

**BTS SIO – Fiche mission E5**

|  |  |  |  |
| --- | --- | --- | --- |
| Rédacteur(s) | Version | Date | Nb pages |
| Zatric Nihad | 1.1 | 04/04/2021 | 22 |

**CRUD AJAX**

Une image contenant table

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

**SOMMAIRE**

[1 Explication du document au jury 6](#_Toc99538113)

[2 Démarrage du projet 8](#_Toc99538114)

[2.1 Charte de projet 8](#_Toc99538115)

[2.2 Expression fonctionnelle du besoin 9](#_Toc99538116)

[2.3 Contraintes 10](#_Toc99538117)

[2.4 Gestion des droits d’accès 10](#_Toc99538118)

[3 Conception de l’application 11](#_Toc99538119)

[3.1 Schématisation de l'application 11](#_Toc99538120)

[3.2 Choix des technologies employées 12](#_Toc99538121)

[3.3 Présentation des composants logiciels 12](#_Toc99538122)

[4 Conception de la BDD 13](#_Toc99538123)

[4.1 Schéma de la base de données 13](#_Toc99538124)

[4.2 Installation et configuration de la BDD 13](#_Toc99538125)

[4.3 Interaction de l'application avec la base de données 14](#_Toc99538126)

[5 Gestion de la qualité 15](#_Toc99538127)

[5.1 Méthodologie et outil de versioning 15](#_Toc99538128)

[5.2 Gestion des tests de l'application 16](#_Toc99538129)

[5.3 Rédaction de la documentation 19](#_Toc99538130)

[6 Mise en production 19](#_Toc99538131)

[6.1 Description de l'environnement de développement 19](#_Toc99538132)

[6.2 Description de l'environnement de production 20](#_Toc99538133)

[6.3 Etape de mise en production 20](#_Toc99538134)

[6.4 Mise en place de la sauvegarde 20](#_Toc99538135)

[7 Gestion de la maintenance (corrective / évolutive) 21](#_Toc99538136)

[7.1 Mise à jour de la documentation du SI 21](#_Toc99538137)

[7.2 Evaluation de la qualité de l’application 21](#_Toc99538138)

[7.3 Procédure de correction d'un dysfonctionnement 21](#_Toc99538139)

[7.4 Gestion des tests de mise à jour 21](#_Toc99538140)

[8 Bilan du projet 22](#_Toc99538141)

# Explication du document au jury

Le plan de ce document a été conçu en se basant sur la circulaire du BTS SIO disponible sur le site du réseau certa (<https://www.reseaucerta.org/sio/circulaire>). Afin de faciliter la compréhension de ce document, vous trouverez ci-dessous un tableau présentant les liens entre le plan, les compétences de bloc et les indicateurs de performance.

Les compétences sont numérotées en fonction des 3 blocs :

1. Concevoir et développer une solution applicative
2. Assurer la maintenance corrective ou évolutive d’une solution applicative
3. Gérer les données

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Circulaire BTS SIO 2022 | |
| Plan du document | | Compétences de bloc correspondantes | Indicateurs de performance correspondants |
| Démarrage du projet | | | |
|  | Charte de projet | 1.1 Analyser un besoin exprimé et son contexte juridique | La proposition de la solution applicative répond au besoin exprimé dans le cahier des charges |
| Expression fonctionnelle du besoin | - la maquette des éléments applicatifs de la solution respecte les fonctionnalités exprimées ; |
| Contraintes | L’interface utilisateur est mise à jour en respectant les contraintes ergonomiques. |
| Gestion des droits d’accès | Les accès aux données sont contrôlés conformément aux habilitations définies par le cahier des charges. |
| Conception de l'application | | | |
|  | Schématisation de l'application | 1.2 Participer à la conception de l’architecture d’une solution applicative |  |
| 1.3 Modéliser une solution applicative |  |
| Choix des technologies employées | 1.4 Exploiter les ressources du cadre applicatif (framework) |  |
| Présentation des composants logiciels | 1.5 Identifier, développer, utiliser ou adapter des composants logiciels | Le choix des composants logiciels à utiliser et/ou à développer est pertinent. |
| le développement tient compte des préoccupations de développement durable. |
| Des composants logiciels sont adaptés pour améliorer la qualité de la solution applicative. |
| 1.6 Exploiter les technologies Web pour mettre en œuvre les échanges entre applications, y compris de mobilité | Un service Web est exploité pour échanger des données entre applications. |
| Conception de la base de données | | | |
|  | Schéma de la base de données | 3.3 Concevoir ou adapter une base de données | Les données sont modélisées conformément au besoin de la solution applicative. |
| Installation et configuration de la base de données | 3.4 Administrer et déployer une base de données |  |
| Interaction de l'application avec la base de données | 1.7 Utiliser des composants d’accès aux données | Les données persistantes liées à la solution applicative sont exploitées à travers un langage de requête lié à la base de données qui peut être le langage de requête proposé par les échanges applicatifs des technologies Web, un langage de requête présent dans l’outil de correspondance objet-relationnel ou toute autre solution de persistance. |
| 3.1 Exploiter des données à l’aide d’un langage de requêtes |  |
| Gestion de la qualité | | | |
|  | Méthodologie et outil de versioning | 1.8 Intégrer en continu les versions d’une solution applicative | Un outil collaboratif de gestion des versions est utilisé. |
| Gestion des tests de l'application |  | Les tests de non régression sont réalisés. |
| Gestion des tests de l'application | 1.9 Réaliser les tests nécessaires à la validation ou à la mise en production d’éléments adaptés ou développés | Les composants logiciels sont validés par les procédures de tests unitaires et fonctionnels. |
| un document est rédigé pour chaque contexte d’utilisation de l’application et est adapté à chaque destinataire tant par son contenu que par sa présentation ; |
| 1.11 Exploiter les fonctionnalités d’un environnement de développement et de tests |  |
| Rédaction de la documentation | 1.10 Rédiger des documentations technique et d’utilisation d’une solution applicative |  |
| Mise en production | | | |
|  | Description de l'environnement de développement |  | Le choix du type de base de données est pertinent. |
| Description de l'environnement de production |  |  |
| Etapes de mise en production |  | La base de données est opérationnelle et stable dans l’environnement de production. |
|  | L’application développée est opérationnelle conformément au cahier des charges et stable dans l’environnement de production. |
| Mise en place de la sauvegarde |  | Des tests de restauration sont effectués. |
|  | La base de données est sauvegardée selon la planification retenue. |
| Gestion de la maintenance (corrective / évolutive) | | | |
|  | Mise à jour de la documentation du SI | 2.1 Recueillir, analyser et mettre à jour les informations sur une version d’une solution applicative | Les composants logiciels sont documentés de manière à être réutilisés. |
| La documentation technique et d’utilisateurs de la solution applicative sont mises à jour. |
| 2.4 Mettre à jour des documentations technique et d’utilisation d’une solution applicative | - une documentation des versions vient appuyer l’intégration continue ; |
| - les composants logiciels sont documentés de manière à être réutilisés ; |
| Evaluation de la qualité du livrable | 2.2 Évaluer la qualité d’une solution applicative |  |
| Procédure de correction d'un dysfonctionnement | 2.3 Analyser et corriger un dysfonctionnement | Le dysfonctionnement de la solution existante est corrigé selon les procédures en vigueur et dans les délais. |
| Gestion des tests de mise à jour | 2.5 Élaborer et réaliser les tests des éléments mis à jour | Les composants logiciels adaptés et/ou corrigés sont validés par les procédures de tests unitaires et fonctionnels. |
| Bilan du projet | | | |
|  | Bilan |  |  |

# Démarrage du projet

## Charte de projet

|  |
| --- |
| **Type de mission** |
| Mission effectuée dans le cadre d’un projet personnel. |
| **Contexte** |
| Dans le cadre d’un projet personnel, j’ai réalisé un modèle de CRUD (create, read, update, delete) en utilisant la méthode AJAX. |
| **Demande du client** |
| Réaliser une page permettant de lire, créer, modifier ou supprimer des destinations dans le cadre de la réalisation d’un blog de voyages. |
| **Budget disponible** |
| Matériel : Un ordinateur avec un serveur Wamp, un hébergement o2switch.  Licences : PhpStorm |
| **Outils disponibles** |
| PhpStorm |
| **Confidentialité** |
|  |

## Expression fonctionnelle du besoin

Liste des fonctionnalités attendues :

|  |
| --- |
| **Front office** |
| Page de destinations :  Ce projet est composé d’une seule page, listant les destinations actuellement stockées dans la base de données. |

|  |
| --- |
| **Back Office** |
| Le back office est situé sur la même page, cependant, il n’est accessible que si l’administrateur se connecte via son compte d’administrateur (2ème mission E5).  Il permet à l’administrateur, via différents boutons, de créer, modifier ou supprimer une destination. |

## Contraintes

|  |
| --- |
| **Générales** |
| Temps : 1 mois. |
| **Juridiques** |
| Aucune donnée utilisateur ne sera traitée ou récoltée par le site. |
| **Techniques** |
| AJAX (asynchronous JavaScript and XML) |
| **Ergonomique** |
| Responsive. |

## Gestion des droits d’accès

|  |
| --- |
| **Administrateur** |
| Moi-même. |

|  |
| --- |
| **Visiteurs** |
| Toute personne possédant l’URL de mon site. |

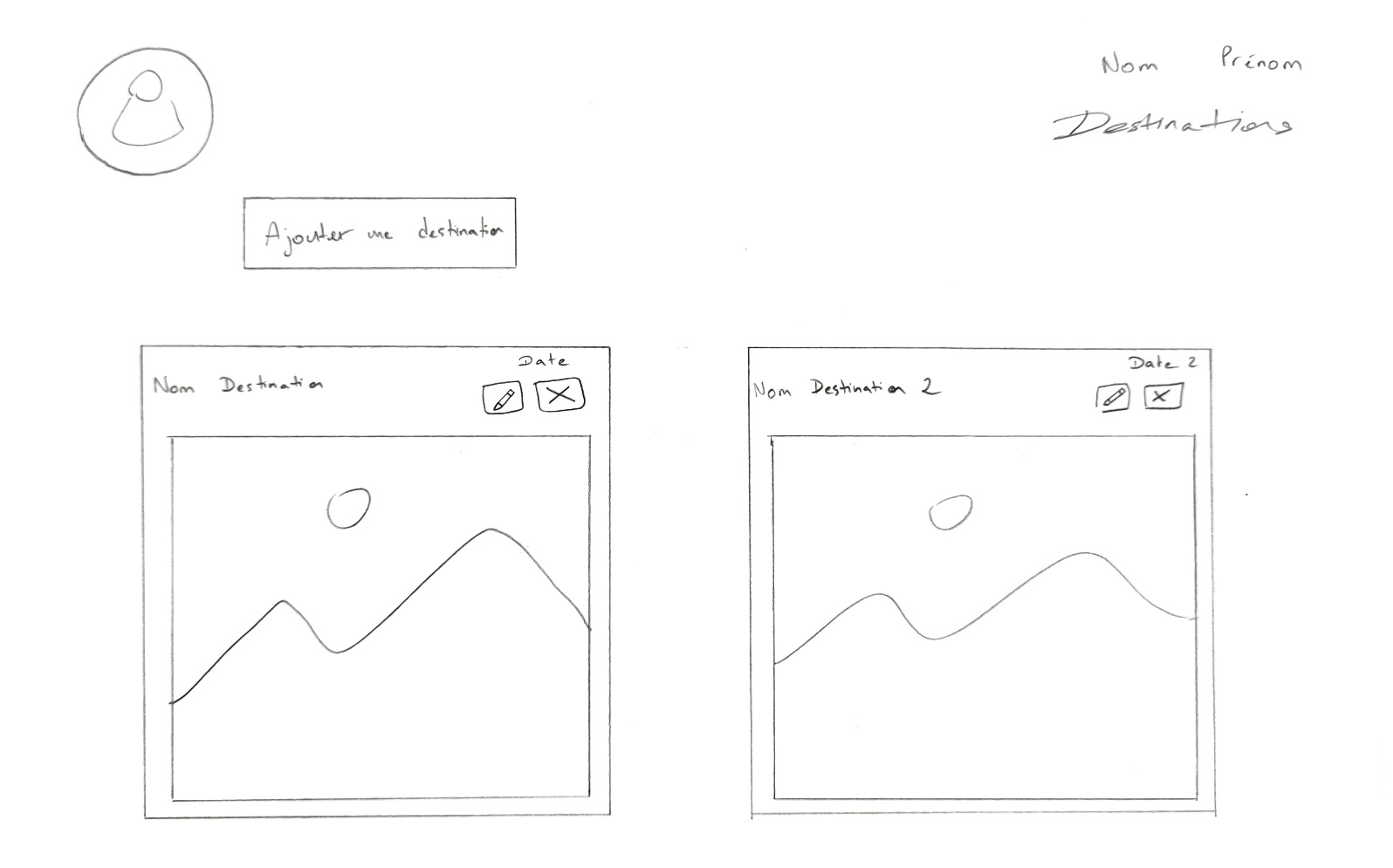
# Conception de l’application

## Schématisation de l'application

1.2 Participer à la conception de l’architecture d’une solution applicative

1.3 Modéliser une solution applicative

Modélisation réalisée sur papier lors du maquettage de l’application.



## Choix des technologies employées

1.4 Exploiter les ressources du cadre applicatif

Langage serveur PHP couplé au langage de données MySQL. Grâce à ces deux langages, je peux interagir avec la base de données afin de lire des informations, en créer, en modifier ou en supprimer.

Langage Javascript pour le contrôle des formulaires, et pour les requêtes asynchrones.

Pas besoin de framework puisque le but est de réaliser un CRUD en vanilla JS (sans framework) de sorte à pouvoir le réutiliser ultérieurement dans d’autres projets.

La méthode AJAX a été retenue afin de permettre d’effectuer des requêtes sans avoir besoin de recharger la page. Cette méthode offre donc un réel avantage en termes de fluidité et d’expérience utilisateur.

## Présentation des composants logiciels

1.5 Identifier, développer, utiliser ou adapter des composants logiciels

1.6 Exploiter les technologies Web pour mettre en œuvre les échanges entre applications, y compris de mobilité

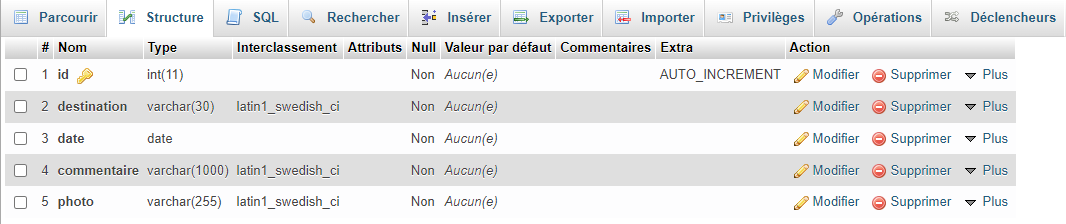
Tutoriel exploité : <https://www.youtube.com/watch?v=oie6T8IVeMA>, adapté à ma problématique.

# Conception de la BDD

## Schéma de la base de données

3.3 Concevoir ou adapter une base de données

Les données sont modélisées conformément au besoin de la solution applicative.



5 champs :

Id : auto-incrémenté, il permet de différencier les destinations présentes dans la BDD et d’effectuer des actions sur ces dernières via des fonctions javascript (modification ou suppression).

Destination : nom de la destination.

Date : Date à laquelle la destination a été visitée.

Commentaire : Texte décrivant la destination.

Photo : URL d’une photo de la destination.

## Installation et configuration de la base de données

3.4 Administrer et déployer une base de données

L’outil de base de données utilisé est PHP my admin pour le langage MySQL.

Les deux sont automatiquement installés en local grâce au serveur Wamp, et sur le site en ligne sur Cpanel.

## Interaction de l'application avec la base de données

1.7 Utiliser des composants d’accès aux données

3.1 Exploiter des données à l’aide d’un langage de requêtes

Les données persistantes liées à la solution applicative sont exploitées à travers un langage de requête lié à la base de données qui peut être le langage de requête proposé par les échanges applicatifs des technologies Web, un langage de requête présent dans l’outil de correspondance objet-relationnel ou toute autre solution de persistance.

* 1. Développer des fonctionnalités applicatives au sein d’un système de gestion de base de données (relationnel ou non)

**Requête de lecture de base de données :**

$sql = "SELECT \* FROM destinations ORDER BY date DESC";

Cette requête permet de sélectionner toutes les destinations en les triant par date décroissante.

**Requête d’insertion à la base de données :**

$sql = "INSERT INTO destinations (destination, date, commentaire, photo)

VALUES (:creer\_destination, :creer\_date, :creer\_commentaire, :creer\_photo)";

Cette requête permet de créer une destination. Aux champs destination, date, commentaire, photo on ajoute les valeurs creer\_destination, créer date, créer commentaire, créer photo (champs remplis par l’administrateur lors de la création d’une destination).

L’id est lui généré automatiquement par le SGBD.

**Requête de modification :**

$sql = "UPDATE destinations SET

destination = :modifier destination, date = :modifier date,

commentaire = :modifier\_commentaire, photo = :modifier photo WHERE id = :modifier\_id";

Cette requête permet de modifier une destination. On modifie les champs destination, date, commentaire, photo en remplaçant les anciennes valeurs de la BDD par les nouvelles, contenues dans les variables modifier\_destination, modifier\_date, modifier\_commentaire, modifier\_photo (champs remplis par l’administrateur lors de la modification d’une destination). L’élément à modifier est identifié grâce au champ id (renseigné par la variable modifier\_id).

**Requête de suppression :**

$sql = "DELETE FROM destinations WHERE id = :supprimer\_id";

Cette requête permet à l’administrateur de supprimer une destination en l’identifiant en renseignant son id.

# Gestion de la qualité

## Méthodologie et outil de versioning

1.8 Intégrer en continu les versions d’une solution applicative

Pour intégrer en continu les versions de la solution applicative, j’utilise l’outil de versioning Github.

Puisque je suis le seul à l’utiliser, je n’ai pas besoin d’autres branches que la branche Main. Au fur et à mesure des modifications de la solution applicative, je « pousse » les modifications vers Github, en laissant un commentaire afin d’identifier la modification réalisée. Cela me permet d’avoir une trace des modifications que j’effectue.

Une image contenant texte, moniteur, capture d’écran, intérieur

Description générée automatiquement

## Gestion des tests de l'application

1.9 Réaliser les tests nécessaires à la validation ou à la mise en production d’éléments adaptés ou développés

Les composants logiciels sont validés par les procédures de tests unitaires et fonctionnels.

1.11 Exploiter les fonctionnalités d’un environnement de développement et de tests

expliquer comment vous gérez les tests

**Unitaires :**

Grâce à des var\_dump en PHP et à des console.log en javascript (fonctions permettant de renvoyer des variables), je teste points par points le logiciel tout en le créant.

**Fonctionnels :**

Je me mets à la place de l’utilisateur final et je teste trois cas de figure :

**La création :**

Je crée une destination en y insérant un nom, une date, une description et un URL d’image (si je n’insère pas un de ces quatre champs, un message d’erreur sous la forme d’une popup apparaît et la création n’a pas lieu).

Message d’erreur si on oublie de renseigner un champ :

Une image contenant texte

Description générée automatiquement

Formulaire rempli :

Une image contenant texte

Description générée automatiquement

Lorsque je valide le formulaire, la page n’est pas rechargée mais l’élément a été créé :

Une image contenant texte, moniteur, capture d’écran, équipement électronique

Description générée automatiquement

**La modification :**

Lorsque je clique sur le bouton bleu en haut à droite de l’élément, le même formulaire qu’avant s’affiche mais avec les informations de cette destination préremplies :



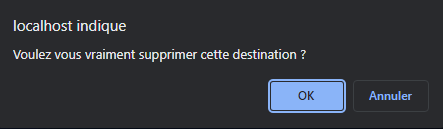
Je change les informations puis valide le formulaire (de la même façon qu’avant, si un champ est vide le formulaire n’est pas validé et un message d’erreur apparaît).

**Formulaire modifié :**

Une image contenant texte, moniteur, équipement électronique, écran

Description générée automatiquement

**Suppression :**

Lorsque je clique sur la croix rouge (suppression), une fenêtre popup apparaît : 

Si l’utilisateur clique sur Ok, la destination est supprimée.

## Rédaction de la documentation

1.10 Rédiger des documentations technique et d’utilisation d’une solution applicative

Puisque je suis le seul à pouvoir accéder à la création, modification, suppression des destinations, il n’est pas nécessaire de rédiger une documentation technique. Cependant, si à l’avenir une autre personne venait à être amenée à utiliser le logiciel, il faudrait en rédiger une.

# Mise en production

## Description de l'environnement de développement

Windows 10 comme système d’exploitation.

PhpStorm comme IDE.

Wampserver comme serveur local.

MySQL couplé à PhpMyAdmin comme SGBD.

Je suis à l’aise avec cet environnement de travail car c’est le même dans l’entreprise ou je suis apprenti.

J’ai choisi o2switch comme hébergement à cause des bons avis concernant le service client.

## Description de l'environnement de production

L’environnement de production est un serveur mutualisé hébergé chez o2switch.



## Etape de mise en production

Une fois que toutes les fonctionnalités essentielles de l’application ont été réalisées et testées et que le cahier des charges a été respecté, j’ai mis en production la page de destinations.

## Mise en place de la sauvegarde

Puisque cette application n’a aucun but autre que de me fournir un modèle pour des travaux personnels ultérieurs, aucune sauvegarde de la base de données n’est nécessaire.

# Gestion de la maintenance (corrective / évolutive)

## Mise à jour de la documentation du SI

2.1 Recueillir, analyser et mettre à jour les informations sur une version d’une solution applicative

Les composants logiciels sont documentés de manière à être réutilisés.

La documentation technique et d’utilisateurs de la solution applicative sont mises à jour.

2.4 Mettre à jour des documentations technique et d’utilisation d’une solution applicative

- une documentation des versions vient appuyer l’intégration continue ;

- les composants logiciels sont documentés de manière à être réutilisés ;

logiquement lorsque vous mettez une application en place en entreprise, vous devez mettre à jour la CMDB pour que le reste de l'équipe soit au courant des interactions entre votre appli et le reste du SI

et qu'en cas de bug, qui se produirait 2 ans plus tard, lorsque tout le monde aura oublié comment fonctionne votre application, qu'il soit possible de débuguer rapidement puisqu'on a toutes les infos

## Evaluation de la qualité de l’application

2.2 Évaluer la qualité d’une solution applicative

Il n’y aura pas d’autres utilisateurs que moi-même.

## Procédure de correction d'un dysfonctionnement

2.3 Analyser et corriger un dysfonctionnement

Lorsqu’un dysfonctionnement est constaté, j’effectue des modifications et tests sur une version en local afin de corriger le problème. Une fois le problème corrigé, je remplace les fichiers impactés sur le site en ligne.

## Gestion des tests de mise à jour

2.5 Élaborer et réaliser les tests des éléments mis à jour

Les composants logiciels adaptés et/ou corrigés sont validés par les procédures de tests unitaires et fonctionnels.

# Bilan du projet

Ce projet m’a permis entre autres de me familiariser avec la méthode AJAX en vanilla JS, de sorte à comprendre le fonctionnement des requêtes asynchrones.

De ce fait, je possède une base solide réutilisable lors de projets ultérieurs.