions du Raspberry initialisé | | | Auteur : Benjamin  Membres :  Matthieu  Clara | Qu'est-ce que la validation subjective ? |
| N° | Description | Testeurs | État | Remarques |
| 1 | Est-ce que la base de données est adapter et fonctionnel au Site Web ? | Clara  Benjamin | Qu'est-ce que la validation subjective ? | Le Site Web n’a pas de difficulté à s’adapter avec la base de données. Ainsi le site et la base de données est opérationnel. |
| 2 | Implémentation de la base de données sur le Raspberry.  Préalable : Installation du serveur phpMyAdmin. (validé) | Matthieu  Benjamin | Qu'est-ce que la validation subjective ? | Aucuns soucis, en passant par des requêtes SQL. |
| 3 | Implémentation du site Web sur le Raspberry.  Préalable : Installation du serveur apache2 et de php. (validé) | Matthieu  Clara | Qu'est-ce que la validation subjective ? | L’implémentation est un succès, le design est à revoir mais rien de grave. |
| 4 | Essayer le site internet depuis le Raspberry.  Se connecter, visualiser des photos déjà retranscrite dans la bdd, créer un utilisateur, changer un mot de passe | Matthieu  Clara  Benjamin | Qu'est-ce que la validation subjective ? | Tout comme en local, sur le Raspberry tout fonctionne avec bien sur toutes les configurations à refaire dans le code. Par exemple :   * Ecrire les bons chemins (path) des images. |
| 5 | Prendre une photo depuis le site Web (1)  Préalable : Créer le bouton pour prendre la photo. (validé) | Matthieu  Clara  Benjamin | Qu'est-ce que la validation subjective ? | Problème de synchronisation du code python et le code php. Transfère de variables non prises en compte.   * Solution 1 : Trouver un moyen d’envoyer des variables du Site Web et les récupérer pour les utiliser dans le code python. * Solution 2 : Écrire du code python depuis le site web. |
| 6 | Prendre une photo depuis le site Web (2)  « Solution 1 » | Matthieu  Clara  Benjamin |  | Le programme python peut récupérer les variables depuis le terminal, il faut essayer d’en envoyer une depuis le site web.   * Sera prêt pour la présentation |
| 7 | Prendre une photo depuis un client distant (programme 1 caméra) | Matthieu  Benjamin | Qu'est-ce que la validation subjective ? | Le poste peut se connecter au Raspberry. La prise de photo depuis un poste distant fonctionne parfaitement bien pour une caméra (ancien programme) |
| 8 | Prendre une photo depuis un client distant  (programme plusieurs caméra) | Matthieu  Benjamin |  | Le programme ne rentre pas dans les conditions if pour prendre les photos.   * Sera prêt pour la présentation |
| 9 | Mise à jour de la base de données | Matthieu  Benjamin |  | La fonction de mise à jour des données fonctionne, il faut adapter la connexion à celle déclaré sur le Raspberry.   * Sera prêt pour la présentation |
| 10 | Mise à jour de la durée de vie des mots de passe | Matthieu  Benjamin |  | Le compteur décrémente comme il faut. Il ne reste qu’à automatiser la tâche pour la rendre autonome et modifier les identifiant de connexion pour permettre la communication avec la base de données du Raspberry.   * Sera prêt pour la présentation |
| Conclusion | | | | |
| Les risques sont :   * Que les tests finaux ne sont pas approuver et accomplit, pour faire le test final. * Fonctionnement de plusieurs caméra.   Le rassemblement de nos partie semble être un succès malgré qui nous reste à configurer certaines liaisons entre le site web et le raspberry. | | | | |