|  |  |
| --- | --- |
|  | Chapitre 1**Présentation du cadre du projet** |

***Introduction***

Dans ce chapitre, nous présenterons le cadre générale du projet et JBA expertise, la société hôte de notre projet de fin d’études. Et nous ferons une étude et une critique de l’existants qui ont menée a la naissance de la solution a proposer dans le cadre de projet.

***I. Cadre générale du projet***

Ce travail s’inscrit dans le cadre du projet de fin d’études à l’Institut Supérieur des Etudes Technologiques de Bizerte (ISET). Il vient compléter notre formation universitaire acquise au sein de cet établissement, en vue de nous introduire dans la vie professionnelle grâce à la réalisation d’un projet réel dans lequel nous metterons en œuvre nos connaissances acquises en développement informatique.

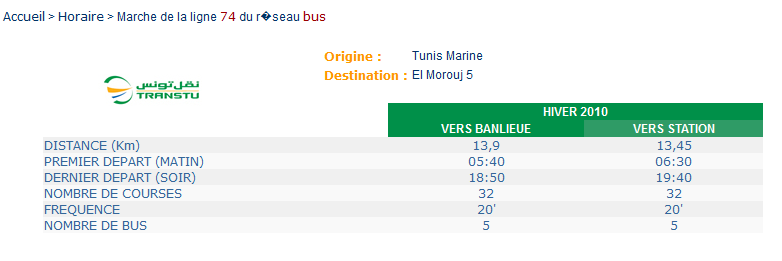
***II. Présentation de l’entreprise d’accueil***

***JBA expertise*** est une entreprise unipersonnelle Fondée en avril 2012 au capital de 1 EURO par Jawhar BEN ABDALLAH, Chef de projet et expert dans l’étude et la réalisation des systèmes d’information. Son siège est immatriculé auprès des greffes et tribunaux de la ville de Nanterre***,*** spécialisée dans les développent et conseil pour la mise en place des plateformes basées sur les technologies Java/J2EE.

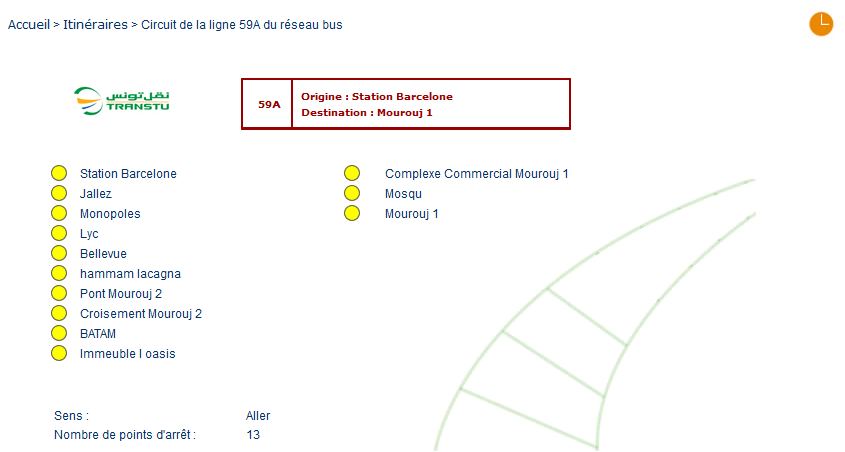
***III. Etude de l’existant***

***III.1 Description de l’existant :***

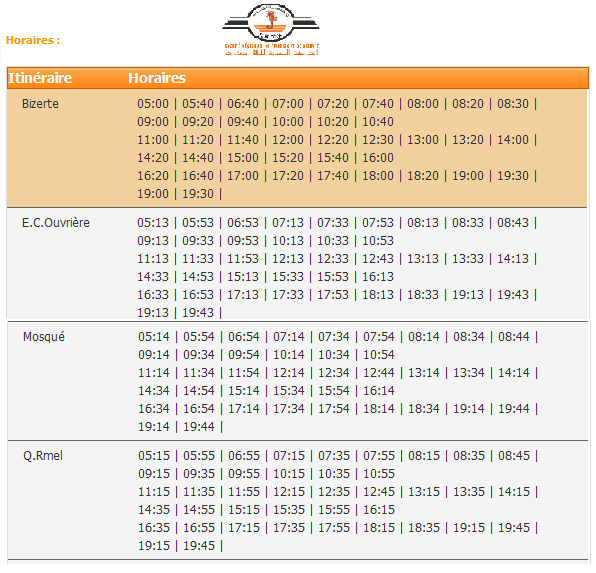
De nos jours les sociétés de transports informent sa clientèles des horaires et des itinéraires des voyages via les voies classiques tels que les guides, les affichages papier dans les stations, leurs sites web, ou bien s’adresser au guichet.



**Capture d'écran de l'application TRANSTU (Horaire du passage)**



**Capture d'écran de l'application TRANSTU (Itinéraire)**



**Capture d'écran de l'application SRTB (Horaires des passages)**

***III.2 Critique de l’existant***

Les affichages en papiers et les guides sont les moyens les plus répandus que les sociétés de transports utilisent pour informer leurs clients des horaires des voyages. Ces moyens se caractérisent par leur cout élevé (cout d’impression) et la non fiabilité des résultats car les informations seront figées à l’ instant de l’impression et ne reflétant pas la réalité.

Suite à l’évolution des technologies de l’information les societes de transports essaient de plus en plus d’être plus proche de leurs clients en leurs fournissant l’information via leur sites d’internet. Mais nous remarquons que dans la plus part des cas que les informations sont obsolètes et dépassés par les événements faute de mise à jour ce qui ne servira pas grand-chose aux clients et ne donnera pas satisfaction.

***III.3 Solution proposée***

Tenant compte des critiques présentés précédemment, nous avons pensé à développer un système d’information en ligne pour les sociétés de transports en commun qui répond aux objectifs et qui pallie aux lacunes constatées au niveau du processus existant.

Ce système va faciliter la tâche au passager et lui permet d’optimiser ses déplacements et ses voyages à travers plusieurs nouveaux services, tels que les horaires des prochains passages à un point d’arrêt en temps réel, le suivi des retards, les calculs des itinéraires et l’abonnement au service des alertes pour recevoir des informations concernant les perturbations sur le réseau de transport.

Ce système vapermettre à l’administrateur de la société de gérer les données du réseau de transport : tels que la gestion des stations, la gestion des lignes, la gestion des parcours, la gestion des alertes, et la supervision du trafic du réseau.

***IV. Méthodologie adoptée***

Pour bien mener notre projet il s'avère nécessaire, de suivre une méthodologie de développement. Notre choix s’est porté vers la processus unifié(PU ou UP en anglais pour Unified Process) qui est une démarche de développement logiciel orienté objet.il s’adresse au membre de l’équipe de développement chargée des activités du cycle de vie qui sont la formulation des besoins, l’analyse, la conception et les tests, d’autre termes des activités produisent des modèle UML. Le processus unifié est pilotée par les cas d’utilisation, l’itération, l’incrémentation, centré sur l’architecture et utilise le langage UML.

Le processus unifié comporte quatre phases qui sont :

* Préetude (Inception) : c’est ici que nous faisons une étude de faisabilité en évaluant la valeur ajoutée du développement et la capacité technique à le réaliser.
* Elaboration : au niveau de cette phase nous confirmons l’adéquation du système aux besoins des utilisateurs et nous livrons l’architecture de base.
* Construction : dans cette phase nous produisons progressivement les différentes fonctions étudiés du système.
* Transition : à la phase finale le produit est livré en version bêta à la disposition des utilisateurs.

***IV.2 Langage de modélisation UML***

Pour concevoir notre application nous avons choisi le langage de modélisation UML (Unified Modelling Language) vu qu’il est standard et peut être appliqué à toutes les méthodes de développement orientées objet. Il se base sur un ensemble de diagrammes allant des cas d’utilisation qui sert à exprimer les besoins de l’utilisateur jusqu’aux diagrammes de déploiement qui illustrent le déploiement physique des composant logiciels et matériels du système.

***V. Planification du projet :***

La planification est parmi les phases d'avant-projet. Elle consiste non seulement à délimiter le périmètre temporel du projet, mais aussi à prévoir le déroulement des activités tout au long de la période allouée au stage.

Les étapes de travail que nous avons suivi :

* Etude du besoin et rédaction d’un document de spécifications des travaux à réaliser
* Définition des cas d’utilisation
* Conception des écrans de l’application
* Conception des données
* Développement et tests unitaires
* Rédaction du rapport du stage

Ci-dessous le chronogramme de la répartition des semaines entre les différentes étapes d’élaboration du projet.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Mois | Février | | | | Mars | | | | Avril | | | | Mai | | | | Juin | |
| Semaines | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Etude de l’existant |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Spécification des besoins |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Conceptions des écrans |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Conceptions des données |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Développement et tests unitaires |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Rédaction du rapport du stage |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

***Conclusion***

Ce chapitre a été consacré à la spécification du projet, à travers lequel nous avons présenté son cadre général ainsi une description et critique des solutions existantes utilises par les sociétés des transports en communs.

Le chapitres suivant sera consacres à la spécification des besoins du projet.

.

|  |  |
| --- | --- |
|  | Chapitre 2**Spécification des besoins** |

***Introduction***

La spécification est la première étape dans un projet. Cette étape est déterminante pour le bon déroulement du projet.

Elle consiste à connaitre le travail demandé et les différents problèmes, posés par le sujet du point de vue organisationnel et technique.

Au niveau de ce chapitre nous allons spécifier en détail les différents besoin fonctionnels et non fonctionnels de notre système d’information, ainsi que les différents acteurs et les cas d’utilisation.

***I. Les besoins fonctionnels***

Dans cette partie, nous exposons l’ensemble des besoins fonctionnels auxquels devraient répondre notre application.

* **Gestion des Station :** Cette fonctionnalité permet à l’administrateur de la société de transport d’ajouter, de modifier, et de supprimer des stations
* **Gestion des Lignes :** Cette fonctionnalité permet à l’administrateur de la société de transport d’ajouter, de modifier, et de supprimer des lignes
* **Gestion des parcours :** Cette fonctionnalité permet à l’administrateur de la société de transport d’ajouter, de modifier, et de supprimer des parcours
* **Gestion des programmes des voyages :** Cette fonctionnalité permet à l’administrateur de la société de transport d’ajouter, de modifier, et de supprimer des programmes des voyages
* **Gestion des conducteurs :** Cette fonctionnalité permet à l’administrateur de la société de transport d’ajouter, de modifier, et de supprimer des conducteurs
* **Gestion des véhicules :** Cette fonctionnalité permet à l’administrateur de la société de transport d’ajouter, de modifier, et de supprimer des véhicules
* **Alerter les perturbations sur le réseau :** Cette fonctionnalité permet à l’administrateur de la société de transport d’informer ses clients voyageurs de l’état du trafic réseau en cas des perturbations.
* **Suivre le trafic réseau :** Cette fonctionnalité permet à l’administrateur de suivre le comportement des bus sur le réseau de transport.
* **Rechercher itinéraire :** Cette fonctionnalité offre au voyageur la possibilité de rechercher le plus court chemin entre deux emplacement.
* **S’informer sur le prochain passage à une station :** Cette fonctionnalité offre au voyageur la possibilité de savoir l’heure des prochains passages du véhicule de transport à une station choisie.
* **Visualiser les points d’arrêt sur GMap :** Cette fonctionnalité offre possibilité au voyageur la de géolocaliser les points d’arrêts sur Google Map
* **Recevoir des alertes concernant les perturbations sur les lignes :** Cette fonctionnalité offre au voyageur la possibilité de choisir ses parcours préférés et recevoir des alerte en fonction de l‘état du trafic sur le réseau de la société

***II. Les besoins non fonctionnels***

Les besoin non fonctionnels représentent les exigences implicites auxquelles le système doit répondre. Parmi ces besoins on cite :

* La rapidité de traitement : Vu que le besoin de trouver une réponse dans les brefs délais, il est impérativement nécessaire que la durée d’exécution des traitements s’approche le plus possible du temps réel.
* La performance et l’utilité : c'est-à-dire donnant des résultats utiles, précis et exacts
* L’ergonomie de l’application : l’application doit présenter des interfaces simples afin que l’utilisateur puisse naviguer d’un écran vers l’autre sans aucune difficulté.
* La maintenabilité : Le code doit être suffisamment clair pour permettre de futures évolutions et améliorations.

***III. Les diagrammes de cas d’utilisation***

Les cas d'utilisation permettent de représenter le fonctionnement du système vis-à-vis de son environnement extérieur. Le diagramme de cas d’utilisation se compose de :

* Acteurs : ce sont les entités externes (personne humaine ou robot) qui utilisent le système.
* Cas d’utilisation : ce sont les fonctionnalités proposées par le système.

***III.1 présentation des acteurs***

Les acteurs principaux avec lesquels interagit le système sont :

* **Administrateur** : chargé principalement de la gestion des stations, des lignes des parcours, des programmes, et les déclarations des alertes.
* **Voyageur** : consulte l'application sans inscription pour utiliser toutes les fonctionnalités permettant de faciliter ses déplacements, à savoir la localisation des stations, la recherche des plus rapides itinéraires, s’informer sur le prochain passage à une station, et l’inscription au service des alertes.

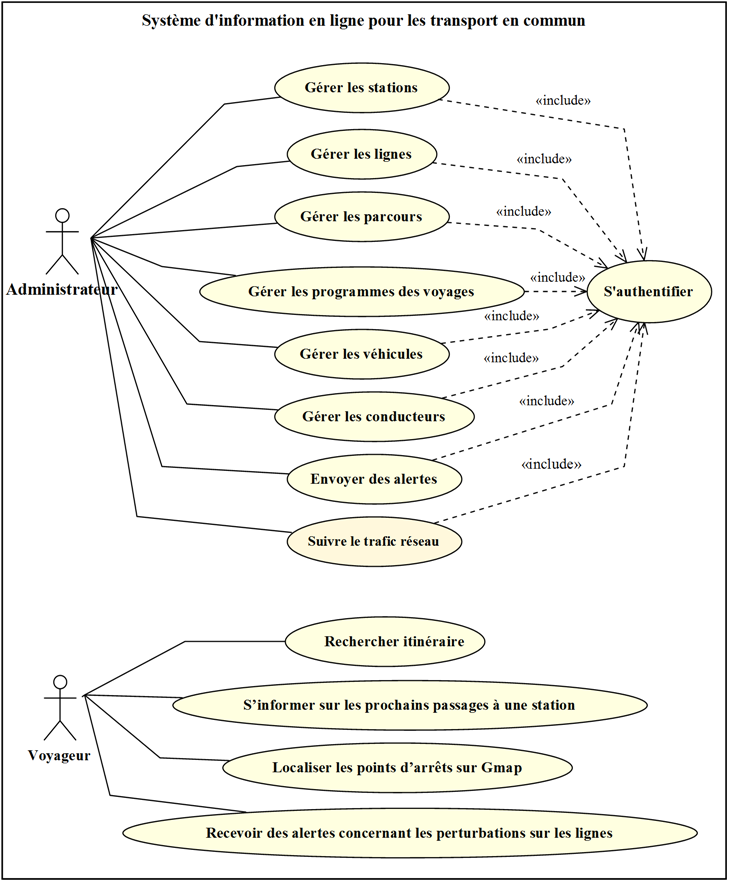
***III.2 Les cas d'utilisation***

Durant notre étude nous avons identifié les différents cas d’utilisation pour chaque acteur du système.

Le tableau ci-dessous contient une classification des cas d’utilisation par acteur :

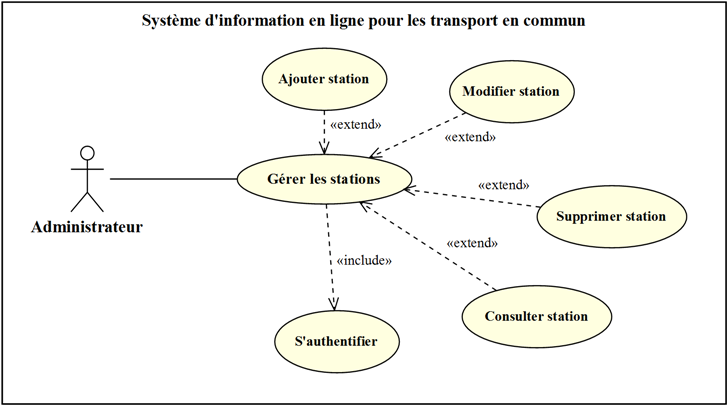
|  |  |
| --- | --- |
| **Acteur** | **Cas d’utilisation** |
| Administrateur | * Gérer les stations * Gérer les lignes * Gérer les parcours * Gérer les programmes des voyages * Gérer les véhicules * Gérer les conducteurs * Envoyer des alertes * Suivre le trafic réseau |
| Voyageur | * Rechercher itinéraire * S’informer sur les prochains passages à une station * Localiser les points d’arrêts sur Gmap * Recevoir des alertes concernant les perturbations sur les lignes |

III.2.1 Diagramme de cas d'utilisation globale



**Figure : Diagramme des cas d’utilisation général**

***III.2.2 Raffinement de cas d’utilisation « Gérer les stations »***

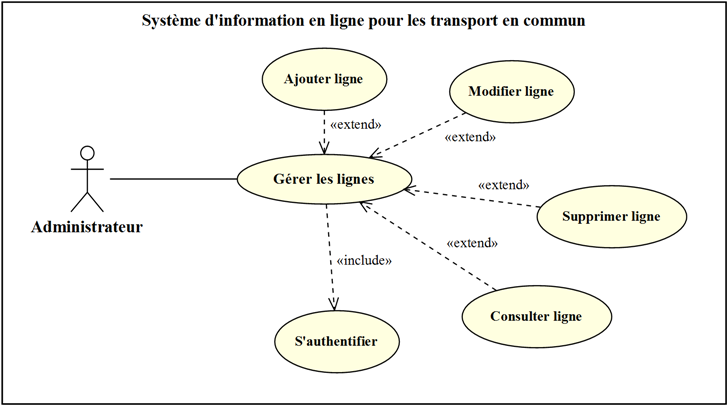


**Figure : Diagramme des cas d’utilisation « *Gérer les stations »***

Description textuelle :

|  |  |
| --- | --- |
| Titre | Gérer les stations |
| Acteur | Administrateur |
| Objectif | Ajouter, Modifier, Supprimer des stations |
| Scénario nominal | 1. L’administrateur consulte l’application.  2. l’administrateur accède au menu station.  3. L’administrateur choisir l’opération à effectuer : Ajouter, modifier, supprimer, ou simple consultation.  4. En cas d’ajout ou de modification d’une station le système affiche un formulaire que l’administrateur doit le remplir.  5. En cas de suppression l’utilisateur doit cliquer sur le bouton de suppression.  6. En cas de consultation le système affiche une liste qui contient tout les stations. |

***III.2.3 Raffinement de cas d’utilisation « Gérer les lignes »***

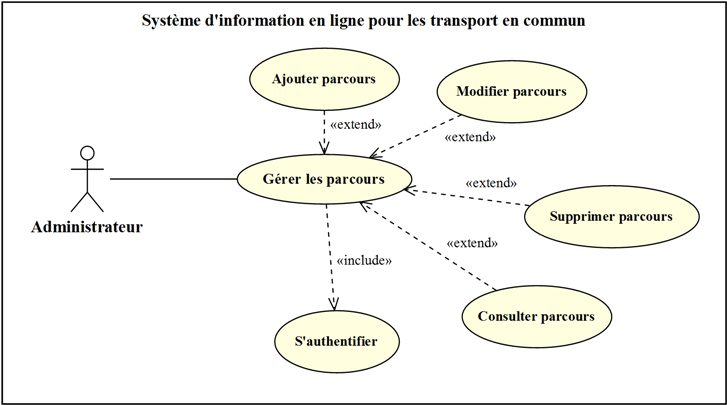
******

**Figure : Diagramme des cas d’utilisation « *Gérer les lignes »***

Description textuelle :

|  |  |
| --- | --- |
| Titre | Gérer les lignes |
| Acteur | Administrateur |
| Objectif | Ajouter, Modifier, Supprimer des lignes |
| Scénario nominal | 1. L’administrateur consulte l’application.  2. l’administrateur accède au menu ligne.  3. L’administrateur choisir l’opération à effectuer : Ajouter, modifier, supprimer, ou simple consultation.  4. En cas d’ajout ou de modification d’une ligne le système affiche un formulaire que l’administrateur doit le remplir.  5. En cas de suppression l’utilisateur doit cliquer sur le bouton de suppression.  6. En cas de consultation le système affiche une liste qui contient tout les lignes. |

***III.2.4 Raffinement de cas d’utilisation « Gérer les parcours »***

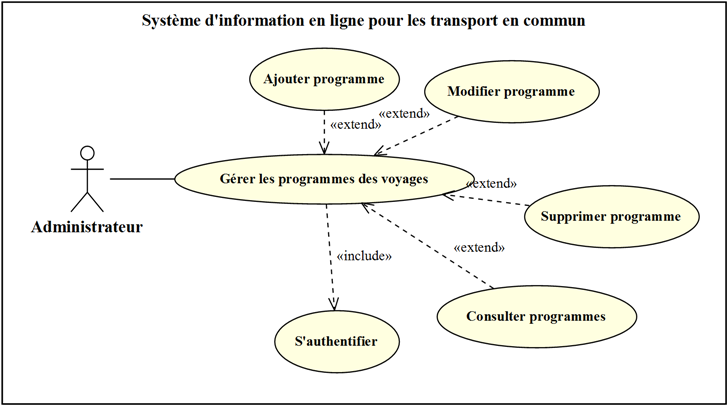
******

**Figure : Diagramme des cas d’utilisation « *Gérer les parcours »***

Description textuelle :

|  |  |
| --- | --- |
| Titre | Gérer les parcours |
| Acteur | Administrateur |
| Objectif | Ajouter, Modifier, Supprimer des parcours |
| Scénario nominal | 1. L’administrateur consulte l’application.  2. l’administrateur accède au menu parcours.  3. L’administrateur choisir l’opération à effectuer : Ajouter, modifier, supprimer, ou simple consultation.  4. En cas d’ajout ou de modification l’administrateur doit sélectionner tout d’abord les stations et définir les différents délais du parcours.  5. En cas de suppression l’utilisateur doit cliquer sur le bouton de suppression.  6. En cas de consultation le système affiche une liste qui contient tout les parcours. |

***III.2.5 Raffinement de cas d’utilisation « Gérer les programmes»***

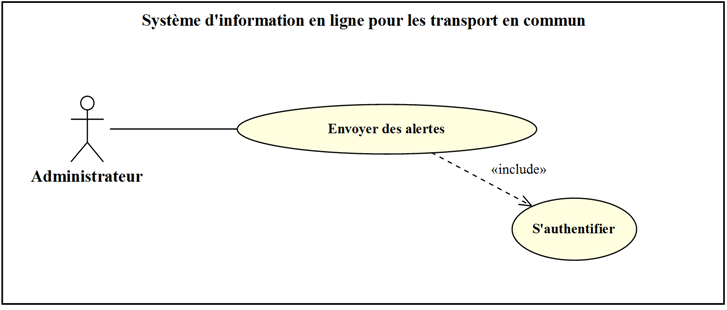
******

**Figure : Diagramme des cas d’utilisation « *Gérer les programmes »***

Description textuelle :

|  |  |
| --- | --- |
| Titre | Gérer les programmes |
| Acteur | Administrateur |
| Objectif | Ajouter, Modifier, Supprimer des programmes |
| Scénario nominal | 1. L’administrateur consulte l’application.  2. l’administrateur accède au menu programmes.  3. L’administrateur choisir l’opération à effectuer : Ajouter, modifier, supprimer, ou simple consultation.  4. En cas d’ajout ou de modification d’un programme le système affiche un formulaire que l’administrateur doit le remplir.  5. En cas de suppression l’utilisateur doit cliquer sur le bouton de suppression.  6. En cas de consultation le système affiche une liste qui contient tout les programmes. |

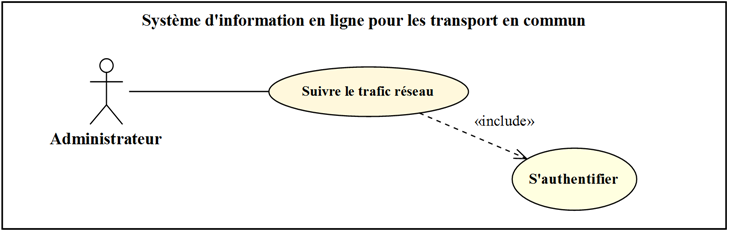
***III.2.6 Raffinement de cas d’utilisation « Envoyer des alertes»***

******

**Figure : Diagramme des cas d’utilisation « *Envoyer des alertes »***

Description textuelle :

|  |  |
| --- | --- |
| Titre | Envoyer des alertes |
| Acteur | Administrateur |
| Objectif | Envoyer des alertes au voyageur |
| Scénario nominal | 1. L’administrateur consulte l’application.  2. l’administrateur accède au menu alerte.  3. L’administrateur doit choisir le parcours concernés par l’alerte et écrire une description pour plus détailler la cause de l’alerte au voyageur.  4. Après la validation un e-mail sera envoyé au voyageur. |

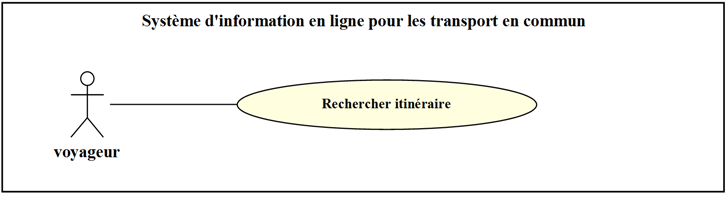
***III.2.6 Raffinement de cas d’utilisation «Suivre trafic réseau»***

**Figure : Diagramme des cas d’utilisation « *Suivre le trafic réseau»***

Description textuelle :

|  |  |
| --- | --- |
| Titre | Suivre le trafic réseau |
| Acteur | Administrateur |
| Objectif | Suivre les déplacements des bus sur le reseau du transport |
| Scénario nominal | 1. L’administrateur consulte l’application.  2. l’administrateur accède au menu supervision.  3. le système affiche tout les événements d’entré ou de sortie aux stations |

***III.2.7 Raffinement de cas d’utilisation «Rechercher itinéraire»***

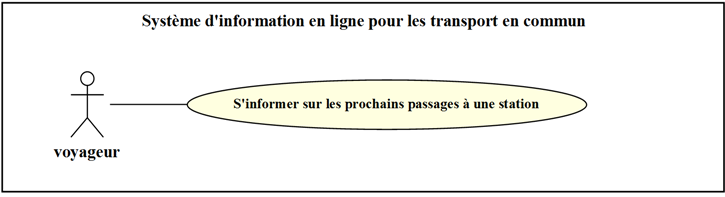
******

**Figure : Diagramme des cas d’utilisation « *Rechercher itinéraire»***

Description textuelle :

|  |  |
| --- | --- |
| Titre | Rechercher itinéraire |
| Acteur | Voyageur |
| Objectif | Rechercher le chemin le plus rapide entre deux stations |
| Scénario nominal | 1. Le voyageur consulte l’application.  2. Le voyageur choisie la fonctionnalité de recherche d’itinéraire.  3. Le voyageur choisie la station de départ, la station d’arrivée et la date de départ  4. Le système détermine le chemin le plus rapide entre les deux stations choisi en fonction des voyage programmé et affiche le résultat. |

***III.2.8 Raffinement de cas d’utilisation «S’informer sur les prochains passage a une station»***

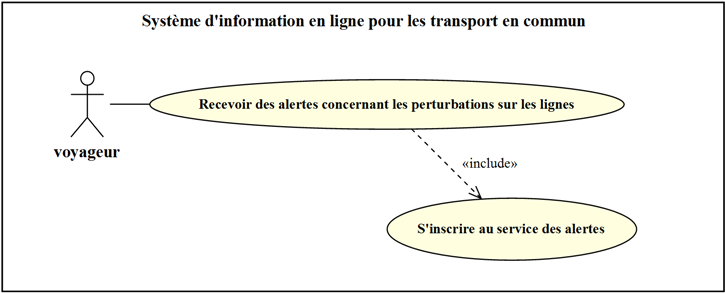
******

**Figure : Diagramme des cas d’utilisation «S’informer sur les prochains passage à une station»**

Description textuelle :

|  |  |
| --- | --- |
| Titre | S’informer sur les prochains passages a une station |
| Acteur | Voyageur |
| Objectif | S’informer sur l’heure de passage du véhicule de transport a une station choisie par le voyageur. |
| Scénario nominal | 1. Le voyageur consulte l’application.  2. Le voyageur choisie la fonctionnalité prochain passage.  3. Le voyageur choisi une station.  4. Le système recherche les lignes qui passent par la station choisie et affiche le résultat au voyageur.  5. Le voyageur choisi la ligne et sa destination  6. Le système calcule l’heure des prochains passages et affiche le résultat au voyageur |

***III.2.9 Raffinement de cas d’utilisation «Recevoir des alertes concernant les perturbations sur les lignes»***

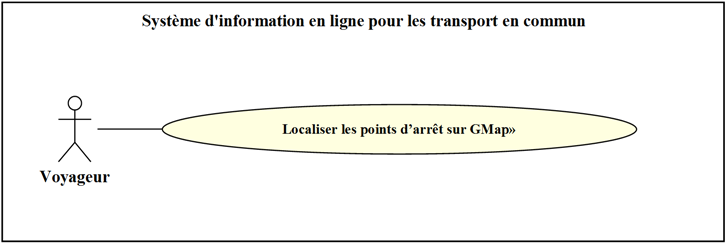
******

**Figure : Diagramme des cas d’utilisation «Recevoir des alertes concernant les perturbations sur les lignes»**

Description textuelle :

|  |  |
| --- | --- |
| Titre | Recevoir des alertes concernant les perturbations sur les lignes |
| Acteur | Voyageur |
| Objectif | Le voyageur reçoit des alertes à tout moment de perturbation ou des incidents sur ses lignes préférer |
| Scénario nominal | 1. Le voyageur consulte l’application.  2. Le voyageur choisi une ligne auquel veut recevoir des alertes  3. Le système envoie un e-mail au voyageur au moment d’une perturbation sur la ligne selectionné |

***III.2.10 Raffinement de cas d’utilisation «Localiser les points d’arrêt sur GMap»***

******

**Figure : Diagramme des cas d’utilisation «Localiser les points d’arrêt sur GMap»**

Description textuelle :

|  |  |
| --- | --- |
| Titre | **Localiser les points d’arrêt sur GMap** |
| Acteur | Voyageur |
| Objectif | Le voyageur peut géolocaliser les stations autour d’une point choisie |
| Scénario nominal | 1. Le voyageur consulte l’application.  2. Le voyageur choisi la fonctionnalité de localiser station.  3. Le système marque les points d’arrêt sur Google map |

Conclusion

Après avoir achevé la phase de spécification des acteurs ainsi que la description des cas d’utilisation, nous entamons la phase de conception de notre projet en s’appuyant sur le formalisme du langage UML.

|  |  |
| --- | --- |
|  | Chapitre 3**Conception** |

***Introduction***

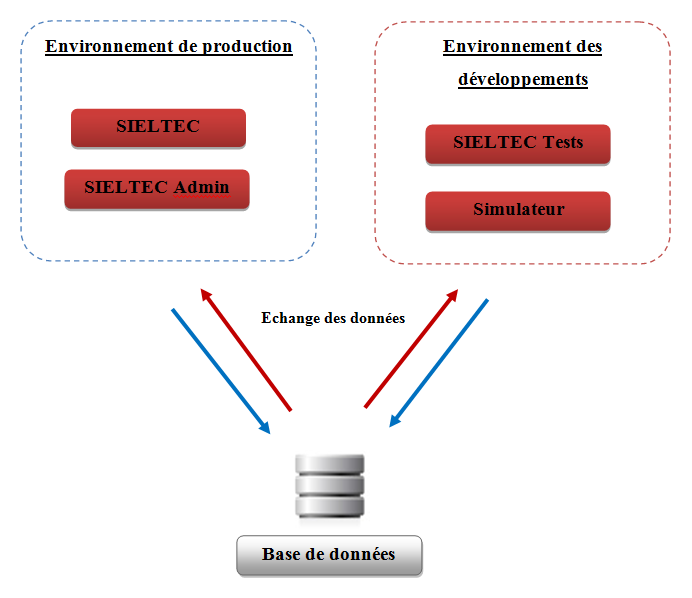
Ce chapitre se consacre en premier lieu, à présenter l’architecture globale à adopter pour la mise en place de notre système

La deuxième partie constitue une suite du chapitre précédent, afin d'enrichir l'analyse fonctionnelle effectuée auparavant à l'aide des diagrammes de cas d'utilisation UML. Les parties qui suivent présentent quelques diagrammes de séquences et le diagramme de classes.

***I. Architecture de la solution***

Notre système d’information est basé sur une architecture J2EE son développement a été pensés dés le début pour être le plus modulaire possible.

Ci-dessous une vue globale de notre système :



**Figure : Vue d’ensemble du système**

**SIELTEC** est une application web J2EE destinée au passager qui lui présente des fonctionnalités avancées afin de lui faciliter ses déplacements.

**SIELTEC Admin** est une application web J2EE destiné à l’administrateur de l’entreprise de transport qui lui permet la gestion des éléments du réseau et le suivie du trafic de transport.

**SIELTEC Test** est composant qui nous permet de tester et de s’assurer de la non régression des certaines fonctionnalités du notre système

**Le simulateur** est une application java autonome (qui ne tourne pas sur un serveur web) qui est un dispositif permettant de reproduire d’une façon virtuelle le comportement réel des moyens de transports sur le réseau du transport.

***II. Présentation de chaque élément de l’architecture***

***I.SIELTEC***

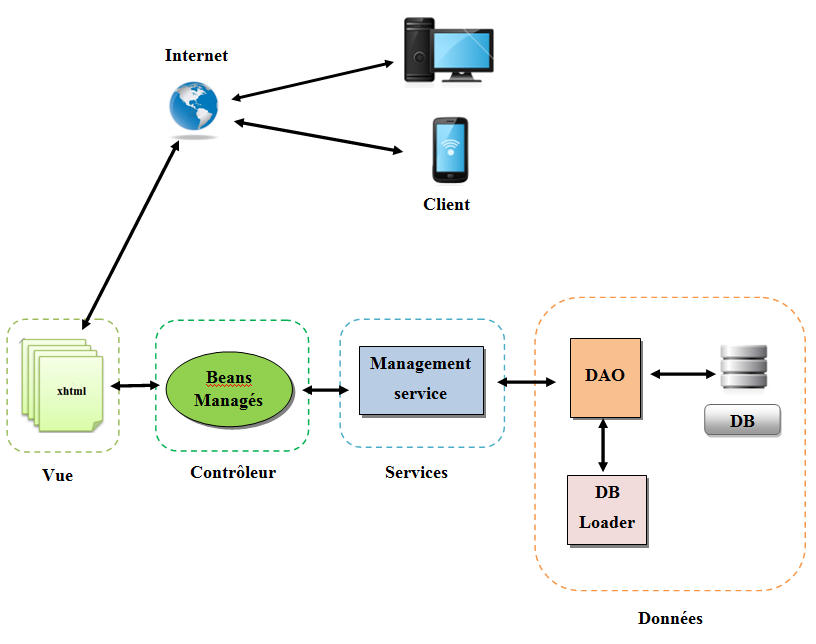
SIELTEC est une application web dynamique JEE qui tourne sur le serveur d’application Tomcat dans sa version 7.0.34 et avec la base de données apache Derby

Nous avons utilisé JSF 2.0 ainsi que la bibliothèque open source OpenFaces dans sa version 3.0 pour les interfaces graphiques.

Cette application est dédié au voyageur et lui offre plusieurs services afin de lui aider à se déplacer tel que :

* Recherche d’itinéraire
* Horaire du prochain passage à une station
* La géolocalisation des stations
* S’inscrire au service des alertes

Ci dessous un schéma récapitulatif de l’architecture de l’application :



***II.SIELTEC Admin***

Sieltec Admin est une application web dynamique JEE qui tourne sur le serveur d’application Tomcat dans sa version 7.0.34 et avec la base de données apache Derby

Nous avons utilisé JSF 2.0 ainsi que la bibliotheque open source OpenFaces dans sa version 3.0 pour les interfaces graphiques.

Cette application est dédié à l’administrateur de la société, elle a comme objectif d’alimenter l’application SIELTEC par les données relatif au réseau routier tel que :

* Les lignes
* Les stations
* Les parcours
* Les programmes
* etc…

Architecture de SIELTEC Admin :

Ci dessous un schéma récapitulatif de l’architecture de l’application :

***II. Conception du niveau données***