



Projet 1 : Système de Gestion de Bibliothèque

En-tête du Document

Nom de la Section : Section B

Table des Matières

1. Introduction
2. Analyse des Besoins
 - 2.1. Acteurs du Système
 - 2.2. Besoins Fonctionnels
 - 2.3. Données et Entités Métier (Modèle du Domaine)
 - 2.4. Exigences Non Fonctionnelles
3. Diagrammes UML
 - 3.1. Diagrammes de Cas d'Utilisation
 - 3.2. Diagrammes de Classes
 - 3.3. Diagramme de Séquences
 - 3.4. Diagramme d'États-Transitions
 - 3.5. Diagramme de Déploiement
4. Discussion

5. Conclusion
 6. Annexes
 - 6.1. Justification de l'Outil de Modélisation
-

1. Introduction

Ce projet résout le problème de gestion d'une bibliothèque en présentiel. Il vise à modéliser un système de gestion de bibliothèque à l'aide d'UML, en décrivant le système et son objectif principal.

2. Analyse des Besoins

2.1. Acteurs du Système

- **Lecteur (ou Visiteur)** : Toute personne consultant les livres sans abonnement.
- **Abonné** : Lecteur possédant une carte d'abonnement, autorisé à emprunter et retourner des livres.
- **Gérant de la Bibliothèque** : Personnel de la bibliothèque responsable de la gestion des emprunts, retours, abonnements, et du stock de livres.

2.2. Besoins Fonctionnels

2.2.1. Gestion des Livres et du Catalogue

- **Consulter le catalogue de livres** : Permettre aux visiteurs et aux abonnés de rechercher et visualiser les informations des livres disponibles. La consultation ne nécessite pas d'abonnement.
- **Gérer le stock de livres** : Le système doit permettre au gérant de mettre à jour le statut et la quantité des livres (disponible/emprunté).

2.2.2. Gestion des Emprunts

- **Emprunter un livre** :
 - **Acteurs** : Abonné (principal), Gérant de la Bibliothèque (secondaire).
 - **Description** : Permettre à un abonné d'emprunter un livre.
 - **Scénario Principal** :

1. L'abonné choisit un livre et le présente avec sa carte d'abonnement au gérant.
 2. Le gérant vérifie la carte et la disponibilité du livre.
 3. Si valide, le gérant valide l'emprunt en saisissant les informations.
 4. Le système enregistre l'emprunt et met à jour le stock (livre comme "emprunté").
 5. Le système génère un reçu d'emprunt.
 6. L'abonné repart avec le livre.
- **Règles / Conditions :** Le livre doit être disponible. La carte d'abonnement doit être valide. La date d'emprunt est automatique. Le délai d'emprunt doit être un nombre positif raisonnable (max 30 jours).
 - **Flux Alternatifs :** Si la carte n'est pas valide ou le livre non disponible, le gérant refuse l'emprunt.

2.2.3. Gestion des Retours

- **Retourner un livre :**
 - **Acteurs :** Abonné (principal), Gérant de la Bibliothèque (secondaire).
 - **Description :** Permettre à un abonné de retourner un livre emprunté.
 - **Scénario Principal :**
 1. L'abonné présente le livre et sa carte d'abonnement au gérant.
 2. Le gérant vérifie l'état du livre et la validité de la carte.
 3. Si tout est correct, le gérant valide le retour.
 4. Le système met à jour les informations de l'abonné et le stock (livre comme "disponible").
 5. Le système génère un reçu de retour.
 6. L'abonné remet son livre à la bibliothèque.
 - **Règles / Conditions :** Le livre doit être indemne. La carte doit être valide. La date de retour est automatique. La validité du retour est calculée en comparant le délai d'emprunt initial avec la durée réelle de l'emprunt.

- **Flux Alternatifs / Pénalités** : Si le livre n'est pas rendu à temps ou n'est pas indemne, le gérant peut réduire la validité de la carte d'abonnement ou la résilier (préciser les conditions).

2.2.4. Gestion des Abonnements

- **Gérer un abonnement :**
 - **Acteur Principal** : Gérant de la Bibliothèque.
 - **Description** : Permettre au gérant de créer, modifier, et gérer les cartes d'abonnement (y compris leur validité).
 - **Règles** : La validité de la carte peut être réduite si l'emprunt n'est pas valide.

2.3. Données et Entités Métier (Modèle du Domaine)

- **Carte d'Abonnement** : ID, Numéro de carte, Type de carte (VIP, Standard), Nom du propriétaire (lié à un Abonné), Validité (durée en jours ou date de fin).
- **Catégorie** : ID, Nom (Science-fiction, Histoire), Liste de Livres.
- **Livre** : ID, Titre, Auteur, Catégorie, Disponibilité (Oui/Non ou Nombre d'exemplaires).
- **Gérant** : ID, Nom, Mail.
- **Emprunt** : ID, Date d'emprunt, Date de retour prévue, Date de retour réelle, Livre concerné, Carte d'abonnement concernée, Statut (en cours, retourné, en retard).

2.4. Exigences Non Fonctionnelles

- **Performance** :
 - Le système doit afficher les résultats de recherche de livres en moins de 2 secondes.
 - L'enregistrement d'un emprunt/retour doit prendre moins de 5 secondes.
- **Sécurité** :
 - Seuls les gérants authentifiés et autorisés peuvent effectuer les opérations d'emprunt, de retour et de gestion des abonnements.

- Les données des abonnés et des emprunts doivent être protégées contre tout accès non autorisé.

- **Utilisabilité :**

- L'interface utilisateur pour le gérant doit être intuitive et facile à utiliser pour minimiser les erreurs.

- **Fiabilité :**

- Le système doit être disponible 99% du temps pendant les heures d'ouverture de la bibliothèque.
- Les données doivent être sauvegardées quotidiennement.

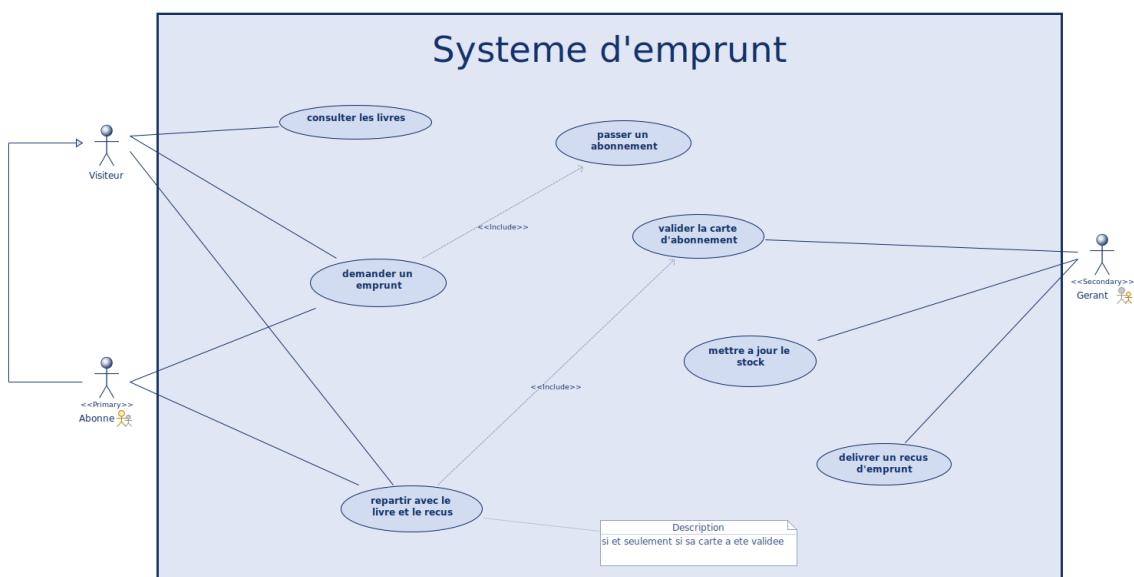
- **Maintenabilité :**

- Le système doit être facilement adaptable pour l'ajout de nouvelles catégories de livres ou de nouveaux types d'abonnement.

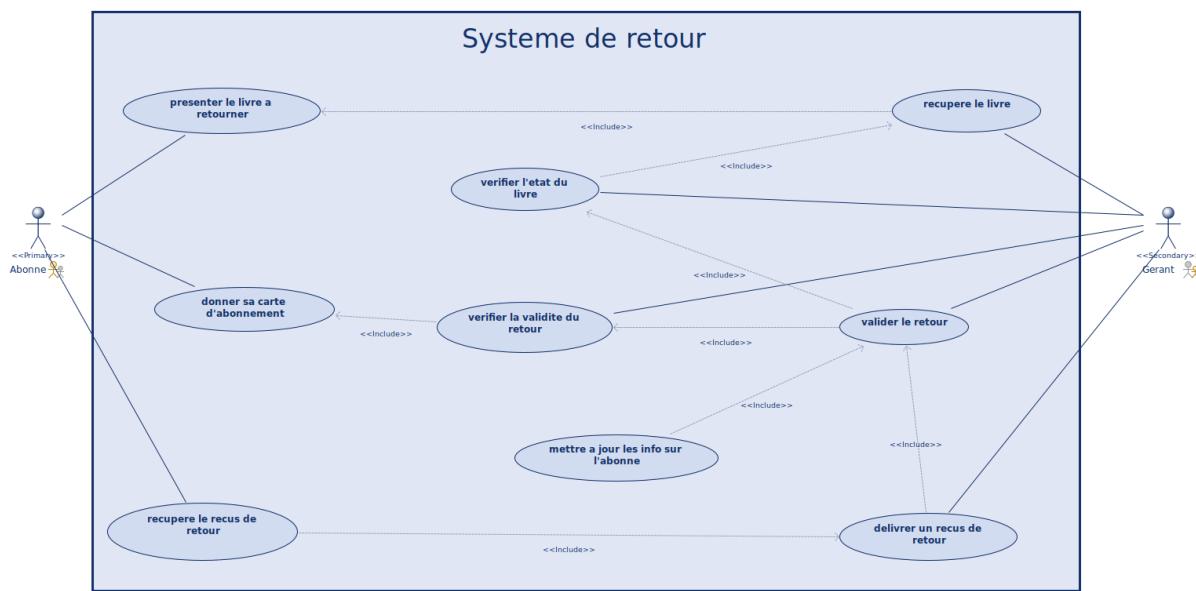
3. Diagrammes UML

3.1. Diagrammes de Cas d'Utilisation

- **Cas de l'Emprunt :** (Image du diagramme "Système d'emprunt" avec "consulter les livres", "demander un emprunt", "passer un abonnement", "valider la carte d'abonnement", "mettre à jour le stock", "délivrer un reçu d'emprunt", "repartir avec le livre et le reçu".) Il montre les interactions entre le Visiteur, l'Abonné et le Gérant.

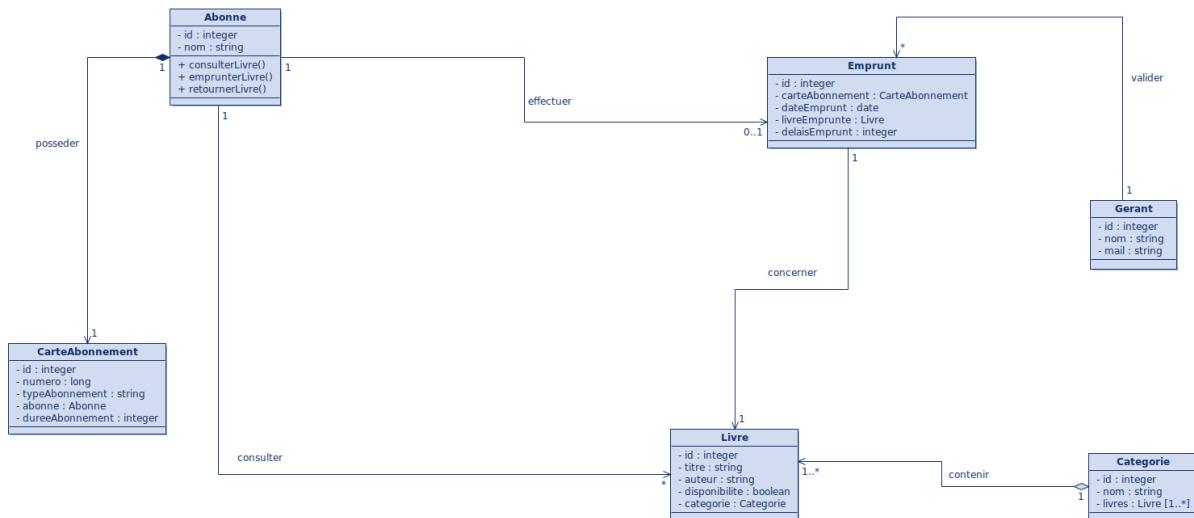


- **Cas du Retour** : (Image du diagramme "Système de retour" avec "présenter le livre à retourner", "vérifier l'état du livre", "donner sa carte", "vérifier la validité du retour", "récupérer un reçu de retour", "mettre à jour les infos sur l'abonné", "récupérer un livre", "valider le retour", "délivrer un reçu de retour".) Ce diagramme illustre le processus de retour d'un livre.

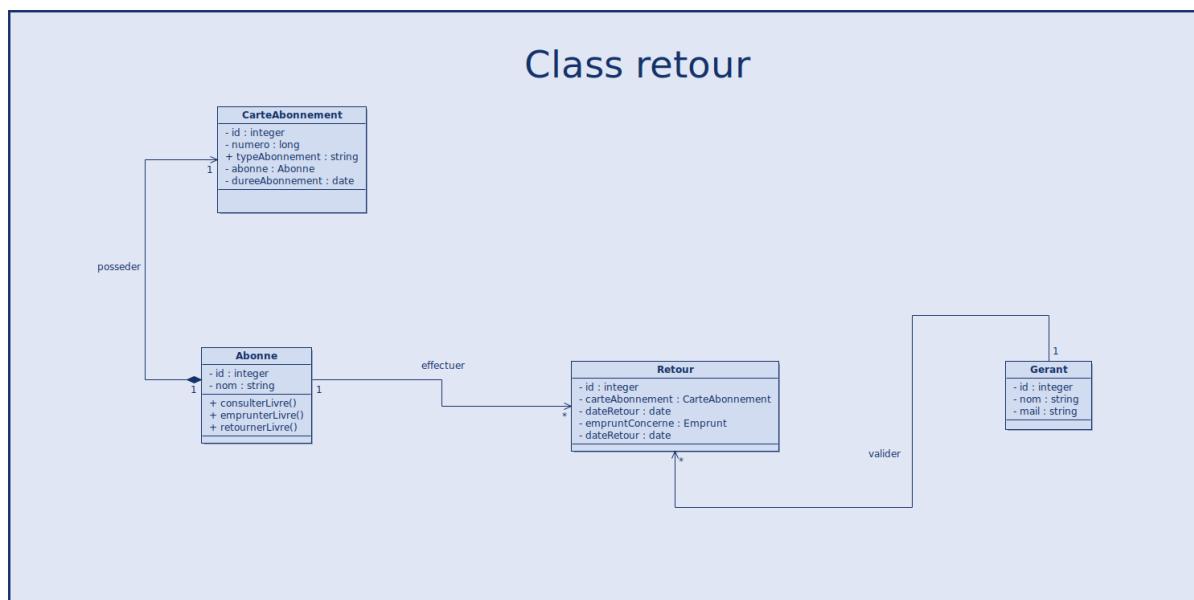


3.2. Diagrammes de Classes

- **Classes intervenant dans le système d'emprunt :** (Image du diagramme de classes montrant `CarteAbonnement`, `Abonne`, `Livre`, `Emprunt`, avec leurs attributs et relations.) Ce diagramme représente les entités clés et leurs relations pour le processus d'emprunt.



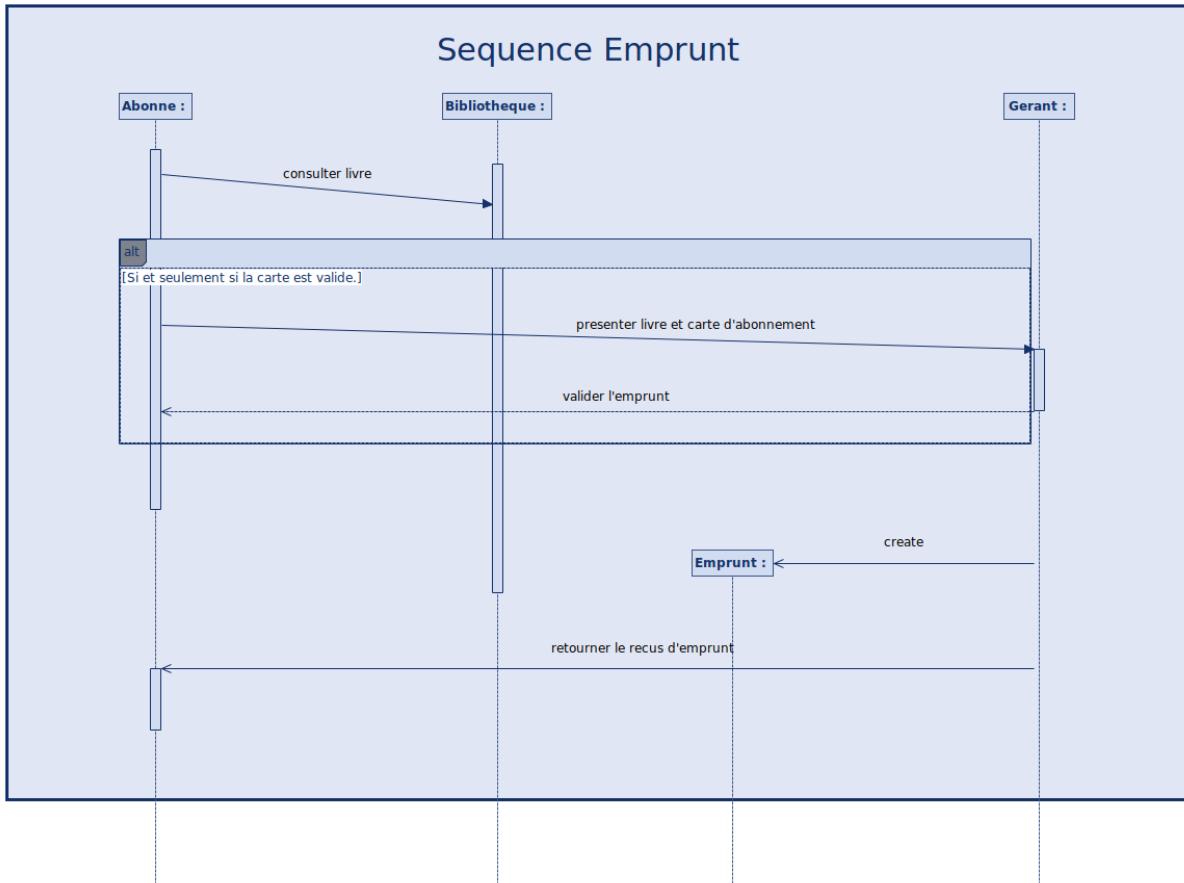
- Classes intervenant dans le système de retour :** (Image du diagramme de classes montrant **Gerant**, **Abonne**, **Livre**, **Emprunt**, **Categorie**, avec leurs attributs et relations.) Ce diagramme étend le modèle de classes pour inclure les entités pertinentes au retour et à la gestion générale de la bibliothèque.



3.3. Diagramme de Séquences

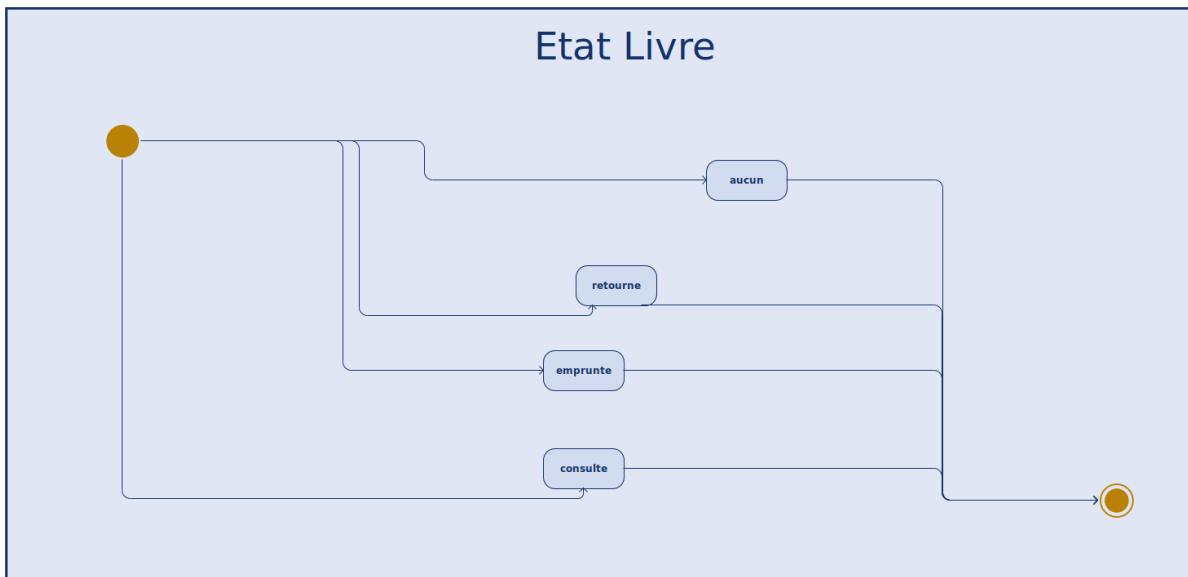
- Séquence Emprunt :** (Image du diagramme de séquences "Sequence Emprunt" montrant l'interaction entre **Abonne**, **Bibliotheque**, **Gerant**, et **Emprunt**.)

Ce diagramme détaille le déroulement chronologique des messages échangés pour le processus d'emprunt, y compris la validation de la carte et la création de l'emprunt.



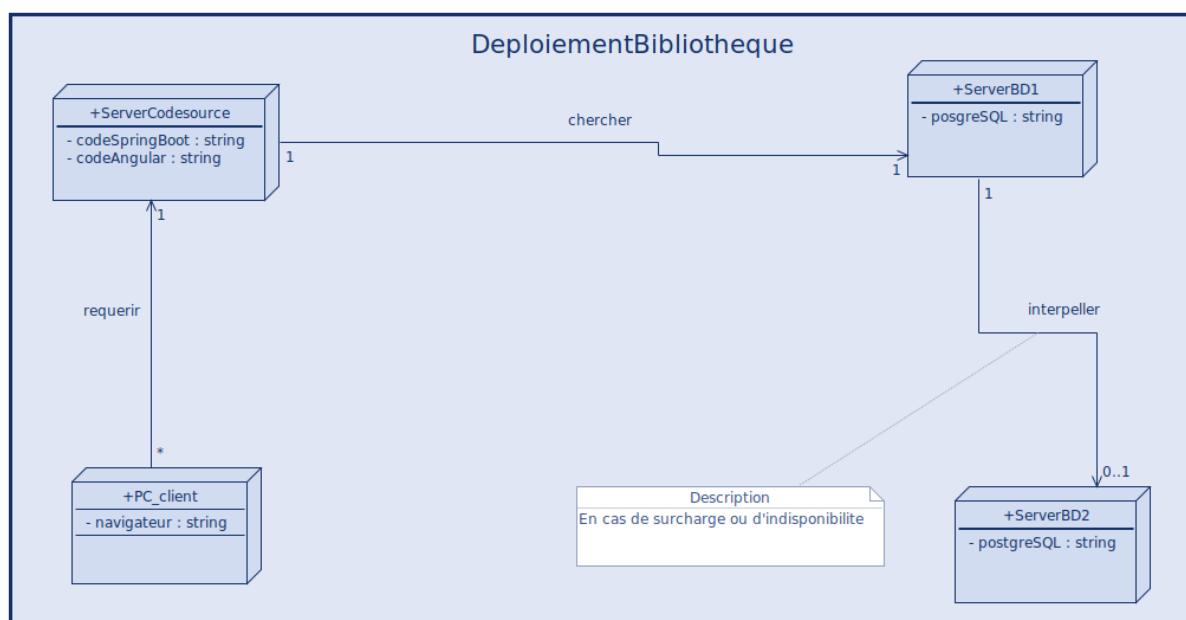
3.4. Diagramme d'États-Transitions

- **État Livre** : (Image du diagramme "Etat Livre".) Ce diagramme tente de modéliser les différents états d'un livre (par exemple, disponible, emprunté, en retard) et les transitions entre ces états. Tel que noté précédemment, ce diagramme pourrait bénéficier d'une révision pour s'assurer que les nœuds représentent bien des états stables (ex: Disponible, Emprunté) et les flèches des transitions déclenchées par des événements spécifiques (ex: Emprunter, Retourner), avec des labels explicites.



3.5. Diagramme de Déploiement

- **Déploiement Bibliothèque :** (Image du diagramme "DeploiementBibliotheque" montrant "ServerBD1 (PostgreSQL)", "ServerCodesource (SpringBoot, Angular)", "PC client (navigateur)", "ServerBD2 (PostgreSQL)". Il est noté "En cas de surcharge ou d'indisponibilité".) Ce diagramme illustre l'architecture matérielle et logicielle où le système sera déployé, y compris la base de données, le serveur de code source et les clients.



4. Discussion

Cette section analyse les choix de conception effectués, les défis rencontrés lors de la modélisation et les solutions adoptées.

Défis Rencontrés :

La réalisation de ce projet a été marquée par plusieurs défis :

- **Manque de temps** : La contrainte de temps a été un facteur limitant, exigeant une gestion rigoureuse des tâches et une priorisation efficace pour livrer le projet dans les délais impartis.
- **Cohérence du groupe** : Assurer une cohésion et une compréhension uniforme des exigences et des choix de modélisation au sein du groupe a demandé des efforts constants de communication et de collaboration.
- **Étude du projet** : La phase d'étude et d'analyse des besoins a parfois présenté des complexités inattendues, nécessitant une analyse approfondie pour bien cerner toutes les fonctionnalités et contraintes du système de gestion de bibliothèque.

Analyse des Choix de Conception et Solutions :

- **Incohérence dans le Diagramme d'États-Transitions** : Le diagramme d'états-transitions pour le livre (Section 3.4) pourrait être amélioré. Les éléments présentés comme des "états" semblent parfois être des actions ou des événements, ce qui ne correspond pas entièrement à la sémantique d'un diagramme d'états-transitions UML. De plus, les transitions entre les états devraient être clairement labellisées avec l'événement déclencheur et, si applicable, l'action associée. Une révision pour s'assurer que les nœuds représentent bien des états stables et les flèches des transitions déclenchées par des événements spécifiques serait bénéfique.
- **Redondance des Diagrammes de Classes** : La présence de deux diagrammes de classes très similaires (Section 3.2) a été une décision initiale, mais pour une meilleure clarté et concision du document, ils pourraient être fusionnés en un seul diagramme complet.
- **Explication des choix** : Nous avons opté pour une modélisation claire des entités telles que `Livre`, `Abonné`, `Emprunt` pour refléter directement les besoins métier de la bibliothèque. Les relations ont été définies pour représenter

fidèlement les interactions entre ces entités, comme la manière dont un abonné **effectue** un **emprunt** d'un **Livre**.

5. Conclusion

Cette section résume les résultats du projet, les principaux enseignements tirés par l'équipe et les perspectives d'évolution future du système.

Synthèse des Résultats :

Ce projet a permis de concevoir un modèle UML complet pour un système de gestion de bibliothèque, couvrant les aspects fonctionnels, structurels et comportementaux. Les diagrammes produits offrent une base solide pour le développement futur de l'application.

Leçons Apprises :

La réalisation de ce projet a souligné l'importance de l'**ardeur** et de la **discipline**. Malgré les défis liés au manque de temps et à la nécessité de maintenir la cohérence du groupe, une approche disciplinée et un engagement soutenu ont été essentiels pour surmonter les obstacles et approfondir notre compréhension de la modélisation UML et de l'analyse de projet.

Perspectives :

Pour les évolutions futures, nous pourrions envisager l'ajout de fonctionnalités telles que la gestion des réservations de livres en ligne, l'intégration d'un système de paiement pour les pénalités de retard, ou l'automatisation des notifications pour les retours.

6. Annexes

6.1. Justification de l'Outil de Modélisation

- **Outils Utilisés** : Pour la réalisation des diagrammes UML de ce projet, nous avons principalement utilisé **Modelio** et **StarUML**.
- **Justification du Choix** :
 - **Modelio** a été choisi pour sa robustesse et sa conformité aux standards UML, offrant des fonctionnalités avancées pour la modélisation d'entreprise et la génération de code, ce qui a facilité une approche structurée de la conception.

- **StarUML** a été privilégié pour sa légèreté, sa facilité d'utilisation et son interface intuitive, ce qui a permis aux membres du groupe de collaborer efficacement sur les diagrammes sans courbe d'apprentissage abrupte, notamment pour les premières ébauches et les révisions rapides. Ces outils ont grandement facilité la réalisation de notre projet en nous permettant de créer des diagrammes clairs et conformes aux spécifications UML.