



Projet JEE août 2022

Manage police

Pirmez Alexandre
Huygebaert Gabriel

Bachelier en informatique de gestion – Bloc 3 Applications informatiques III – SGBD V

Consignes de seconde session

Énoncé

Réalisez le développement d'un site web par groupe de 2 (obligatoirement) en utilisant les technologies vues aux cours d'applications informatiques et SGBD V.

Les groupes doivent être formés au plus tard pour le 11 juillet 2022. Avant cette date, vous devez envoyer le nom des membres de votre groupe par mail. Un mail par groupe suffira. Ce mail doit être adressé aux 2 professeurs et en mettant l'autre membre en copie.

Thème imposé

La police souhaite mettre à disposition une application web pour ses équipes d'intervention afin de gérer les amendes de roulage constatées.

Un policier doit pouvoir encoder une nouvelle amende dans le système. Pour ce faire, il devra choisir le type d'infraction constatée parmi une liste d'infractions. Ensuite, il devra renseigner le type de véhicule (voiture, moto, vélo, camion, ...), le numéro de plaque du véhicule (s'il y en a une), la date et l'heure de l'infraction, le nom et le prénom de l'usager (si celui-ci est présent), ainsi qu'un commentaire.

Lorsqu'il encode le numéro de plaque, le système pourra analyser si le conducteur est en ordre d'assurance (il l'est si la plaque est renseignée dans le système). Si ce n'est pas le cas, le système devra en informer le policier qui pourra alors aussi verbaliser cette infraction. La recherche sur le numéro de plaque ne devra donc pas se faire lorsque le type d'infraction concerne un défaut d'assurance.

Un policier peut ainsi signaler une série d'infractions pour un même véhicule. Lorsqu'il a terminé d'encoder les infractions, le système calculera le montant total des amendes à régler et l'affichera pour que le policier puisse éventuellement en informer le contrevenant. Chaque type d'amende a un montant défini dans le système.

Les chefs de brigade peuvent également utiliser l'application. Celle-ci leur permet de lister les infractions déclarées par leur corps de police (les policiers dont ils sont le supérieur). Ils pourront alors passer en revue les différentes amendes déclarées et les valider ou les invalider.

Les amendes invalidées seront simplement retirées du système. Les amendes validées devront être envoyées à l'adresse du contrevenant.

Un policier dépend toujours d'un seul chef de brigade. Il y a plusieurs chefs de brigade.

Le percepteur des amendes peut également utiliser l'application pour consulter les amendes validées dans tous les corps de police afin d'envoyer le courrier au contrevenant. L'envoi ou la génération du courrier n'est pas demandé dans l'application.

Un administrateur pourra utiliser l'application afin de gérer (création, modification et suppression) les types d'amendes, les types de véhicules et les comptes des utilisateurs. Un utilisateur ne peut donc pas créer son propre compte.

L'administrateur et les chefs de brigade ont la possibilité de fixer et modifier le montant prévu pour un type d'amende.

Toutes les fonctionnalités doivent être authentifiées par un matricule et un mot de passe. Un matricule est composé de chiffres et de lettres. Voici un exemple de matricule: "BH1971234". Les mots de passe doivent être cryptés.

Contraintes techniques

Applications informatiques (JEE) :

L'application web devra être écrite pour la plateforme JEE. Elle permettra de gérer les requêtes et d'afficher les résultats. Tout appel à la base de données doit passer par les APIs (voir section suivante).

Le projet sera réalisé avec Eclipse et tournera sur un serveur d'application Tomcat 9.0.

L'application doit être réalisée à l'aide du JDK 15.0.2.

Les technologies à utiliser sont les suivantes :

- Des JSP ;
- Des Servlets ;
- Mettre en place le modèle MVC ;
- Écrire des DAO.

L'analyse UML est laissée à votre appréciation, **mais** il faudra fournir **un diagramme de classes** de votre application.

Applications informatiques (API) :

Les services web à développer sont de type REST. L'API devra accéder à une base de données (Oracle). Les informations retournées vers l'application cliente seront transmises au format JSON.

SGBD :

Votre projet devra utiliser la base de données de l'école (XE_CHAR). Vous devrez créer et documenter (schéma conceptuel/ERD) toutes les tables, index et contraintes d'intégrité nécessaires à ce projet.

Vos tables devront contenir des données de base permettant d'utiliser votre application sans devoir tout encoder au préalable.

Toutes les commandes de modifications de données de votre base de données devront passer par des procédures stockées, fonctions ou packages. Outre les commandes de sélection de données, aucune commande INSERT/UPDATE/DELETE ne devra donc être présent dans votre code java, seules les commandes SELECT pourront s'y trouver directement, mais pas obligatoirement.

Vous devrez utiliser

- des curseurs et types dérivés partout où cela est possible ;
- des tableaux et/ou collections de données ;
- des opérations en BULK ;
- des triggers et séquences pour générer tous vos ID primaires, ainsi que des clauses RETURNING INTO pour récupérer des informations après certaines commandes DML
- la gestion de toutes exceptions particulières, aussi précises que possible (le moins de OTHERS possibles)
- Le mot de passe des utilisateurs doit être crypté par une procédure de la base de données (pas par le code Java).

Remise du projet

Le projet sera à remettre pour le **mardi 16 août 2022 à 8h00**. Aucun délai supplémentaire ne sera accordé. Si le projet n'est pas rendu dans les délais, l'étudiant se verra attribuer la note de 0 pour ce travail et donc pour l'examen du cours d'applications informatiques.

Si l'application ne fonctionne pas (problème à la compilation, problème au déploiement sur le serveur, problème pour contacter la base de données, ...), l'élève recevra également la note de 0. N'oubliez donc pas, par exemple, de renseigner les JARs en chemin relatif. Il vous est conseillé de tester l'application sur d'autres ordinateurs. Rien ne doit être modifié par les professeurs dans le programme remis.

Vous devrez remettre un fichier au format .zip sur la plateforme Moodle du cours d'Applications Informatiques 3, dans la section prévue à cet effet. Ce fichier devra porter le nom de famille des membres du groupe (sous la forme : Nom1_Nom2.zip). Ce fichier .zip contiendra le projet (le workspace) et tout ce qui serait nécessaire pour son fonctionnement. Il devra également contenir un rapport **au format PDF**.

Le rapport devra contenir :

- Une description en quelques lignes de votre application,
- L'adresse de l'application (la page d'accueil du site),
- Un mode d'emploi pour l'installation de votre application (pour la partie JEE et pour la partie API),
- Le diagramme de classes UML,
- Le schéma conceptuel / ERD.

Tous les scripts de création de vos objets de base de données, les commandes d'insertion de vos données de base devront se trouver dans un ou plusieurs fichiers textes .sql, ainsi que déjà créés/exécutés, sans erreur de compilation, dans la base de données de l'école dans le schéma d'un étudiant du groupe (spécifier lequel dans le dossier).

Christian Clemmen et Laurent Masset

Explication du programme

Gestion minimaliste d'une application pour des policiers souhaitant déposer des contraventions. S'y retrouvent les quatre types de compte suivants et leurs actions. Toute action nécessite une authentification.

- Administrator

L'administrateur gère principalement des actions « créer, mettre à jour et suppression ». Notamment dans la gestion des comptes, des types de contravention et des types des véhicules. C'est également lui qui attribue un corps de police à un chef de police. Il ne peut pas accéder à la partie d'ajout de contravention, ni les consulter.

- Policeman

Le policeman est un policier lambda qui peut ajouter des contraventions. Il est obligé de choisir au minimum un type d'infraction. La date récupérée est celle du jour de l'envoi de la contravention. Il est possible qu'un accusé soit inconnu ou que le véhicule ne possède pas de plaque. Si la plaque n'existe pas, un type de faute est automatiquement ajouté « Default insurance » à la contravention. Le policeman peut également ajouter une plaque dans le système si celle-ci n'existe pas déjà ou si celle-ci n'est pas déjà attribuée à un véhicule. Il peut ajouter un véhicule et sa plaque, s'il y en a une, et des accusés, pour les récidivistes. Lorsqu'une contravention est ajoutée, le prix total calculé est affiché à l'écran durant quelques secondes.

Si l'accusé « M. xyz » est ivre au volant, a ses phares allumés en plein jour et téléphone en conduisant et ne possède pas de plaque sur son véhicule, une seule contravention peut être envoyée. Non pas autant qu'il y a de type de fautes commises.

- Chief

En outre des actions possibles pour un policeman, un chef de brigade peut consulter et accepter ou décliner une contravention que ses subordonnés ont créé. Lorsque lui ajoute une contravention au système, cette dernière est automatiquement validée. Il peut aussi modifier le prix d'un type de contravention sans nécessiter l'action d'un administrateur.

- TaxCollector

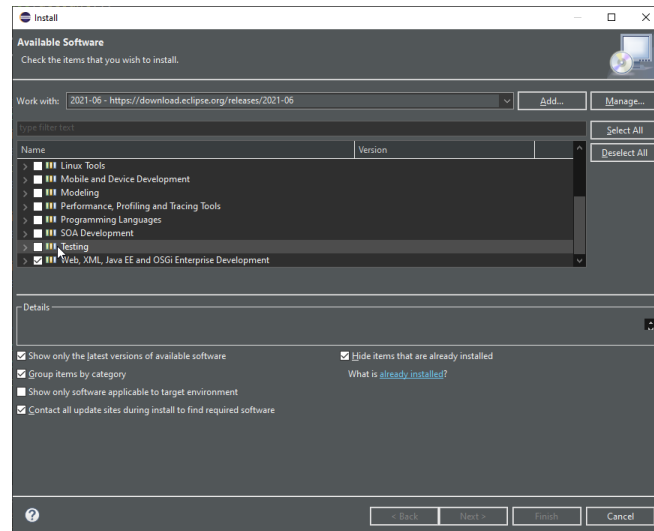
Le tax collector peut consulter les contraventions qui ont été acceptées par les chefs de police et « envoyer une lettre » à l'accusé, s'il n'est pas inconnu. L'envoi de lettre n'est pas géré par le programme mais il n'est possible d'appuyer sur le bouton d'envoi qu'une seule fois. Une extension du programme est alors possible.

Table des matières

Énoncé	1
Explication du programme	5
Façon d'utiliser le programme	7
Comptes existants	16
Lien Git.....	16
Base de données	17
Schéma conceptuel	17
PL SQL	18
DDL	18
Packages	18
Analyse	19
Use case diagram.....	19
Jet 1	19
Jet 2	20
Class diagram.....	21
Jet 1	21
Jet 2	21
Conclusion	22

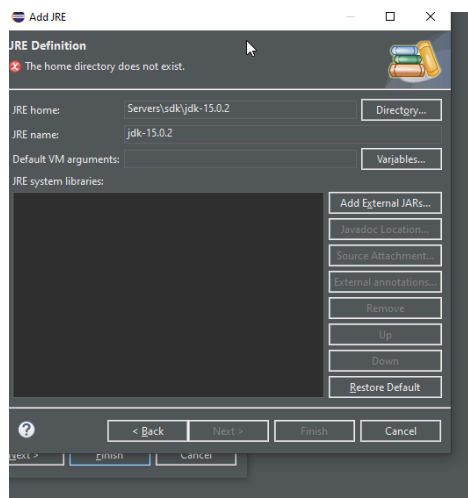
Façon d'utiliser le programme

- Importer le projet dans eclipse qui possède une perspective JEE.
 - S'il n'est pas installé : « help » -> « install new software » et cet écran apparaît.



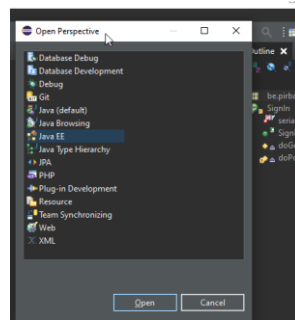
Choisir « Work with : 2021-06 – ... », cocher « WEB,XML,... » et laisser s'installer.

Les chemins relatifs ne fonctionnent pas pour le serveur, c'est pourquoi nous vous avons laissé un SDK et un tomcat dans le projet, dans le dossier Servers.

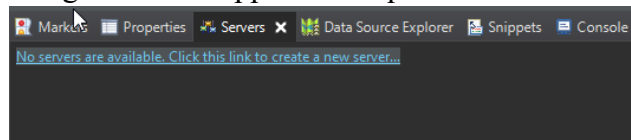


Afin de configurer le serveur, suivre ces étapes.

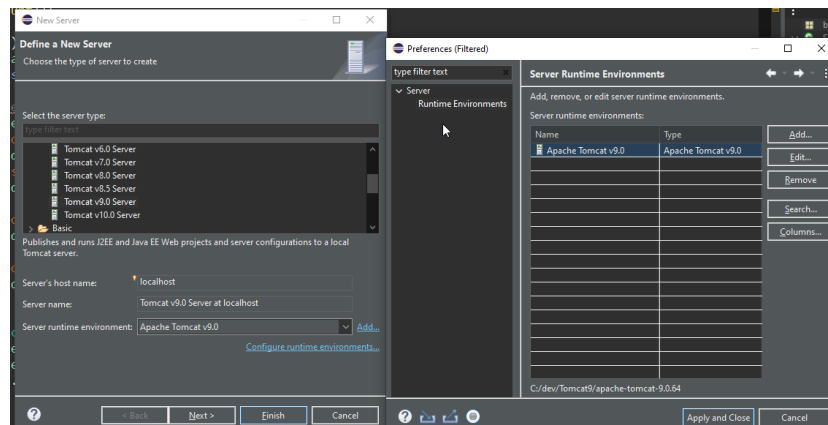
Ouvrir la perspective JEE



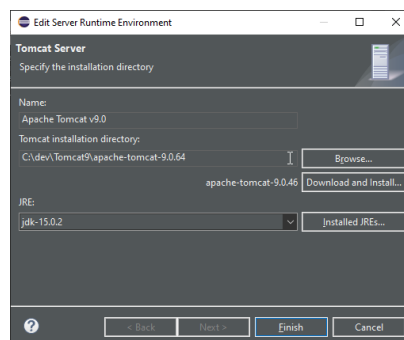
Cliquer dans l'onglet serveur apparu et cliquer sur « No Servers Available »



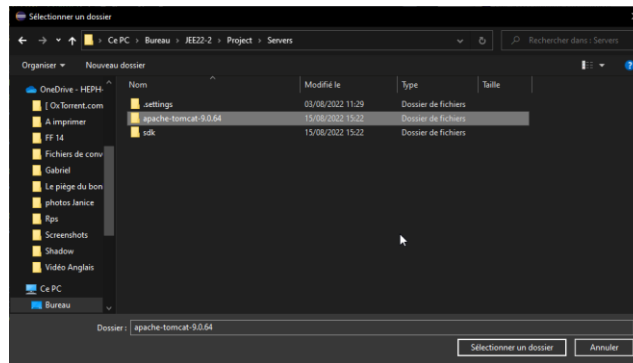
Choisir tomcat 9 dans la liste et cliquer sur « Configure runtime environments... »



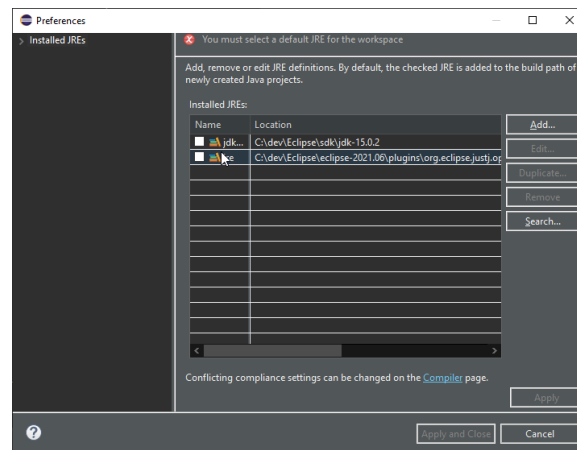
Une autre page s'ouvre. Sélectionner Apache Tomcat et edit.



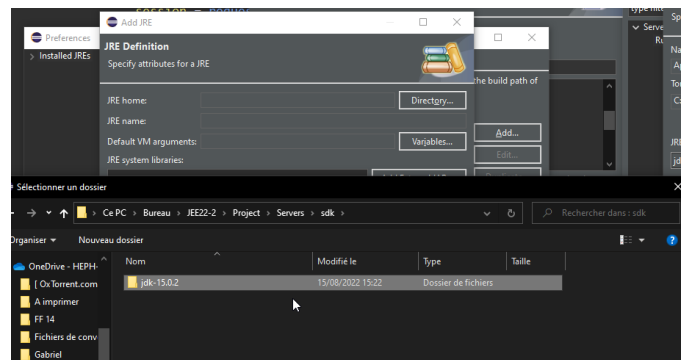
Aller chercher le dossier Tomcat qui se trouve dans « Servers » depuis cette page en cliquant sur « Browse ».



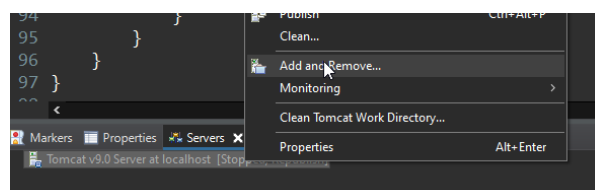
Il se peut que la JDK ne soit pas bonne. (JDK-15.0.2 est bon). Si le votre n'est pas bon, cliquer sur « Installed JREs.. ».



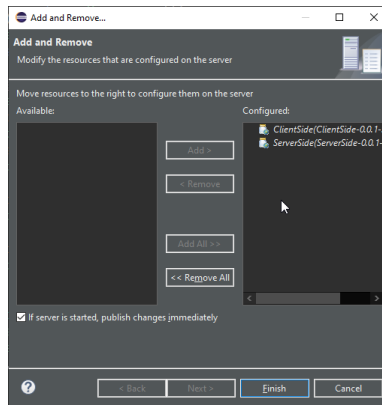
« Add » -> Standard VM -> Directory, aller chercher dans Servers.



Accepter et puis :



Ajouter le « client side » et le « server side. »



- Lancer via la servlet SignIn, depuis eclipse, dont le chemin est le suivant :

/ClientSide/src/main/java/be/pirbaert/servlets/SignIn.java

- Lancer via un URL, après avoir démarré le serveur

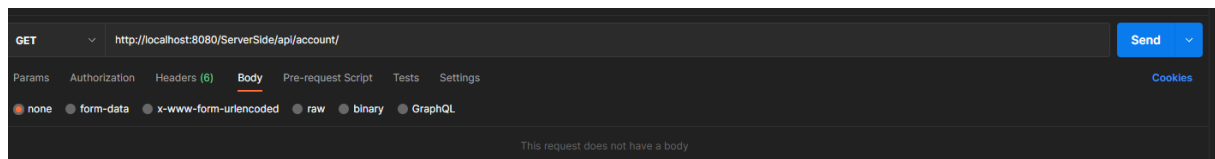
<http://localhost:8080/ClientSide/SignIn>

API

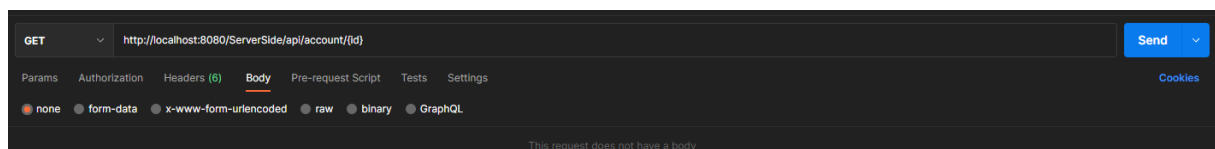
Base d'url : <http://localhost:8080/ServerSide/api/>

Account

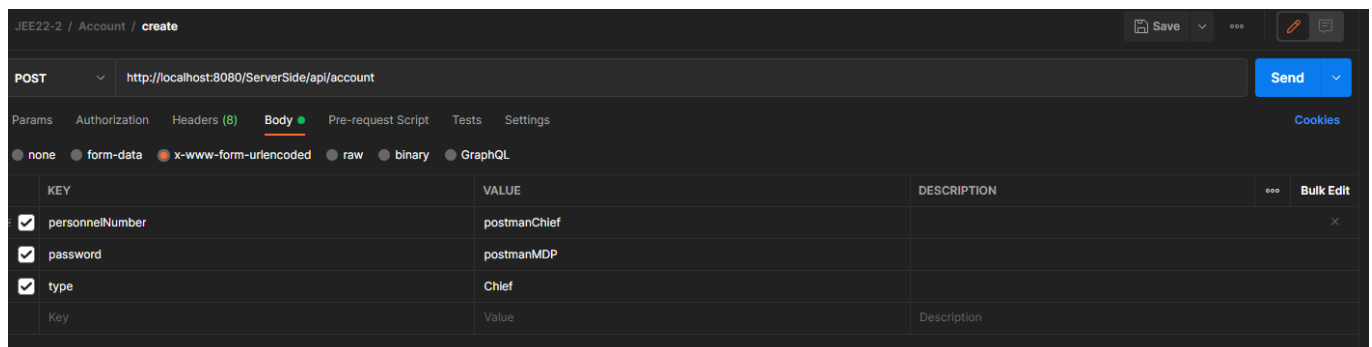
Get all account



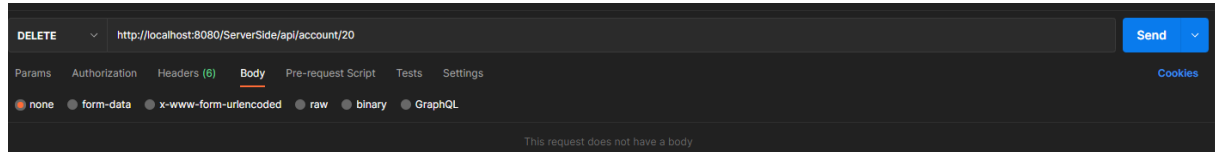
Get an account



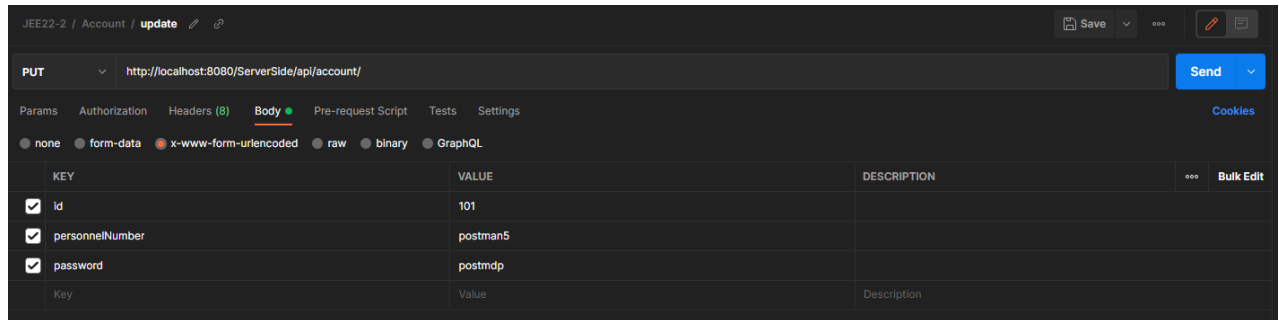
Create an account



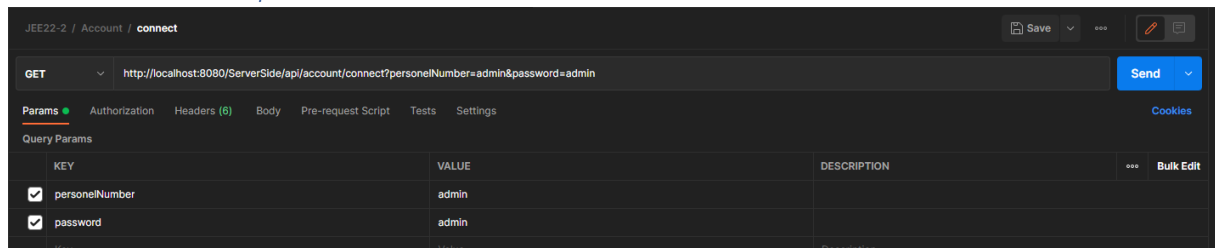
Delete an account



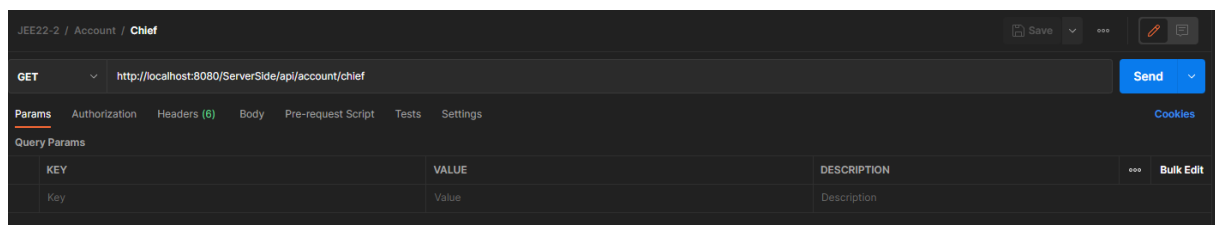
Update an account



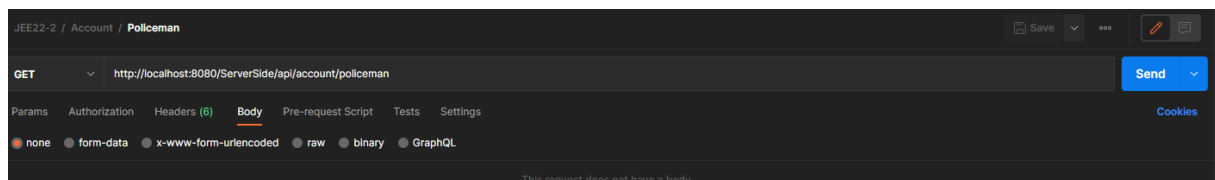
Check username and password



Get all Chief Account

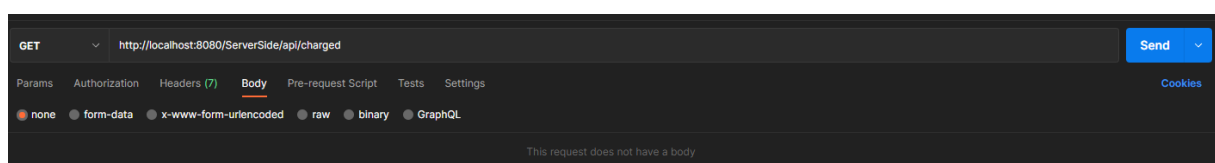


Get all Policeman Account



Charged

Get all charged



Get a charged

JEE22-2 / Charged / **Charged** Save ... ✎ 📄

GET ▼ Send ▼

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

This request does not have a body

Create a charged

JEE22-2 / Charged / **Charged** Save ... ✎ 📄

POST ▼ Send ▼

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> firstname	jhon			
<input checked="" type="checkbox"/> lastname	doe			
<input checked="" type="checkbox"/> address	rue eur			
Key	Value	Description		

Fine

Get all fines

JEE22-2 / Fine / **Fine** Save ... ✎ 📄

GET ▼ Send ▼

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Get a fine

JEE22-2 / Fine / **Fine** Save ... ✎ 📄

GET ▼ Send ▼

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Delete a fine

JEE22-2 / Fine / **Fine** Save ... ✎ 📄

DELETE ▼ Send ▼

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Create a fine

JEE22-2 / Fine / **Fine** Save ... Edit Send

POST ▼ http://localhost:8080/ServerSide/api/fine Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> Ids_violation				
<input checked="" type="checkbox"/> id_police				
<input checked="" type="checkbox"/> id_vehicle				
<input checked="" type="checkbox"/> comment				
<input checked="" type="checkbox"/> date				
<input checked="" type="checkbox"/> id_charged				
<input checked="" type="checkbox"/> id_charged				
Key	Value	Description		

Update a fine

JEE22-2 / Fine / **Fine** Save ... Edit Send

PUT ▼ http://localhost:8080/ServerSide/api/fine Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> id_fine				
<input checked="" type="checkbox"/> validated				
<input checked="" type="checkbox"/> letterSent				
Key	Value	Description		

Registration

Get registration

JEE22-2 / Registration / **Registration** Save ... Edit Send

GET ▼ http://localhost:8080/ServerSide/api/registration/ Send

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Get a registration

JEE22-2 / Registration / **Registration** Save ... Edit Send

GET ▼ http://localhost:8080/ServerSide/api/registration/{id} Send

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Delete a registration

JEE22-2 / Registration / **Registration** Save ... Edit Send



DELETE ▼ http://localhost:8080/ServerSide/api/registration/{id} Send

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Create a registration

JEE22-2 / Registration / **Registration**  



POST ▼ http://localhost:8080/ServerSide/apl/registration/ Send ▼

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> serialNumber				
Key	Value	Description		

Update a registration

JEE22-2 / Registration / **Registration**  

PUT ▼ http://localhost:8080/ServerSide/apl/registration/ Send ▼



Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> id				
<input checked="" type="checkbox"/> serialNumber				
Key	Value	Description		

Vehicle type

Get a Vehicle type

JEE22-2 / TypeVehicle / **TypeVehicle**  



GET ▼ http://localhost:8080/ServerSide/apl/typeVehicle Send ▼

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Create a Vehicle type

JEE22-2 / TypeVehicle / **create**  



POST ▼ http://localhost:8080/ServerSide/apl/typeVehicle Send ▼

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> name	POSTMAN			
Key	Value	Description		

Delete a Vehicle type

JEE22-2 / TypeVehicle / **delete**  

DELETE ▼ http://localhost:8080/ServerSide/apl/typeVehicle/10 Send ▼

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Update a Vehicle type

JEE22-2 / TypeVehicle / **update** Save ... Send

PUT http://localhost:8080/ServerSide/api/typeVehicle Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> id	10			
<input checked="" type="checkbox"/> name	bob			
Key	Value	Description		

Vehicle

Get all vehicle

JEE22-2 / Vehicle / **Vehicle** Save ... Send

GET http://localhost:8080/ServerSide/api/vehicle Send

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Create a vehicle

JEE22-2 / Vehicle / **Vehicle** Save ... Send

POST http://localhost:8080/ServerSide/api/vehicle Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> id_type				
<input checked="" type="checkbox"/> id_registration				
Key	Value	Description		

Violation

Get all violation

JEE22-2 / Violation / **Violation** Save ... Send

GET http://localhost:8080/ServerSide/api/violation Send

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Create a violation

JEE22-2 / Violation / **create** Save ... Send

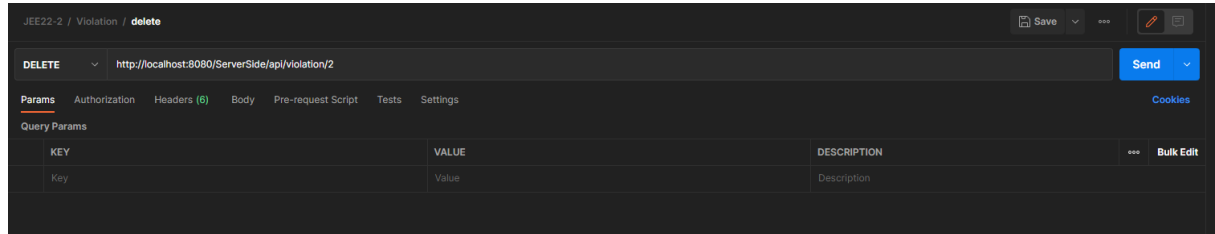
POST http://localhost:8080/ServerSide/api/violation Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

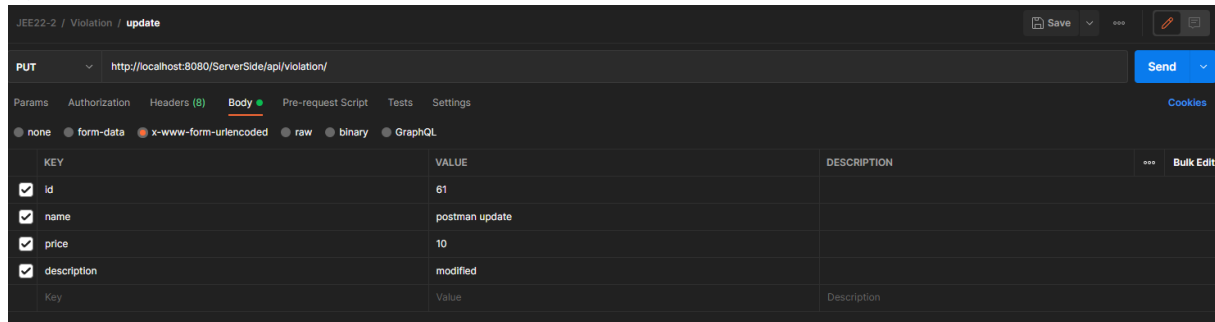
☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> name	postman			
<input checked="" type="checkbox"/> price	1000			
<input checked="" type="checkbox"/> description	create with postman			
Key	Value	Description		

Delete a violation



Update a violation



Comptes existants

- Administrator

ID : admin

MDP : admin

- Chief
- 1.

ID : Chief0

MDP : 012345

2.

ID : Chief1

MDP : 012345

- Policeman

ID : Policeman0

MDP : pm0

- TaxCollector

ID : tx0

MDP : 012345

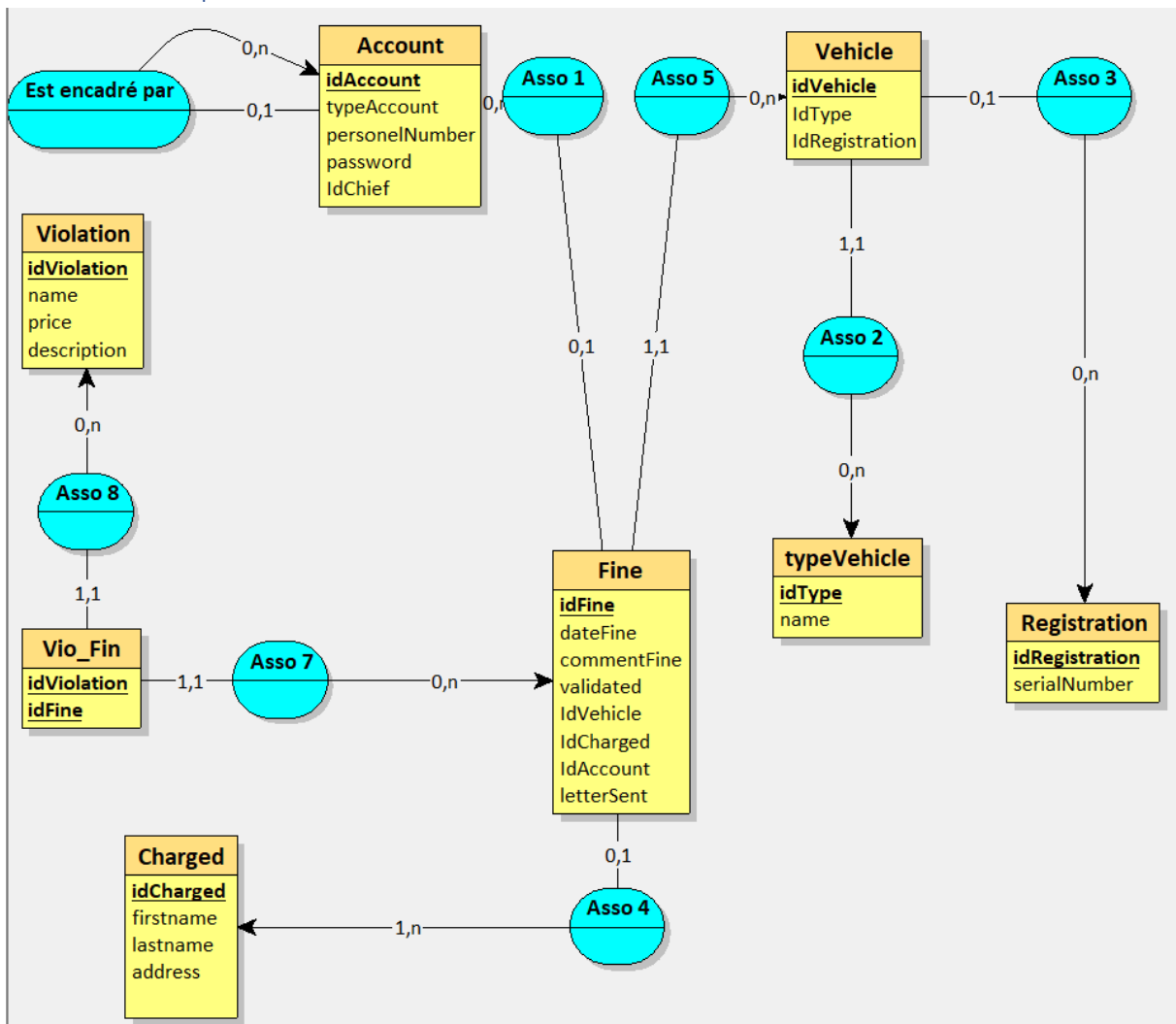
Lien Git

- <https://github.com/Kira-Atha/JEE22-2>
 - Appez : Alexandre
 - Kira-Atha : Gabriel

Base de données

L'utilisateur utilisé est : STUDENT03_13.

Schéma conceptuel



PL SQL

DDL

- SGBD\Usely\ Script-Police-Create.sql

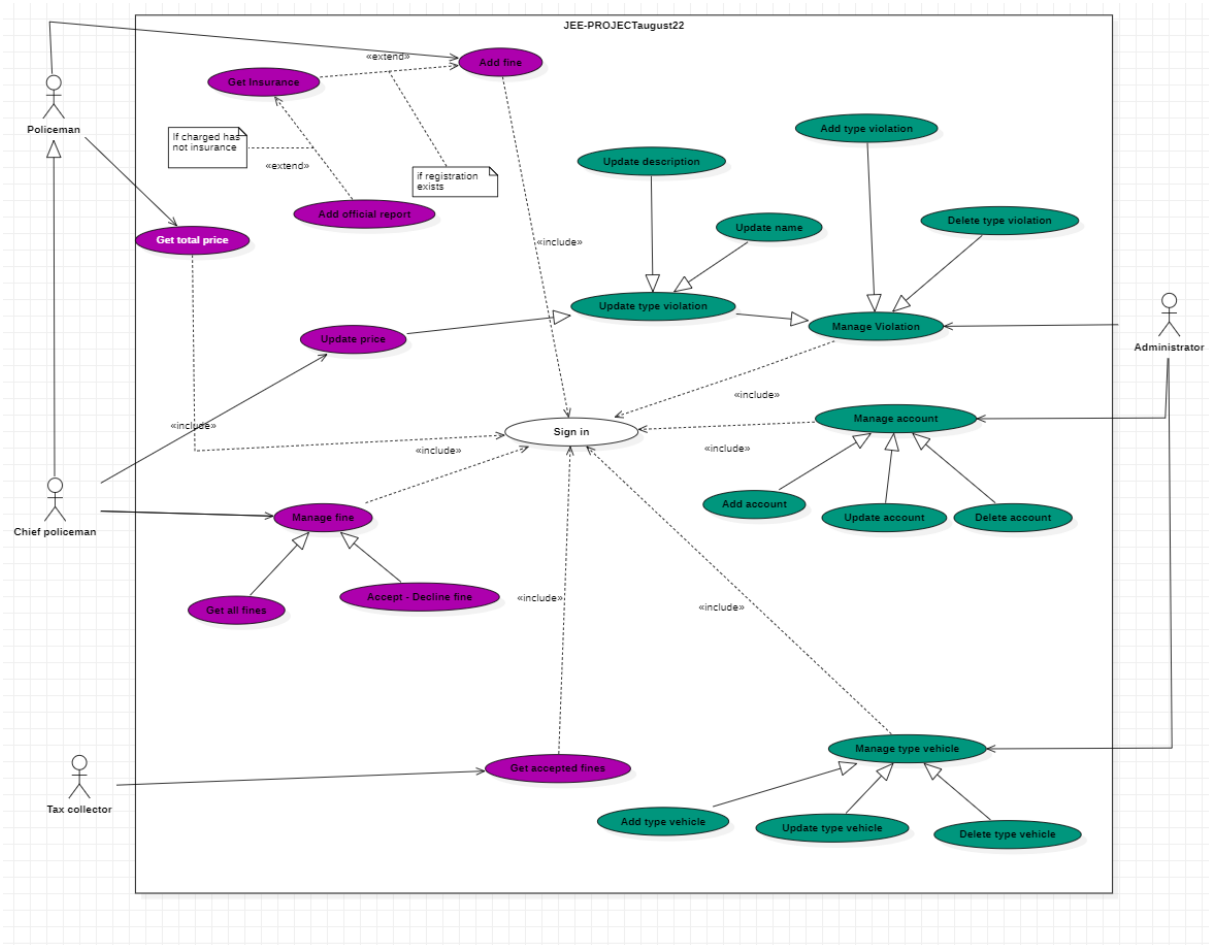
Packages

- SGBD\proc

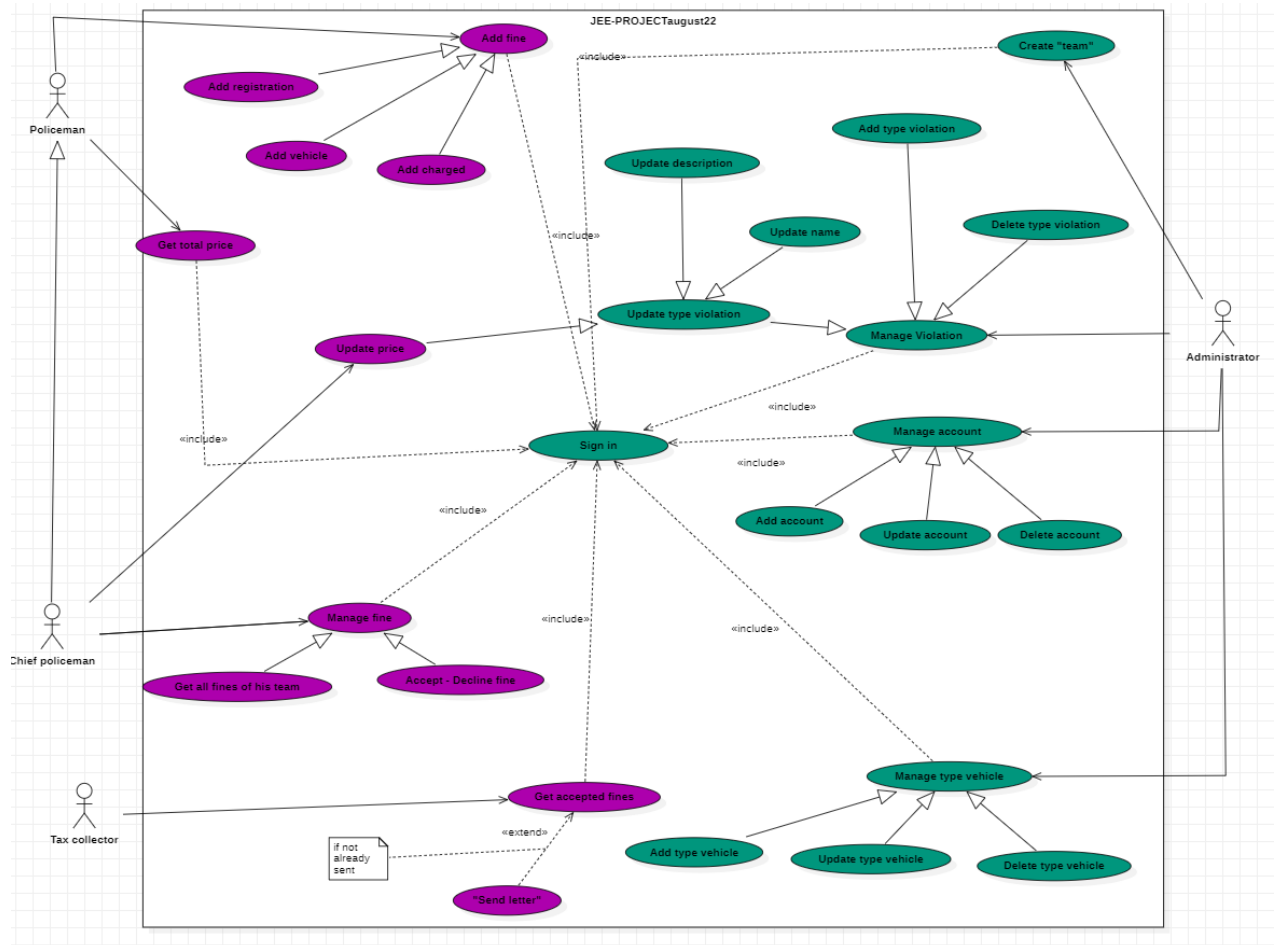
Analyse

Use case diagram

Jet 1



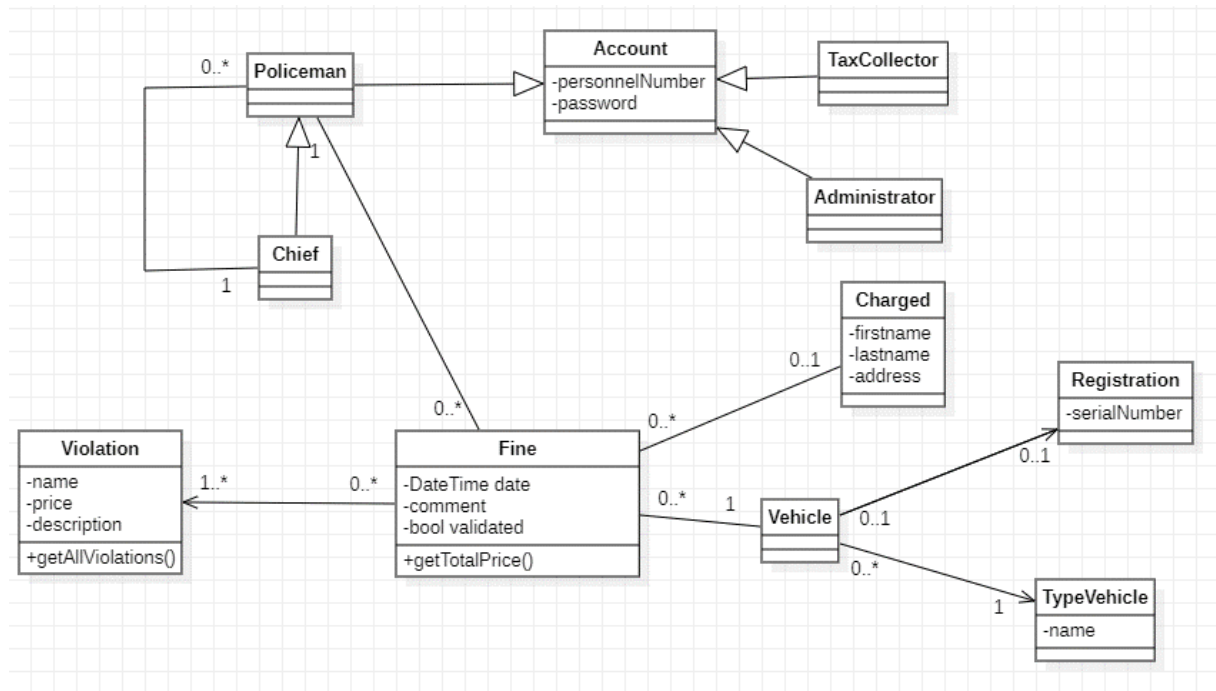
Jet 2



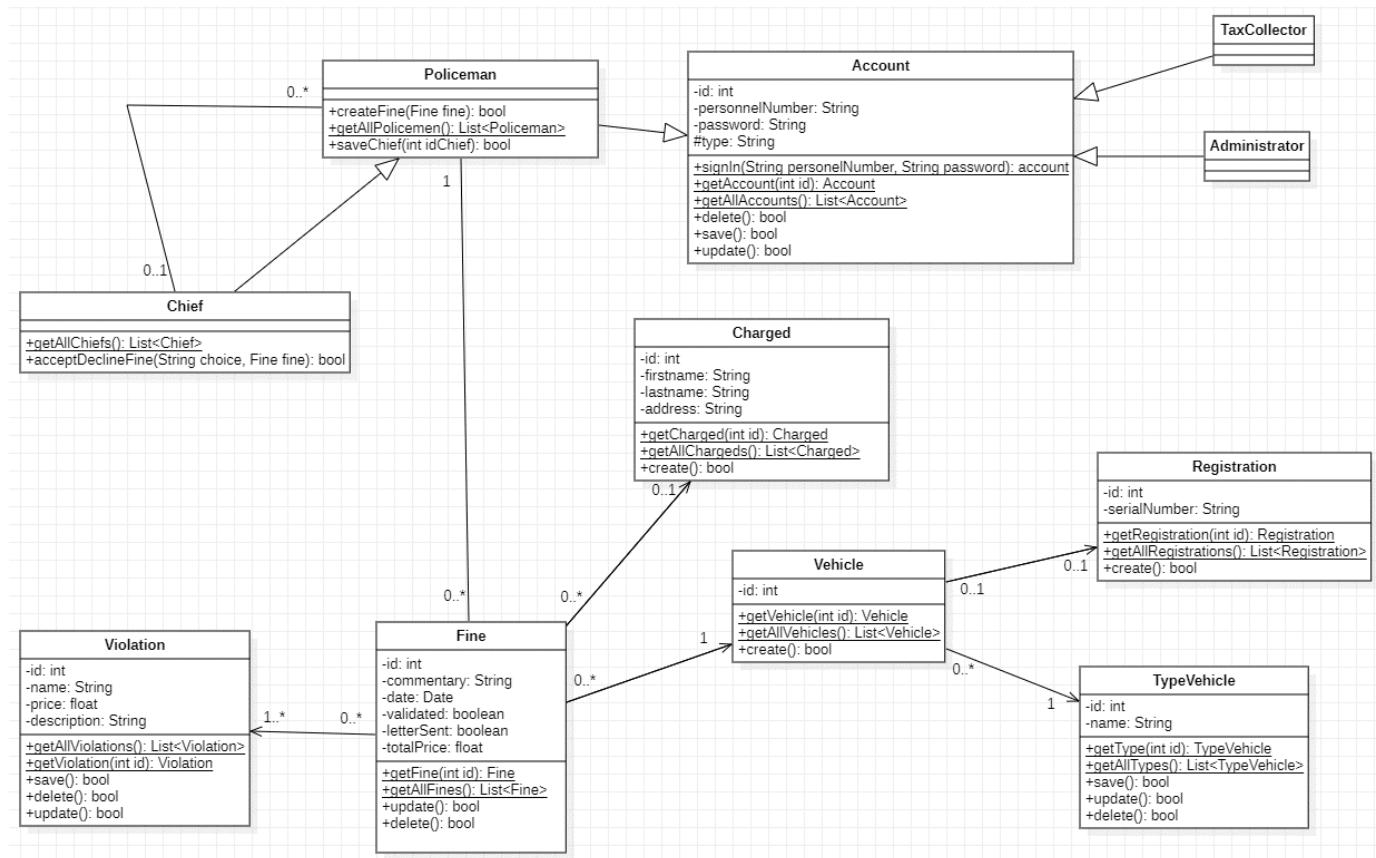
Ce jet correspond plus aux limites du programme. Les couleurs correspondent à la répartition du travail. Les bleu-verts ont été gérées par Alexandre et les violettes par Gabriel.

Class diagram

Jet 1



Jet 2



Conclusion

Réaliser cette application nous a permis de mettre en application la matière qui a été vue en classe. Nous pensons avoir mieux compris le cours.