151 - Le Classique 8

Documentation de Projet

Date de création : 28.01.2025

Baptiste Schneider

|  |  |  |
| --- | --- | --- |
|  |  | Module du 28.01.2025 au 25.02.2025 |

Table des matières

[1 Introduction 4](#_Toc190086776)

[2 Analyse 5](#_Toc190086777)

[2.1 Uses Case 5](#_Toc190086778)

[2.2 Maquettes 6](#_Toc190086779)

[2.3 Diagramme activité 7](#_Toc190086780)

[2.4 Diagramme de séquences systèmes 8](#_Toc190086781)

[2.5 Schéma ER 9](#_Toc190086782)

[3 Conception 10](#_Toc190086783)

[3.1 Diagrammes de classe 10](#_Toc190086784)

[3.1.1 Client 10](#_Toc190086785)

[3.1.2 Serveur 11](#_Toc190086786)

[3.2 Schéma relationnel 11](#_Toc190086787)

[3.3 Diagramme séquence interactions 12](#_Toc190086788)

[3.4 Test fonctionnels 12](#_Toc190086789)

[4 Conception des tests 13](#_Toc190086790)

[4.1 Descente de code 13](#_Toc190086791)

[4.2 Problèmes rencontrés 13](#_Toc190086792)

[4.3 Tests fonctionnels 13](#_Toc190086793)

[4.4 Hébergement 13](#_Toc190086794)

[5 Synthèse 14](#_Toc190086795)

[5.1 Présentation réalisation 14](#_Toc190086796)

[5.2 Différences entre planning et réalisation 14](#_Toc190086797)

[5.3 Conclusion 14](#_Toc190086798)

# Introduction

Durant ce projet, je vais créer une application web de gestion pour une salle de billard appelée Le Classique 8. Cette application centralisera des fonctionnalités telles que la réservation des tables ou encore la visualisation des tables.

Elle sera composée d’un client web pour l’interface utilisateur, d’un serveur pour gérer la logique métier et les échanges, et d’une base de données pour stocker les informations essentielles.

L’objectif est de concevoir une solution intuitive, performante et évolutive, adaptée aux besoins spécifiques de la salle.

# Analyse

## Uses Case

Ce diagramme use-case illustre l'architecture client-serveur de mon projet. Il représente les interactions entre deux types d’utilisateurs, Visiteur et Administrateur, avec un client HTML et un serveur PHP via HTTP(S). Le Visiteur peut se connecter et visualiser les tables, tandis que l’Administrateur peut également réserver une table et se déconnecter.

Chaque action déclenche une requête HTTP(S) vers le serveur PHP, qui gère les processus backend comme le contrôle du login, la réservation, la déconnexion et la fermeture de session, en interagissant avec une base de données (BD).

Ce schéma met en évidence la répartition des responsabilités entre le client et le serveur ainsi que le flux des interactions entre les différentes entités du système.

Une image contenant texte, diagramme, capture d’écran, cercle

Description générée automatiquement

## Maquettes

Les maquettes présentent une interface utilisateur pour une plateforme de réservation de tables de billard. L'application comprend une page de connexion permettant aux utilisateurs de s'authentifier ou d'accéder directement à l'accueil via un bouton de passage.

La page d'accueil affiche les tables disponibles avec une interface de réservation où les joueurs peuvent saisir leur nom et réserver une session. Une section de navigation permet d'accéder à une page de contact fournissant des informations pour joindre le service client via téléphone, réseaux sociaux ou email.

L’ensemble des pages est structuré avec une barre de navigation facilitant le passage entre les sections principales.

Une image contenant texte, capture d’écran, diagramme, affichage

Description générée automatiquement

## Diagramme activité

Ce diagramme d'activité illustre le processus de réservation d'une table par un utilisateur via un client HTML interagissant avec un serveur PHP et une base de données MariaDB.

L'utilisateur clique sur "Réserver" ; si une erreur survient, un message s'affiche, sinon les informations de l'utilisateur sont présentées.

Les paramètres sont transmis au serveur PHP, qui vérifie leur présence ainsi que l'authentification de l'utilisateur avant de se connecter à la base de données.

Si la table est pleine, la réservation échoue, sinon elle est validée. MariaDB gère la recherche et l'enregistrement des réservations, garantissant une gestion efficace des demandes.

Une image contenant texte, capture d’écran, diagramme, Parallèle

Description générée automatiquement

## Diagramme de séquences systèmes

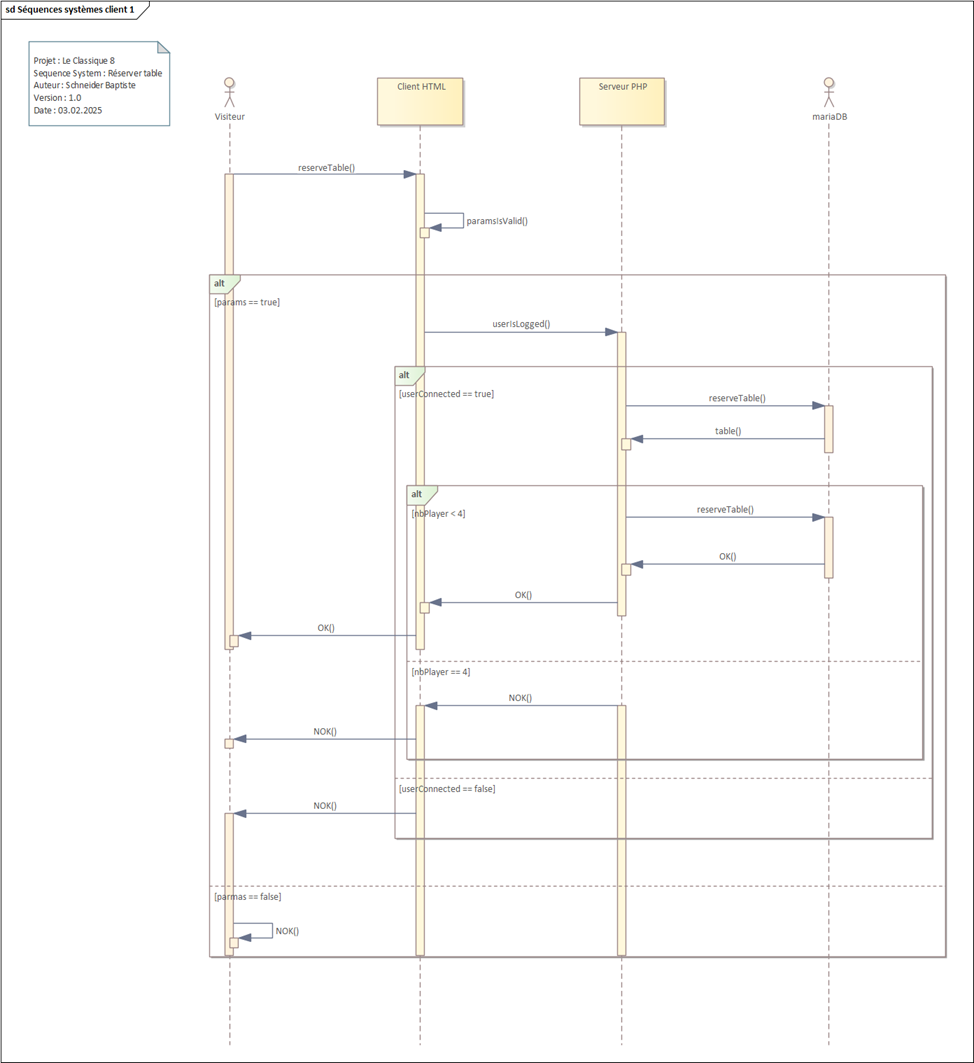
Ce diagramme de séquence décrit le processus de réservation d'une table. Il illustre les interactions entre un visiteur, un client HTML, un serveur PHP et une base de données MariaDB.

Le processus commence par la demande de réservation, suivie de la validation des paramètres. Si ces derniers sont valides, le système vérifie si l'utilisateur est connecté. Si l'utilisateur est connecté, une autre vérification s'assure que le nombre de joueurs est inférieur à 4 pour autoriser la réservation.

En cas de succès, la demande est transmise au serveur PHP, qui interagit avec la base de données pour finaliser la réservation.

Si l'une des conditions (paramètres invalides, utilisateur non connecté ou nombre de joueurs insuffisant) n'est pas remplie, la réservation est refusée avec un message de non-validation (NOK).

Ce diagramme met en évidence la logique de validation et les échanges entre les différents composants du système pour assurer un processus de réservation structuré et fiable.



## Schéma ER

Le diagramme entité-association présenté illustre la gestion des réservations de tables par des clients. Il comprend trois entités principales : Client, Table et Localité. Un client peut effectuer plusieurs réservations de tables, et une table peut être réservée par plusieurs clients

Chaque client est également associé à une localité via la relation habite, signifiant qu'un client réside dans une seule localité.

Ce diagramme structure les interactions essentielles pour la gestion des réservations et la localisation des clients, facilitant ainsi l'organisation et l'optimisation du service.



# Conception

## Diagrammes de classe

### Client

Le diagramme de classe illustre l'organisation des interfaces et leurs relations au sein de l'application cliente. Il se compose de plusieurs interfaces : loginCtrl.js, IndexCtrl.js, TableCtrl.js, httpService.js et Table.js, chacune définissant des attributs et des méthodes spécifiques.

Les connexions entre ces interfaces sont représentées par des associations et des relations d'utilisation. Les attributs sont indiqués avec leur type de données, tandis que les méthodes affichent leurs paramètres et types de retour. L'ensemble du diagramme structure l'interaction entre les différentes composantes du projet en mettant en évidence les dépendances et la répartition des responsabilités.

Il est important de savoir que ce diagramme de classe n’est pas une version complète de ce que mon application utilisera car je ne sais pas encore ce que j’utiliserais pour la faire fonctionner

Une image contenant texte, capture d’écran, diagramme, ligne

Description générée automatiquement

### Serveur

Le diagramme de classe pour la partie serveur décrit la structure et les relations entre plusieurs interfaces. Il inclut serveur.php, Ctrl.php et WrkDB.php, chacune possédant ses propres attributs et méthodes.

Les relations entre ces interfaces sont représentées par des associations et une relation d'utilisation, indiquant les dépendances entre les composants. Les attributs sont précisés avec leurs types de données, et les méthodes affichent leurs paramètres et types de retour. Ce diagramme met en évidence l'organisation modulaire du système, définissant clairement les interactions entre les différentes parties du projet.

Il est important de savoir que ce diagramme de classe n’est pas une version complète de ce que mon application utilisera car je ne sais pas encore ce que j’utiliserais pour la faire fonctionner

Une image contenant texte, capture d’écran, diagramme, Police

Description générée automatiquement

## Schéma relationnel

Ce diagramme représente la structure d'une base de données relationnelle pour la gestion des clients et de leurs associations avec des tables.

La table T\_Client stocke les informations des clients, y compris leur nom, prénom, email, mot de passe et une référence à leur localité via FK\_localite, qui est une clé étrangère pointant vers la table T\_Localite. Cette dernière contient des informations sur les localités, incluant un identifiant unique (PK\_Localite), le nom de la localité et son code postal.

La table de relation TR\_Client\_Table établit une relation entre T\_Client et T\_Table, permettant d'associer plusieurs clients à plusieurs tables. Enfin, T\_Table contient les tables disponibles, identifiées par PK\_Table et un numéro unique.

Ce modèle assure une gestion structurée des clients, de leur localisation et des tables auxquelles ils sont associés.

Une image contenant texte, capture d’écran, ligne, Police

Description générée automatiquement

## Diagramme séquence interactions

## Test fonctionnels

Liste des tests pour le processus de réservation d’une table de billard

**Test de validation des paramètres**

* Test de réservation d’une table avec un paramètre manquant.
* Test de réservation d’une table avec un format de paramètre invalide.

**Test de validation de l’identifiant de la table**

* Test de réservation d’une table avec un identifiant inexistant.
* Test de réservation d’une table avec tous les paramètres valides.

**Test de validation de l’authentification avant d’exécuter la réservation**

* Test de réservation d’une table sans être authentifié.
* Test de réservation d’une table en étant authentifié.

**Test de disponibilité de la table**

* Test de réservation d’une table qui est déjà pleine.
* Test de réservation d’une table qui est libre.

# Conception des tests

* 1. Implémentation

## Descente de code

## Problèmes rencontrés

## Tests fonctionnels

## Hébergement

# Synthèse

## Présentation réalisation

## Différences entre planning et réalisation

## Différence entre la conception et la réalisation

## Conclusion