



Site et application mobile GILLA

Documentation technique

Sommaire

Cahier des Charges	2
Organisation du projet GILLA.....	6
Spécifications fonctionnelles	12
Spécifications techniques.....	27
Conception de la solution	29
Codage du composant com_gilla	34
Application AndroidGilla	47
Intégration et tests d'intégration du site GILLA	50
Tests et recette fonctionnelle	53
Déploiement	56

Cahier des Charges

Présentation du Lycée Louis Armand

Le lycée polyvalent Louis Armand situé à Paris 15^e offre aux élèves de seconde générale un parcours diversifié :

- Bac général : Bac Spé Maths/PC/NSI/SES/HG/SVT.
- Bac technologique : Bac STI2D et Bac STMG.
- Bac professionnel : Bac Pro MELEC / Bac Pro CIEL /Bac Pro Photographie.
- CAP électricien.
- UPEAA.

Après le BAC, le lycée offre les formations supérieures suivantes :

- BTS CIEL (anc. Systèmes Numériques).
- BTS Gestion de la PME.
- BTS Management Commercial Opérationnel.
- BTS Services Informatiques aux Organisations.
- BTS Assistant Technique d'Ingénieur
- BTS Photographie en alternance.
- BTS Assistant Technique d'Ingénieur en alternance.

Objectif du projet GILLA

L'objectif du projet GILLA est de mettre en place un site permettant aux élèves, étudiants, professeurs et personnels du lycée de signaler des problèmes de tout type (éclairage, fermeture porte ou fenêtre, matériel informatique ou logiciel, etc.) et prendre en charge leur résolution.

Le site GILLA doit permettre une gestion centralisée des « tickets d'incidents » et une affectation automatique aux personnels de maintenance selon le type d'incident identifié, pour une prise en charge rapide.

Analyse de l'existant

La gestion des incidents au lycée se fait actuellement de façon désordonnée par email ou oralement à divers personnels responsables au lycée. Les incidents ne sont pas gérés de façon centralisée pour pouvoir les traiter efficacement en fonction des ressources de maintenance disponibles, selon leurs expertises.

Une application gérée par la région permet de signaler des problèmes informatiques qui sont traités toutes les deux ou trois semaines par un intervenant technique de la région. Elle est en pratique très peu utilisée au lycée.

L'organisation du personnel du lycée est donnée en page suivante.

Organigramme des services

Service direction :	Proviseur : Mme G.Malleville Proviseur Adjoint : Mme C. Ali-Cherif	
Service intendance : <i>Cantine / Bourses</i>	Intendant : L. Cherfaoui Secrétariat d'intendance : R. Vairac Secrétariat d'intendance : W.Bancoul	
Service DDFPT : <i>Stages / PFMP / Ateliers / Partenariats professionnels</i>	DDFPT : G. Idres Assistante DDFPT : M-P Lebreton	
Service vie scolaire : <i>AED / Suivi scolaire</i>	CPE : H. Horatius CPE : S. Fratta CPE : C. Zannier <i>Quelle CPE pour quelle classe ici</i>	
Pôle médico social : <i>Santé / Orientation</i>	Infirmière : N. Bourreau Médecin scolaire : C. Visseriat Assistante sociale : P. Jeay PsyEn : Mme O.Descout	
Pôle maintenance et entretien	Responsable : M. Gesbert	

L'équipe de maintenance comprend les quatre agents suivants :

- M. Gualbert GESBERT : Responsable
- M. El Hassane BELLAOUI : Agent de maintenance
- M. Tarek BRIDIA : Agent de maintenance
- M. Manuel DA-CUNHA : Agent de maintenance

Ils prennent en charge l'entretien des éléments des trois types suivants :

- Bâtiment : portes, fenêtres, murs et sols.
- Mobilier : tables, chaises et armoires.
- Informatique : vidéoprojecteurs, ordinateurs, imprimantes.

Chaque élément est situé dans un emplacement de type salle, local, couloir ou escalier.

Le plan du lycée permet d'identifier les salles par étage sous la forme XYnn, X étant la lettre du bâtiment (A, B, C, D ou F), Y le numéro de l'étage et nn étant les deux chiffres du numéro de salle (ex : A313 : bâtiment A, étage 3, numéro 13).

D'un point de vue technique, l'infrastructure technique du lycée dispose :

- D'un réseau Wifi avec des bornes dans chaque salle et comportant plusieurs réseaux privés : LARM élèves, LARM Professeur, etc.
- D'un réseau local filaire (LAN) Ethernet commuté, accessible depuis toutes les salles par prises RJ45 et configuré en plusieurs VLAN.
- D'un serveur proxy académique filtrant les accès Internet et accessible par le réseau Wifi et le réseau local.
- Un serveur de domaine « LouisArmand » comprenant un annuaire LDAP pour gérer les autorisations de connexion de tous les utilisateurs du lycée et un serveur de fichier.
- Des serveurs réservés à certains groupes d'utilisateurs (Administration, BTS-SIO...), accessibles à certains VLAN.

Analyse des besoins

A l'issue d'une série d'entretiens réalisés auprès du directeur technique du lycée, M. IDRES, les besoins généraux de conception du système de gestion des incidents au lycée Louis Armand se résument comme suit :

- Faciliter le signalement d'incidents au lycée, quelque-soient leurs types.
- Rendre accessible le signalement d'incidents à toute personne travaillant au lycée.
- Centraliser l'enregistrement des incidents et leurs affectations aux personnels de maintenance.
- Accélérer la prise en charge des incidents et les opérations de maintenance correspondantes.
- In fine, favoriser une amélioration continue de la qualité des infrastructures du lycée.

Les besoins spécifiques du site GILLA sont notamment les suivants :

- Permettre à toute **personne** travaillant au lycée (élève, étudiant, enseignant, AED, CPE, personnel administratif, agent technique) de signaler un incident et de préciser le type d'élément concerné (par sélection dans une liste) et son emplacement avec un commentaire et une photo si besoin ; lui permettre aussi de suivre la prise en charge des incidents qu'il a signalé.
- Permettre à un **agent** de maintenance de consulter les incidents du ou des type(s) dont il a la charge, de gérer leurs priorités (Haute, Moyenne ou Basse) et leurs états d'avancement (Affecté, Résolu ou Fermé) et de les clôturer avec un commentaire.
- Permettre au **responsable** de la maintenance de gérer l'équipe de maintenance et les affectations des agents aux types d'incidents ; lui permettre aussi de disposer d'un tableau de bord pour suivre la prise en charge des incidents par type et par personnel de maintenance.
- Permettre à un **administrateur** du site de gérer les profils des utilisateurs inscrits et leurs rôles associés (groupe d'utilisateur) ; lui permettre aussi de mettre à jour le site et la base de données associée.

Une analyse détaillée de ces besoins à partir d'interview des principaux utilisateurs concernés devra permettre d'établir un document de spécifications fonctionnelles à faire valider par les parties prenantes du projet en charge de sa coordination, avant de lancer les phases d'exécution technique du projet.

D'un point de vue technique, le site GILLA devra être accessible depuis tout navigateur Internet et être « *responsive* », c'est-à-dire adapté à tout type d'écran : PC/Mac, tablette ou téléphone

(smartphone).

Le prestataire (MOE) devra concevoir le site à partir d'un framework PHP (Joomla 4 ou Symfony 5) pour sécuriser la solution et faciliter son développement et sa maintenance.

Une application mobile (Android dans un premier temps) sera aussi proposée aux utilisateurs pour signaler des incidents et suivre leurs prises en charge depuis leur téléphone. Celle-ci devra communiquer via une API sécurisée à la base de données du site.

Tout au long de son déroulement, ce projet de développement logiciel devra être soigneusement documenté dans ses aspects fonctionnels et techniques afin de faciliter la communication entre la MOA et la MOE.

Organisation du projet GILLA

Organisation générale du projet

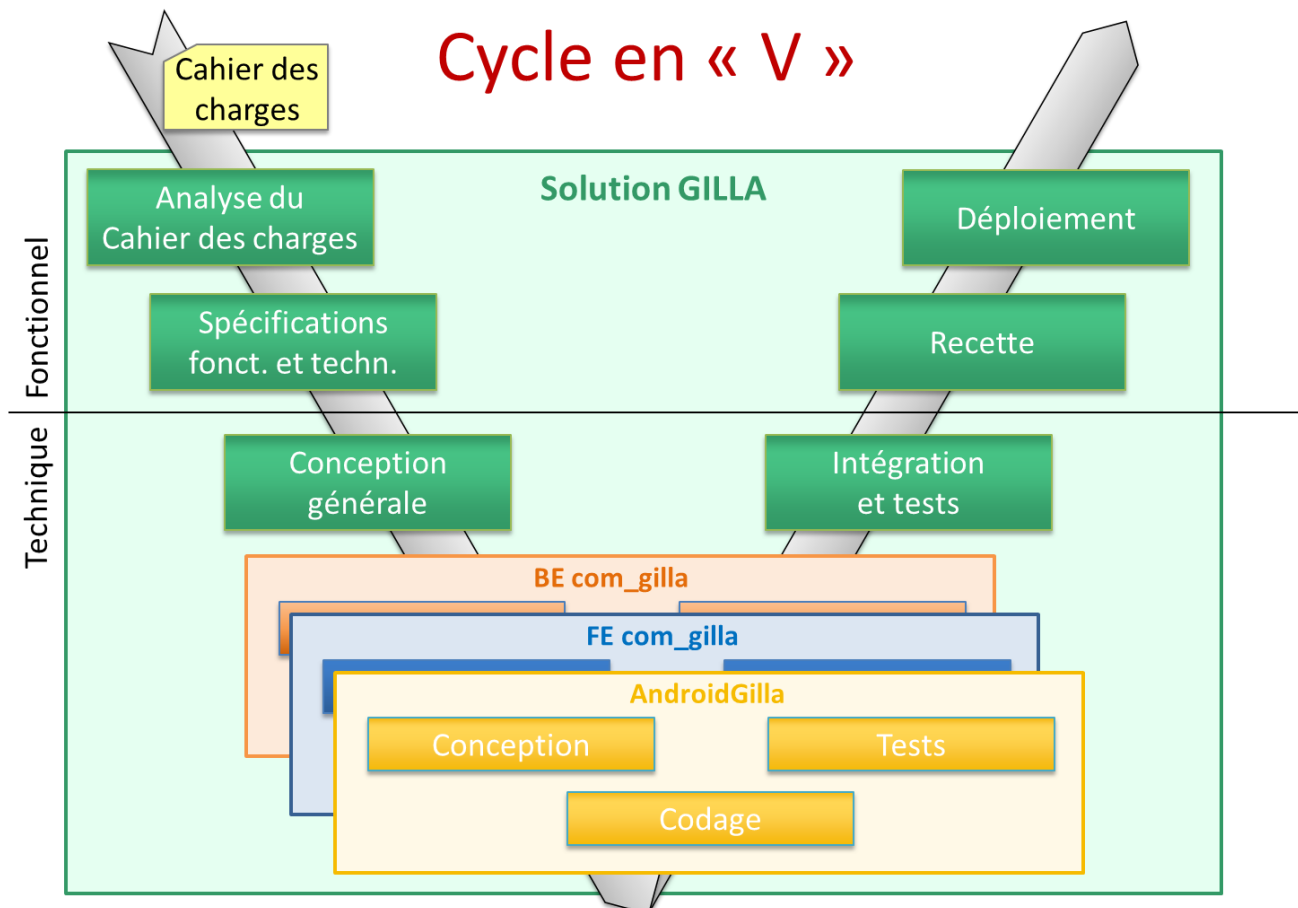
Le développement du projet GILLA a été confié aux étudiants de deuxième année de BTS SIO option SLAM du lycée. L'organisation du projet mise en place par le professeur en charge du bloc 2 du référentiel du BTS SIO, a coïncidé avec la progression pédagogique établie pour ce bloc, le projet GILLA servant de fil conducteur au déroulé des cours et des travaux pratiques.

Le projet GILLA a d'abord été décomposé en trois livrables principaux liés à l'architecture du framework Joomla retenu et présenté plus loin (cf. chapitre conception et architecture) :

- Le **site d'administration**, avec le développement de la partie backend du composant com_gilla (extension Joomla spécialement développée pour le projet).
- Le **site public** incluant l'accès réservé aux utilisateurs enregistrés avec la configuration du CMS Joomla et le développement de la partie frontend du composant com_gilla.
- Le développement de l'**application mobile Android** et de l'API associée.

Approche méthodologique

Le développement s'est fait selon la méthodologie du cycle de vie des logiciels (cycle en « V ») définissant les grandes étapes du projet depuis l'analyse du cahier des charges jusqu'à la recette fonctionnelle et au déploiement :



Le cycle en « V » de la solution GILLA commence à partir du cahier des charges, avec les trois étapes suivantes :

- Analyse du cahier des charges.
- Spécifications fonctionnelles et techniques.
- Conception générale.

A partir de cette étape, l'architecture générale de la solution est définie et trois développements sont identifiés :

1. Le backend du composant com_gilla.
2. Le frontend du composant com_gilla.
3. L'application mobile AndroidGilla.

Ces trois développements se décomposent en trois sous-cycles en « V » :

- Conception
- Codage
- Test

Une fois ces trois développements terminés, le cycle en « V » principal continue, avec les trois étapes suivantes :

- Intégration et tests.
- Recette.
- Déploiement.

Le déploiement aboutit dans ce projet école à la mise en production d'une solution Web et mobile prototype, sur des serveurs de l'hébergeur OVH.

Planning général

Le planning général du projet GILLA et de ses trois sous-projets (BE : Backend, FE : Frontend, AM : Application Mobile) s'est inscrit dans le calendrier (progression pédagogique) mis en place sur l'année scolaire 2023-2024 comme suit :

Semaine de cours			Thème	Séquence
N°	Lu	Sa		
0	28/8	2/9		
1	4/9	9/9	Analyser la demande	1 - Analyse du cahier des charges de la solution
2	11/9	16/9		
3	18/9	23/9	Spécifier la solution	2 - Spécifications technique et fonctionnelle de la solution
4	25/9	30/9		
5	2/10	7/10	Concevoir la solution	3 - Architecture de la solution
6	9/10	14/10		
7	16/10	21/10	Concevoir le BE	4a - Architecture du BE du composant
Vac. Toussaint				
8	6/11	11/11	Coder le BE	5a - Codage du BE du composant
9	13/11	18/11	Tester le BE	6a - Tests du BE du composant
10	20/11	25/11	Concevoir le FE	4b - Architecture du FE du composant
11	27/11	2/12	Coder le FE	5b - Codage du FE du composant
12	4/12	9/12		
13	11/12	16/12	Tester le FE	6b - Tests du FE du composant
14	18/12	23/12		Stage en entreprise (6 semaines)
Vac. Noël				
15	8/1	13/1		
16	15/1	20/1		
17	22/1	27/1		
18	29/1	3/2		
19	5/2	10/2		
Vac. Hiver				
20	26/2	2/3	Concevoir l'AM	4c - Architecture de l'application mobile
21	4/3	9/3	Coder l'AM	5c - Codage de l'application mobile
22	11/3	16/3		
23	18/3	23/3	Tester l'AM	6c - Tests de l'application mobile
24	25/3	30/3	Documenter	8 - Recette et documentation de la solution
25	1/4	6/4		CCF E5
Vac. Printemps				

Gestion du projet

Des groupes de travail ont été constitués pour faciliter le partage de connaissance et la collaboration entre les étudiants mis en situation professionnelle lors des travaux pratiques. Des contributions individuelles au projet ont été aussi demandées à chaque étudiant pour permettre à chacun de développer son autonomie.

Les tableaux suivants identifient, les **responsables des développements par acteur et par rubrique**, pour les groupes A et B.

Groupe A :

Backend de com_gilla :

Acteur	Rubrique	Vue liste		Vue détail	
Administrateur	Affectations	RDP	Aymen	CRU	Gustave
	Agents	RDP	Matéo	CRU	Hugo
	Emplacements	RDP	Davy	CRU	Radu
	Etats	RDP	Ilan	CRU	
	Incidents	RDP	Ryan	CRU	Mathis
	Priorités	RDP		CRU	
	Prises en charge	RDP	Malik	CRU	Abdel-Karim
	Types	RDP	Bayrone	CRU	Oumar

Frontend de com_gilla :

Acteur	Rubrique	Vue liste		Vue détail	
Utilisateur	Incidents ouverts	R	Hugo	R	Aymen
	Signaler un incident			CRU	Davy
	Mes incidents	R	Gustave	R	Malik
Agent	Incidents à gérer	R	Radu	R	Matéo
	Prises en charge	R	Mathis	CRU	Ilan
Responsable	Agents	R	Oumar	CRU	Ryan
	Affectations	R	Abdel-Karim	CRU	Bayrone

AndroidGilla :

Acteur	Rubrique	Vue liste		Vue détail	
Utilisateur	Incidents ouverts	R	(tous)	R	(tous)
	Signaler un incident			CRU	(tous)
	Mes incidents	R		R	
Agent	Incidents à gérer	R		R	
	Prises en charge	R		CRU	

Groupe B :

Backend de com_gilla :

Acteur	Rubrique	Vue liste		Vue détail	
Administrateur	Affectations	RDP	Agahlya	CRU	Aïssata
	Agents	RDP	Abdenour	CRU	Alexy
	Emplacements	RDP	Tyron	CRU	Yanis
	Etats	RDP	Gabriel F.	CRU	Gabriel F.
	Incidents	RDP	Gabriel P.	CRU	Omar
	Priorités	RDP		CRU	
	Prises en charge	RDP	Melvyn	CRU	Julian
	Types	RDP	Clément	CRU	Jean-François

Frontend de com_gilla :

Acteur	Rubrique	Vue liste		Vue détail	
Utilisateur	Incidents ouverts	R	Omar	R	Jean-François
	Signaler un incident			CRU	Melvyn
	Mes incidents	R	Aïssata	R	Agahlya
Agent	Incidents à gérer	R	Yanis	R	Clément
	Prises en charge	R	Gabriel F.	CRU	Abdenour
Responsable	Agents	R	Alexy	CRU	Gabriel P.
	Affectations	R	Julian	CRU	Tyron

AndroidGilla :

Acteur	Rubrique	Vue liste		Vue détail	
Utilisateur	Incidents ouverts	R	(tous)	R	(tous)
	Signaler un incident			CRU	(tous)
	Mes incidents	R		R	
Agent	Incidents à gérer	R		R	
	Prises en charge	R		CRU	

Gestion des versions logicielles

La gestion des versions logicielles est faite avec le logiciel libre **Git** qui permet :

- D'enregistrer des **états de la base de code** appelés *commits* (versions) associés à des commentaires, permettant de suivre plus facilement l'évolution du projet et de revenir en arrière si besoin.
- D'organiser les versions dans des **branches**, permettant de développer une version sans impacter les autres.
- De **travailler à plusieurs** sur le même projet, grâce à un système de merge (fusion).
- De disposer de **sauvegardes** du projet grâce à une logique de distribution. Car chaque ordinateur dispose du projet dans son intégralité.

Le système d'identification des utilisateurs de Git fonctionne avec une simple adresse email. Chaque utilisateur peut créer des *commits* à chaque fonctionnalité développée.

Pour simplifier le travail à plusieurs, un dépôt privé accessible en ligne est également créé sur le site **github.com**.

Chaque utilisateur peut ainsi envoyer ses *commits* sur github et fusionner son code avec celui des autres.

Une seule branche a été créée, la branche **master**, afin de simplifier l'intégration du code (*merge*) au fil de l'eau. Le nombre de conflits de *merge* a été limité par le fait que chaque développeur travaille sur des fichiers différents, à quelques exceptions près.

Spécifications fonctionnelles

Le site GILLA se décompose en deux parties :

- Le **backend GILLA**, site d'administration du site GILLA.
- Le **frontend GILLA**, site GILLA constitué d'une partie publique et d'une partie à accès réservé aux utilisateurs enregistrés.

Les présentes spécifications fonctionnelles définissent les acteurs, les diagrammes de cas d'utilisation et les diagrammes de séquence associés, les interfaces utilisateurs et le modèle des données commun aux deux sites.

Acteurs

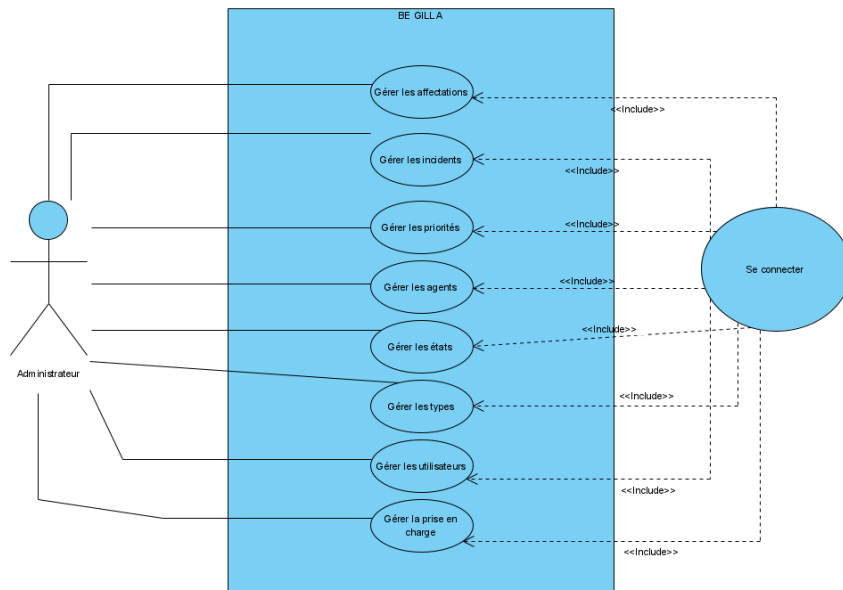
Au sens UML, les acteurs concernés par le système GILLA sont les suivants :

- Backend GILLA: administrateur.
- Frontend GILLA: utilisateur, agent et responsable.

Diagramme de cas d'utilisation du backend GILLA

Le diagramme des cas d'utilisation UML du backend GILLA est le suivant :

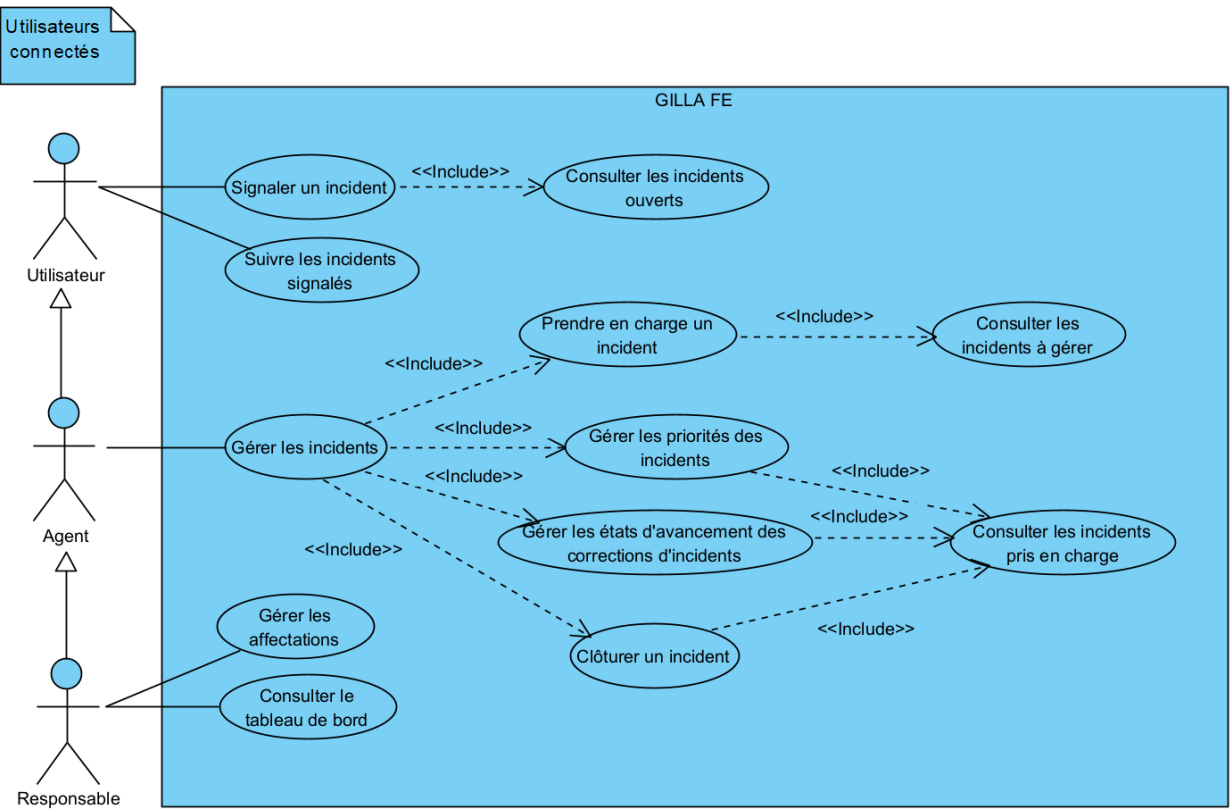
(#01) Abdenour



« Gérer... » inclut les 5 cas d'utilisation : Ajouter, Lire, Modifier, Supprimer et Publier/Dépublier.

Diagramme de cas d'utilisation du frontend GILLA

Le diagramme des cas d'utilisation du frontend GILLA est le suivant :



Utilisateur / Signaler un incident :*(#02) Tyron***Description**

L'utilisateur est connecté et sélectionne la rubrique « Signaler un incident » dans le menu principal du site GILLA pour voir s'afficher le formulaire de signalement d'un incident. Il remplit le formulaire de signalement d'incident (type d'incident, emplacement, description de l'incident) et le valide.

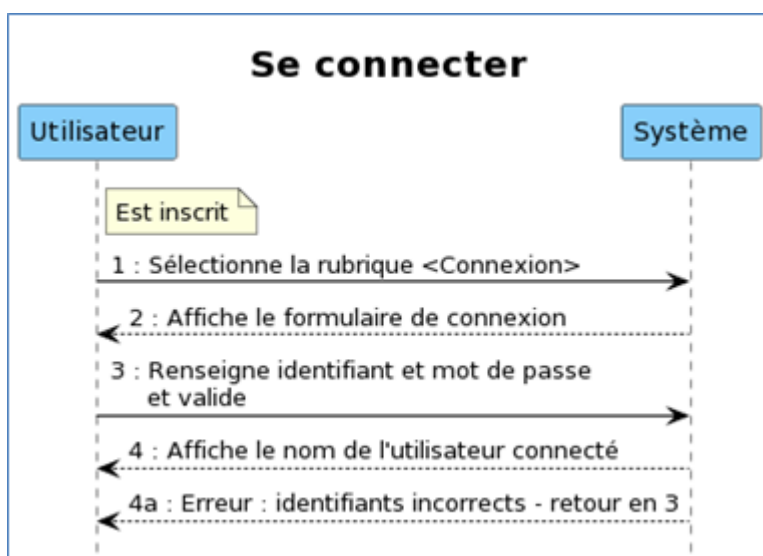
Son signalement s'affiche en cas de requête réussie. Sinon, un message d'erreur lui indique « erreur de signalement » et lui propose un nouvel essai.

S'il a oublié ou perdu son mot de passe, l'utilisateur clique sur le lien « Mot de passe perdu » (voir cas d'utilisation « Réinitialiser son mot de passe »).

S'il n'est pas encore inscrit, il clique sur le lien « Créer un compte » (voir cas d'utilisation « Créer un compte »).

Maquette IHM du formulaire de signalement d'incident

Type	Emplacement	Description de l'incident	Photos
Matériels	Porte/table (bâtiment/étage/salle)	Porte défectueuse (badge défectueux)	Présentez photo
Informatique	Pc/projecteur/réseau (bâtiment/étage/salle)	Pc défectueux (ne veut pas s'allumer)	Présentez photo
Bâtiment	Mur/Toiture (bâtiment/étage/salle)	Mur dégrader (cassure/fissure)	Présentez photo

Diagramme de séquence UML**Utilisateur / Consulter les incidents ouverts :**

(#03) Agahlya

Description

L'utilisateur est déjà connecté. Il sélectionne la rubrique « Consulter les Incidents ouverts », le système lui affiche la page de la liste de tous les incidents, avec leurs noms, les date d'ouverture de l'incident, le lieu.

Maquette IHM

Type	Nom	Etat	Date d'ouverture	Lieu	Salle	Photo
Informatique	Clavier non fonctionnel	En cours	29-09-2023	Bâtiment E	E001	
Bâtiment	Porte bloqué	En cours	01-09-2023	Bâtiment A	A318	
Informatique	Ecran cassé	En cours	15-09-2023	Bâtiment F	F003	

Diagramme de séquence UML

Consulter les incidents ouverts

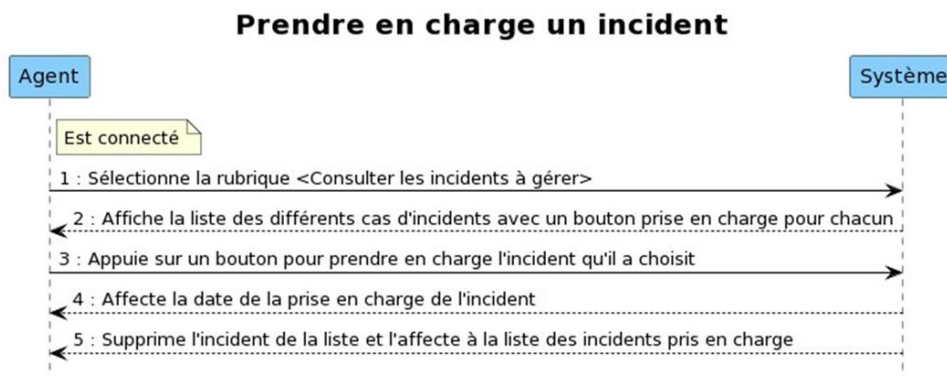


Agent / Prendre en charge un incident :*(#04) Omar***Agent : Prendre en charge un incident****Description**

L'agent est connecté et sélectionne la rubrique « Consulter les incidents à gérer » du site GILLA pour que la liste des incidents à gérer s'affiche. Il choisit un incident à gérer clique sur le bouton pour le prendre en charge et la date est affecté à l'incident ainsi que le nom de l'agent. Suppression de l'incident pris en charge et affecter à la liste des incidents pris en charge

Maquette IHM

GILLA		
Incident (date+ salle)	Type d'incident	Bouton Prendre en charge
14_09_2023_A312	Souris cassé	Oui
26_09_2023_B012	Porte qui ne s'ouvre plus	Oui
11_09_2023_F001	TNI	Oui
04_10_2023_A102	Chaise bancale	Oui

Diagramme de séquence UML**Agent / Consulter les incidents pris en charge :**

(#05) Clément

Agent: Consulter les incidents pris en charge

Description

L’Agent est déjà connecté et sélectionne la rubrique “incident à gérer” dans le menu principal du site GILLA pour pouvoir accéder aux incidents. Cette rubrique affiche les incidents sous forme de tableau verticale où sont afficher en ligne les différents incidents. On peut voir dans cette liste l’incident, s’il est géré ou non, une description de l’incident c’est-à-dire l’endroit où il a été signalé, par qui, l’heure du système et une photo (optionnelle). Cette consultation va permettre à l’agent de prendre ou charge ou non les différents incidents.

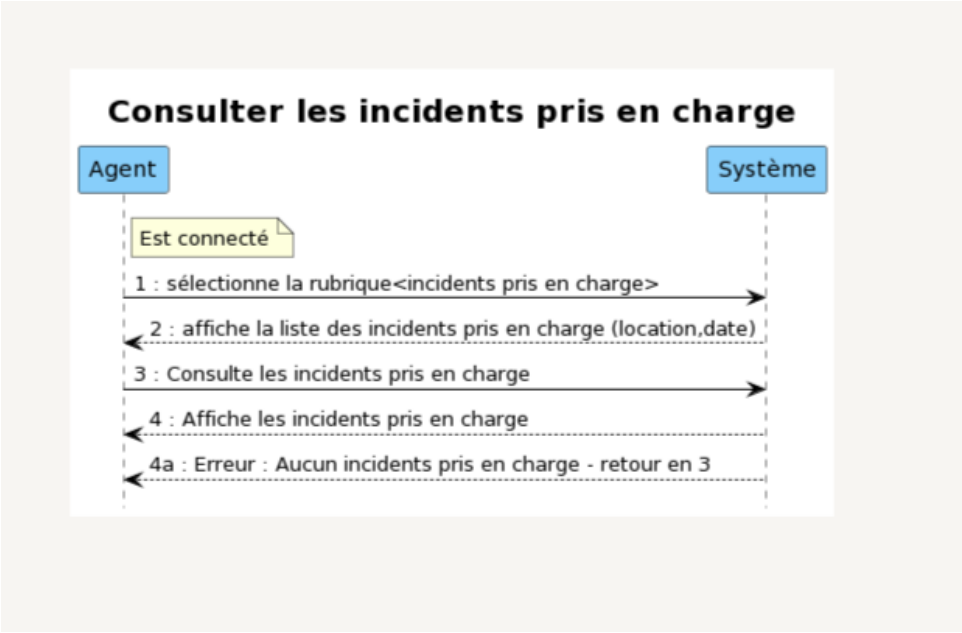
Maquette IHM

GILLA				
NOM DES INCIDENTS	MATÉRIEL ENDOMMAGÉ	LIEU (ÉTAGE, SALLE)	PERSONNE AYANT SIGNALÉ ET HEURE DU SYSTEME	INCIDENT VALIDÉ ?
CL.A138.2.V	Clavier	A318	Etudiant	OUI
EC.A210.5.V	Écran	A210	CPE	OUI

VU DE DÉTAILS DE L'INCIDENT SÉLECTIONNÉ

INCIDENT PRIS EN CHARGE : CL.A138.2.V
MATÉRIEL ENDOMMAGÉ : Clavier
LIEU (BATIMENT, ÉTAGE, SALLE) : A318
PERSONNE AYANT SIGNALÉ ET HEURE DU SYSTEME : Etudiant
DESCRIPTION DETAILLE OU PHOTO(S) : Clavier inutilisable car les touches ZQSD et le paver numérique est manquant.
INCIDENT VALIDÉ ? Oui

Diagramme de séquence UML

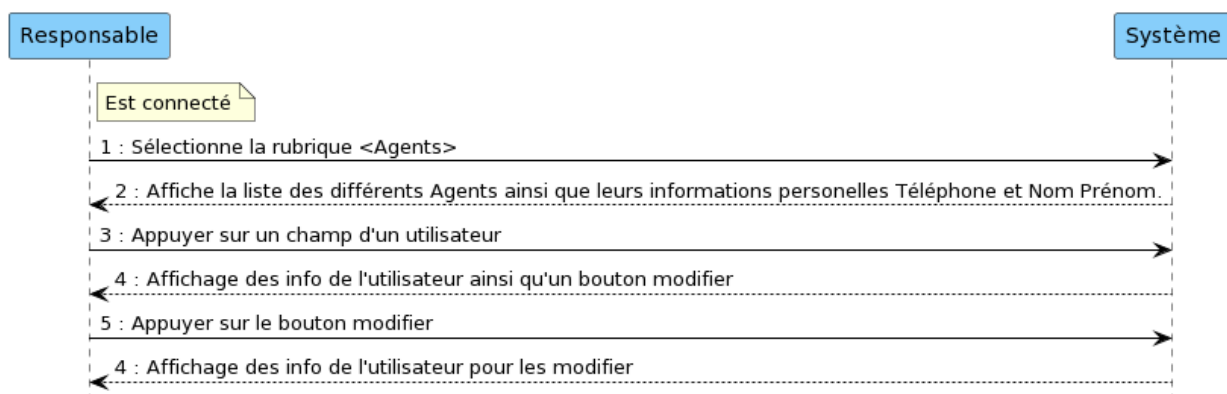


Responsable / Gérer les agents :*(#06)Gabriel F.***Description**

Le responsable est connecté, il sélectionne la rubrique « Agent» dans le menu principal du site GILLA pour voir s’afficher la liste des différents agent, avec leurs informations et la possibilité de modifier leurs informations.

Maquette IHM du formulaire

GILLA	
Telephone :	Agent :
Numéro	Nom Prénom
07262521456	Henry BOGART
0786475896	Marc DUPLAN

Diagramme de séquence UML**Gérer les agents**

Responsable / Gérer les affectations :

(#07) Melvyn

Description

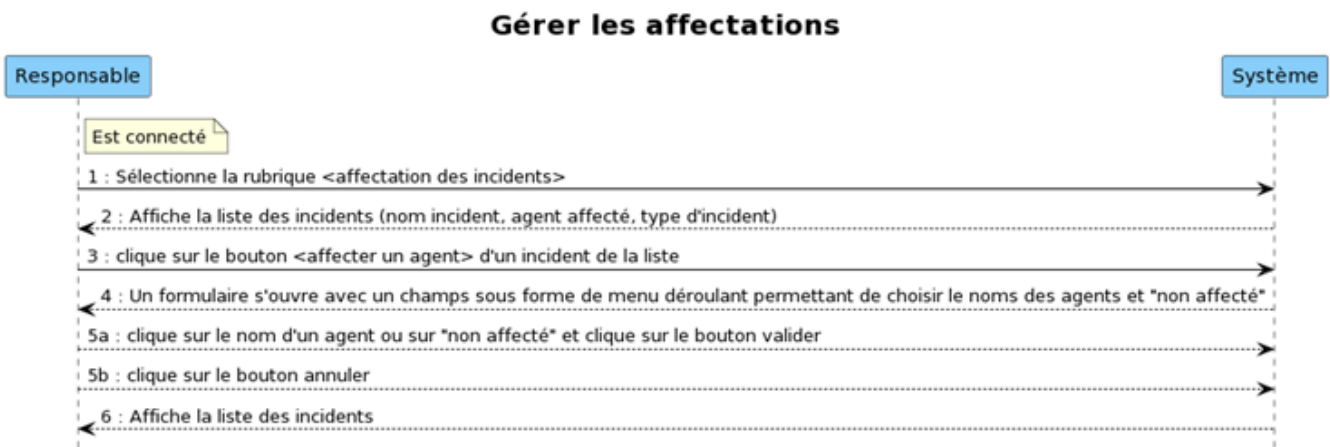
Le responsable déjà connecté sélectionne la rubrique affectation des tickets pour voir s’afficher la liste des incidents. Dans la liste chaque incident est affiché avec le nom de l’incident, la personne affectée, le type d’incident et un bouton affecter un agent. Si personne n’est affecté alors il est marqué "non affecté".

Lorsque le responsable clique sur un incident de la liste, un formulaire s’ouvre permettant de choisir grâce à un menu déroulant l’agent à affecter ou "non affecté". Puis de cliquer sur le bouton valider ou annuler.

Maquette IHM du formulaire

GILLA	
Affecter un agent	
Non affecté	
Valider	Annuler

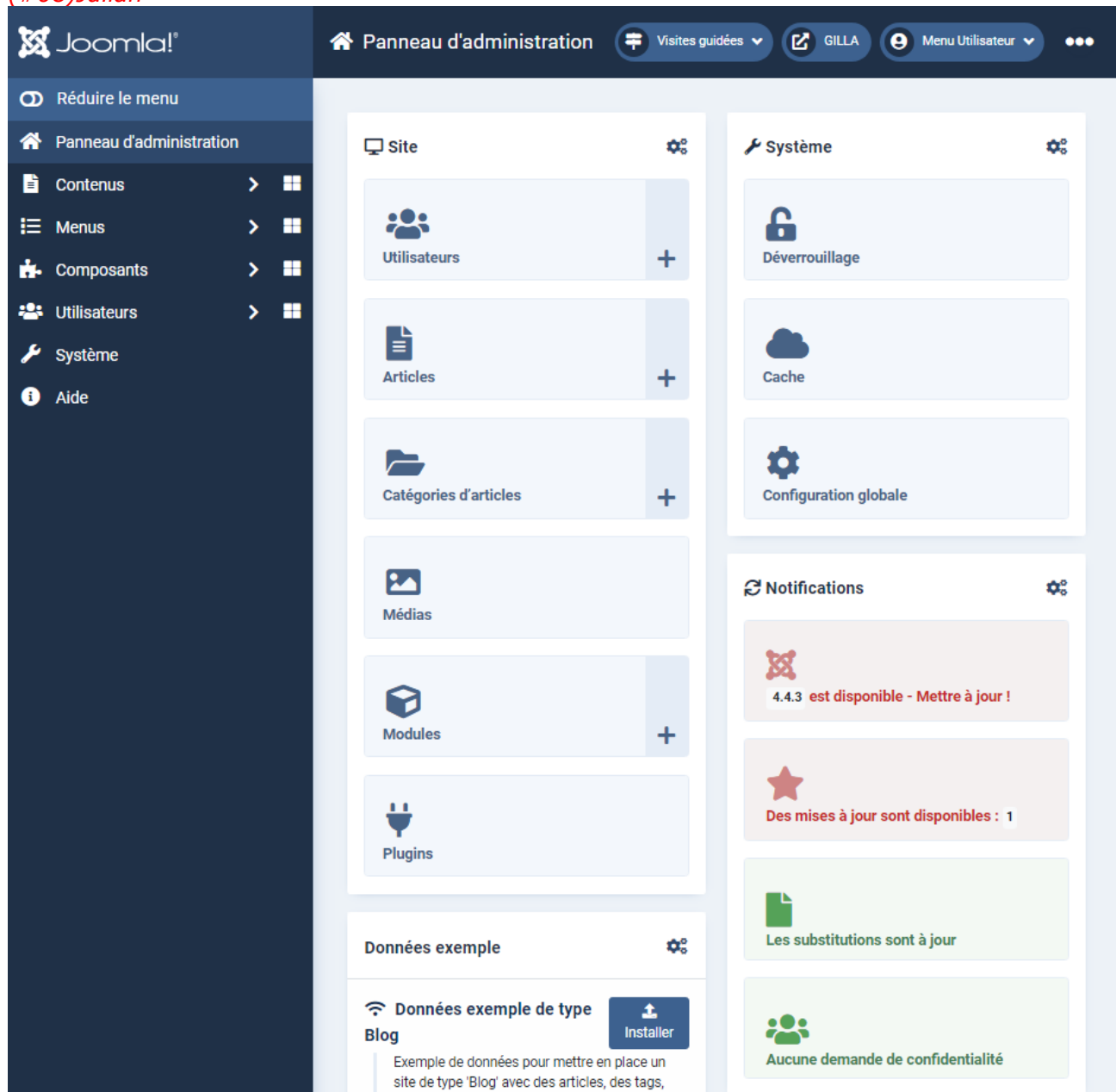
Diagramme de séquence UML



Interfaces utilisateurs du backend GILLA

Voici la page d'accueil du backend de Joomla 4 :

(#08)Julian



Après sélection du composant GILLA et de la rubrique « Prises en charge », on souhaite obtenir l'interface suivante :

(#08) Gabriel

The screenshot displays the GILLA 'Prises en charge' (Incident Management) interface. The header bar includes the title 'GILLA : Prises en charge' and navigation links for 'Visites guidées', 'GILLA', and 'Menu Utilisateur'. Below the header, there are buttons for '+ Nouveau', 'Actions', 'Paramètres', and 'Aide'. A search bar with the placeholder 'Recherche...' and a magnifying glass icon is present, along with a 'Filtres d'affichage' dropdown and an 'Effacer' button. A sorting section shows 'Trier les listes par :' with a dropdown arrow and a '20' items per page selector. The main content area features a table with the following data:

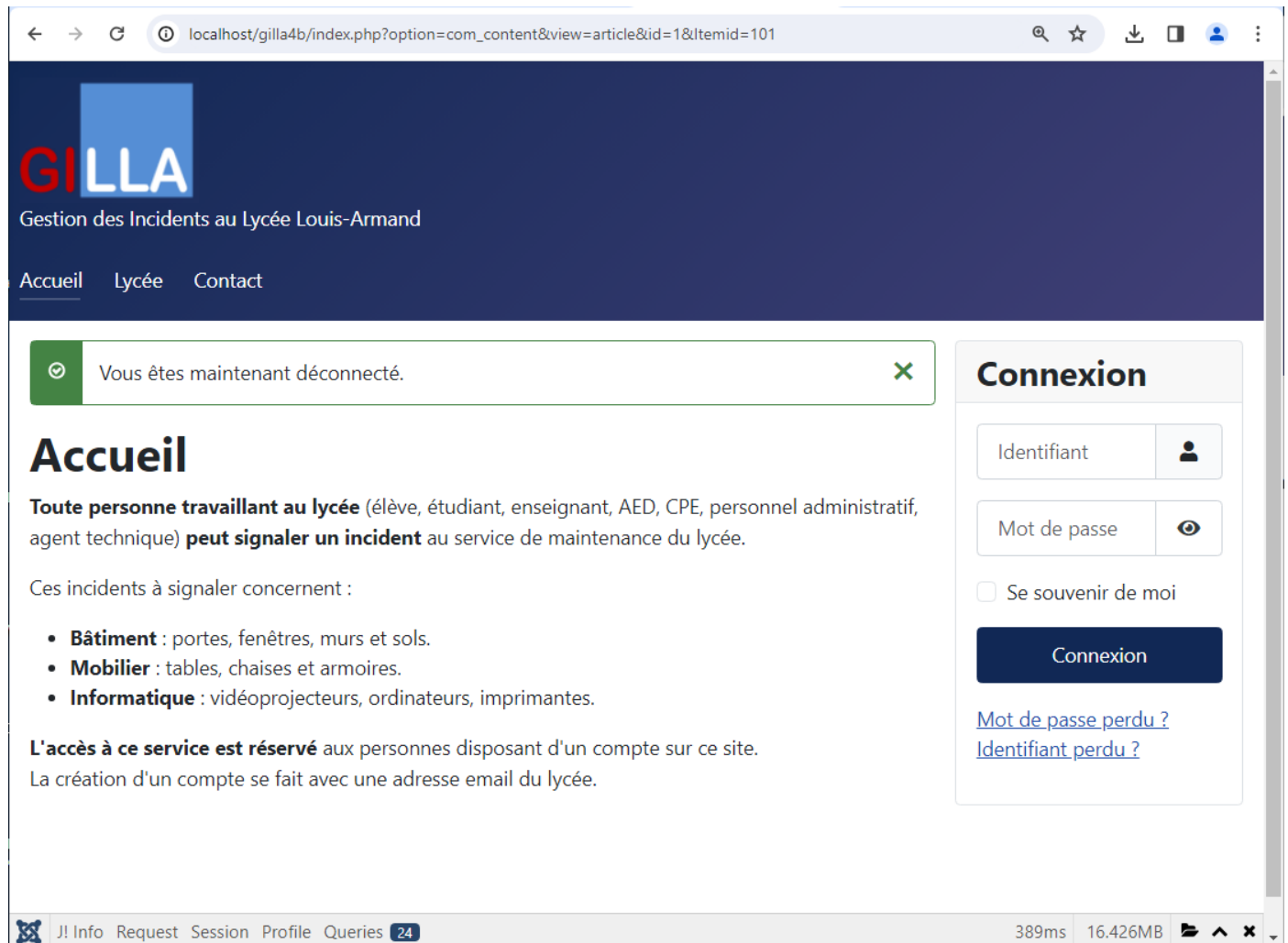
<input type="checkbox"/>	Statut	Incident	Agent	Date et heure de début	Priorité	Etat	Id
<input type="checkbox"/>	✓	20211123-0001	Théo BELMONDO	23-11-2021 08:20:00	2 - Moyenne	2 - Pris en charge	1
<input type="checkbox"/>	✓	20211123-0002	Marc DUPLAN	23-11-2021 08:36:00	2 - Moyenne	2 - Pris en charge	2
<input type="checkbox"/>	✓	20211123-0003	Henry BOGART	23-11-2021 08:39:00	1 - Haute	2 - Pris en charge	3

The footer bar shows system information: 'JI Info Request Session Profile Queries 35', '1.12s', '18.293MB', and window control icons.

Interfaces utilisateurs du frontend GILLA

La page d'accueil du frontend GILLA devrait se présenter comme suit :

(#09) Omar



Après connexion d'un agent, le menu « Gestion des incidents » et la page correspondante à la rubrique « Signaler un incident » devraient apparaître comme suit :

(#09) Tyron

The screenshot displays the GILLA web application interface. The browser's address bar shows the URL: `localhost/gilla4b/index.php?option=com_gilla&view=incident_e&layout=edit&id=0&Itemid=120`. The page header features the GILLA logo and the text "Gestion des Incidents au Lycée Louis-Armand". Below the header is a navigation menu with links: Accueil, Lycée, Contact, Mon profil, and Evènements.

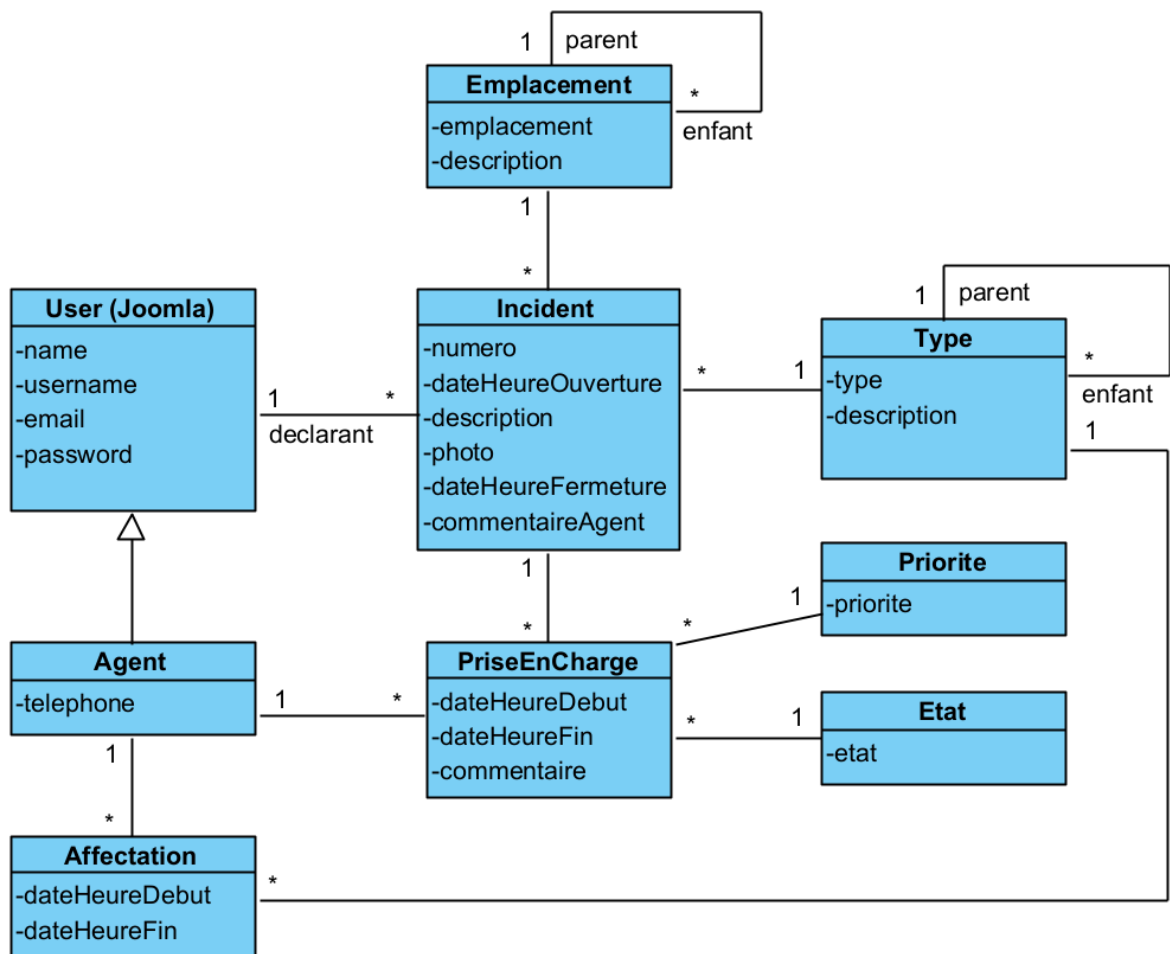
The main content area is divided into two sections. On the left, the "Incident" form is visible, containing the following fields:

- Numéro *
- Date et heure d'ouverture *
- Emplacement *
- Type *
- Description *
- Date et heure de fermeture

On the right, the "Gestion des incidents" sidebar is displayed, showing the user's name "Utilisateur : Marc DUPLAN" and a "Déconnexion" button. Below this, the "Connexion" section is visible, showing the user's name "Bonjour, Marc DUPLAN" and a "Déconnexion" button.

Modèle des données GILLA

Le modèle des données GILLA est décrit par le diagramme de classes UML suivant :



Les valeurs prédéfinies des attributs sont les suivantes :

type	emplacement	priorite	etat
Bâtiment	Bât A	1 - Haute	1 - Ouvert
- Aération	- Et A0	2 - Moyenne	2 - Pris en charge
- Eclairage	- - A001	3 - Basse	3 - Fermé
- Chauffage	- - ...		
- Fenêtre	- - A009		
- Mur	- Et A1		
- Plafond	- - A101		
- Porte	- - ...		
- Prise de courant	- - A117		
- Prise réseau	- Et A2		
- Sol	- - A201		
Informatique	- - ...		
- Clavier	- - A223		
- Ecran	- Et A3		
- PC portable	- - A301		
- Souris	- - ...		
- Unité centrale	- - A318		
- Vidéoprojecteur	- Et A4		
Mobilier	Bât B		
- Armoire	Bât C		
- Chaise	Bât D		
- Table	Bât E		
	Bât F		

Spécifications techniques

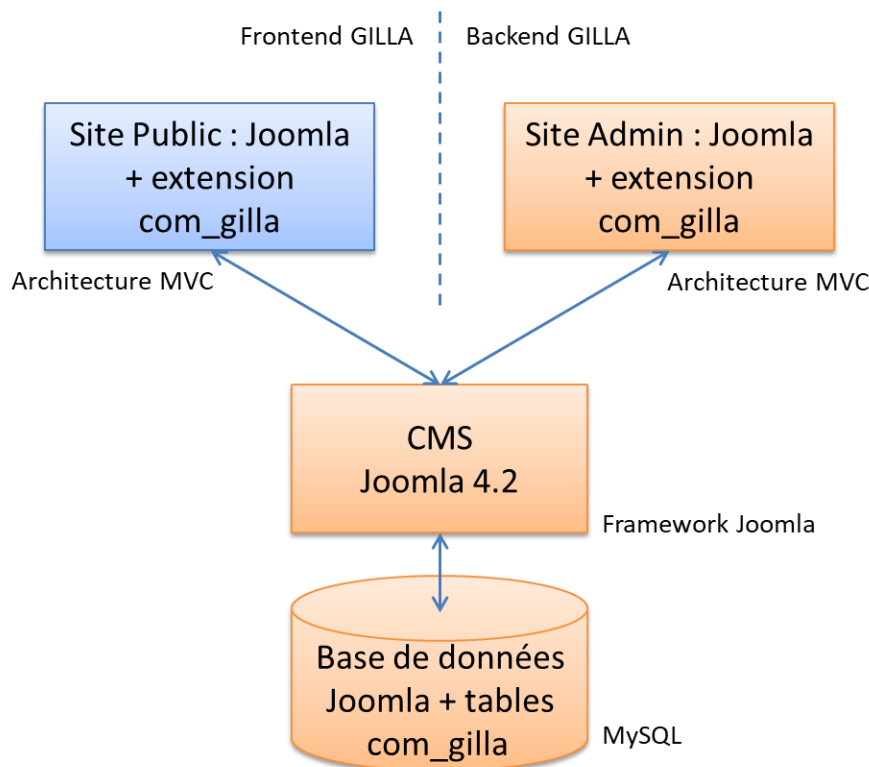
Framework et CMS Joomla

Afin de réduire les risques liés à la maîtrise d'œuvre d'un tel projet, l'équipe projet GILLA a retenu le principe d'une architecture logicielle ouverte (architecture MVC : Modèle-Vue-Contrôleur) avec Framework/CMS Joomla 4.4 et un développement PHP en deux phases :

1. Développement du backend GILLA et de la base de données « GILLA » sous la forme d'un composant « com_gilla » installable dans le CMS Joomla 4.4.
2. Développement du frontend GILLA avec le composant « com_gilla » pour implémenter les fonctionnalités attendues des utilisateurs.

La technologie Joomla a été choisie parmi les trois CMS (Content Management System : outil de gestion de contenu sur Internet) open-source et gratuits les plus populaires du marché : WordPress, Drupal et Joomla. Sa simplicité d'utilisation, la qualité de ses extensions et le dynamisme de ses communautés d'utilisateurs et de développeurs en France, en Europe et dans le monde assurent à Joomla un suivi et une évolution de très grande qualité.

Le schéma d'architecture logicielle est le suivant :



Les éléments coloriés en orange correspondent aux développements relatifs à la première phase et comprennent les deux éléments du socle commun, la base de données Joomla sous MySQL et l'installation de l'API Joomla 4.2, permettant ensuite l'intégration du composant « com_gilla » spécialement développé pour les besoins du site GILLA. L'élément colorié en bleu correspond à la deuxième phase du développement explicitée ci-dessus.

Environnements de développement et de tests

L'environnement de développement et de test qui a été mis en place est le suivant :

- Installation sur chaque PC étudiant de :
 - **XAMPP 8.1.10** (PHP 8.1.10, Apache 2.4.54, MariaDB 10.4.25, phpMyAdmin 5.2.0)
 - **Visual Studio Code 1.75.1**
 - **Joomla 4.4.2**
- Installation sur le serveur du labo SLAM :
 - Machine virtuelle Debian 8.2
 - Services Apache2, PHP et MySQL
 - Logiciel phpMyAdmin

L'environnement de production choisi a été une plateforme mutualisée de l'hébergeur OVH.com (offre Pro sur serveur LAMP) dont les détails sont donnés dans le chapitre déploiement.

Conception de la solution

Architecture logicielle MVC

Le choix d’une architecture logicielle MVC répond aux besoins d’ouverture, de maintenance et d’évolutivité du site par une organisation standardisée du code source. Le but d’une telle architecture est de structurer le code, pour chaque cas d’utilisation ou bloc fonctionnel, en trois parties :

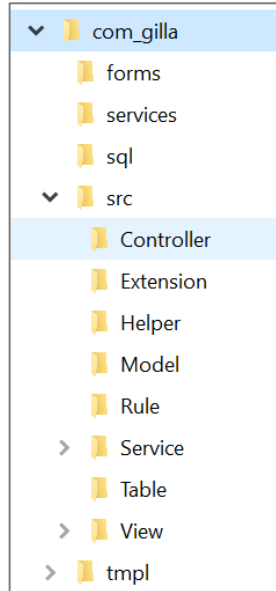
- Le **modèle** gère les données du site. Il récupère les informations dans la base de données, les organise et les assemble pour qu'elles puissent ensuite être traitées par le contrôleur et la vue. Cette partie contient donc les requêtes SQL organisées sous forme de fonctions PHP.
- La **vue** gère l'affichage. Elle organise la façon dont les données sont affichées à l'écran. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples.
- Le **contrôleur** gère la logique du code et les demandes utilisateurs. Le contrôleur récupère les données du modèle, les analyse/les traite, et renvoie les données à afficher à la vue. Le contrôleur contient exclusivement du code PHP, organisé sous forme de fonctions.

L'architecture du composant `com_gilla` respecte l'arborescence et les règles de nommage définies pour CMS Joomla 4.2. A chaque vue du backend ou du frontend correspond une structure MVC de trois classes qui héritent des classes MVC de base du CMS. Ainsi pour le backend, les classes `AdminController`, `AdminModel` et `HtmlView` suivantes seront étendues :

The image shows a Visual Studio Code editor with three PHP files open. The foreground file is AdminController.php, which is part of a Joomla! MVC application. It shows the AdminController class extending BaseController and implementing AdminControllerInterface. The class has a constructor, a __construct method, and a __call method. The background files are AdminModel.php and HtmlView.php, which follow a similar pattern for the model and view layers. The editor interface includes a sidebar with a file explorer, a search bar, and a status bar at the bottom showing the current file, line, column, and encoding.

Architecture du backend de com_gilla

L'arborescence des dossiers du backend du composant com_gilla est la suivante :
(dossier **gilla/administrator/components/com_gilla**)



Les règles de nommage des fichiers php des contrôleurs, modèles et vues sont les suivantes :

- nom au pluriel pour les listes (ex : incidents)
- nom au singulier pour les formulaires de détail (ex : incident)

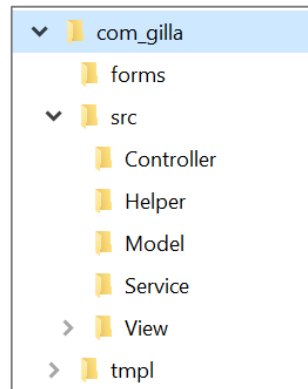
Les dossiers **Controller**, **Model**, **Table**, **View** et **tmpl** contiennent les fichiers suivants :

(#10) Abdenour

▼ tmpl	▼ Controller	▼ Model	▼ Table	▼ View
> Affection	🔗 AffectionController.php	🔗 AffectionModel.php	🔗 AffectionsTable.php	> Affection
> affectations	🔗 AffectionsController.php	🔗 AffectionsModel.php	🔗 AgentsTable.php	> Affectations
> Agent	🔗 AgentController.php	🔗 AgentModel.php	🔗 EmplacementsTable.php	> Agent
> agents	🔗 AgentsController.php	🔗 AgentsModel.php	🔗 EtatsTable.php	> Agents
> emplacement	🔗 DisplayController.php	🔗 EmplacementModel.php	🔗 Event_usersTable.php	> Emplacement
> Emplacements	🔗 EmplacementController.php	🔗 EmplacementsModel.php	🔗 EventsTable.php	> Emplacements
> etat	🔗 EmplacementsController.php	🔗 EtatModel.php	🔗 IncidentsTable.php	> Etat
> etats	🔗 EtatController.php	🔗 EtatsModel.php	🔗 Prise_en_chargesTable.php	> Etats
> event	🔗 EtatsController.php	🔗 Event_userModel.php	🔗 TypesTable.php	> Event
> event_user	🔗 Event_userController.php	🔗 Event_usersModel.php		> Event_user
> event_users	🔗 Event_usersController.php	🔗 EventModel.php		> Event_users
> events	🔗 EventsController.php	🔗 EventsModel.php		> Events
> incident	🔗 IncidentController.php	🔗 IncidentModel.php		> Incident
> incidents	🔗 IncidentsController.php	🔗 IncidentsModel.php		> Incidents
> prise_en_charge	🔗 Prise_en_chargeController.php	🔗 Prise_en_chargeModel.php		> Prise_en_charge
> prise_en_charges	🔗 Prise_en_chargesController.php	🔗 Prise_en_chargesModel.php		> Prise_en_charges
> types	🔗 TypesController.php	🔗 TypesModel.php		> Types

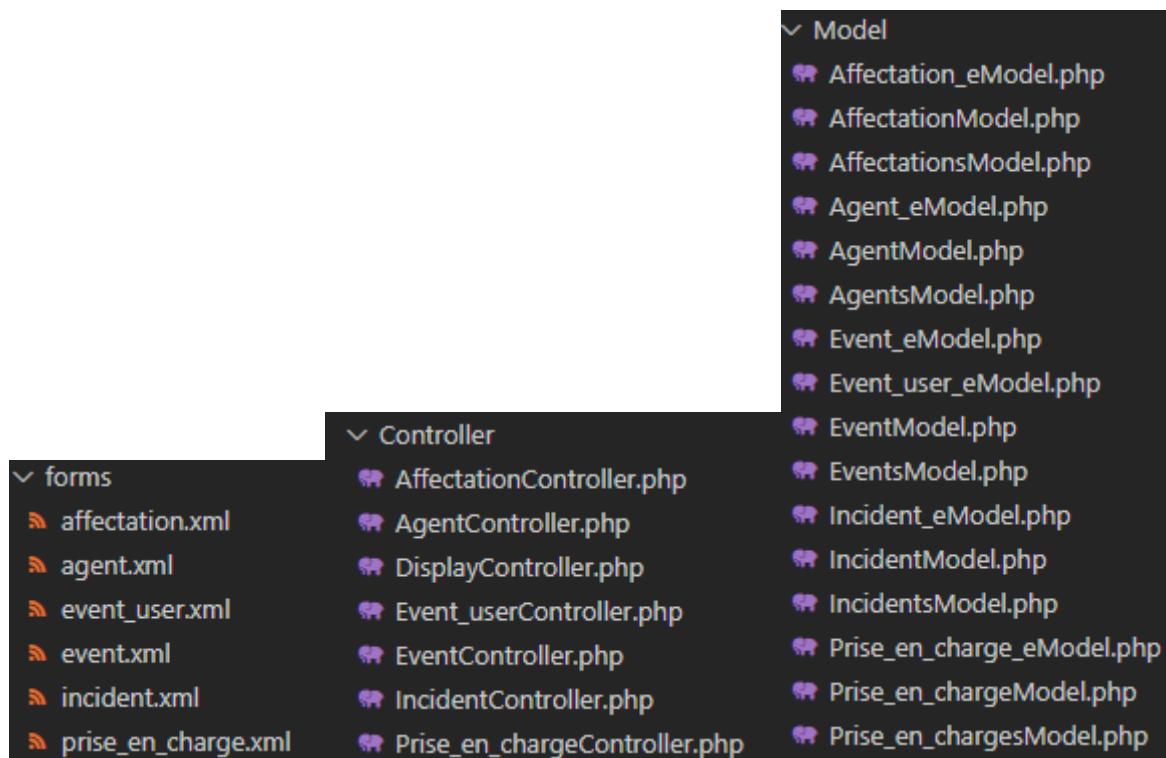
Architecture du frontend de com_gilla

L'arborescence du frontend du composant com_gilla est la suivante :
(dossier **gilla/components/com_gilla**)



Les dossiers **forms**, **Controller**, **Model**, **View** et **tmpl** contiennent les fichiers suivants :

(#11) Agahlya

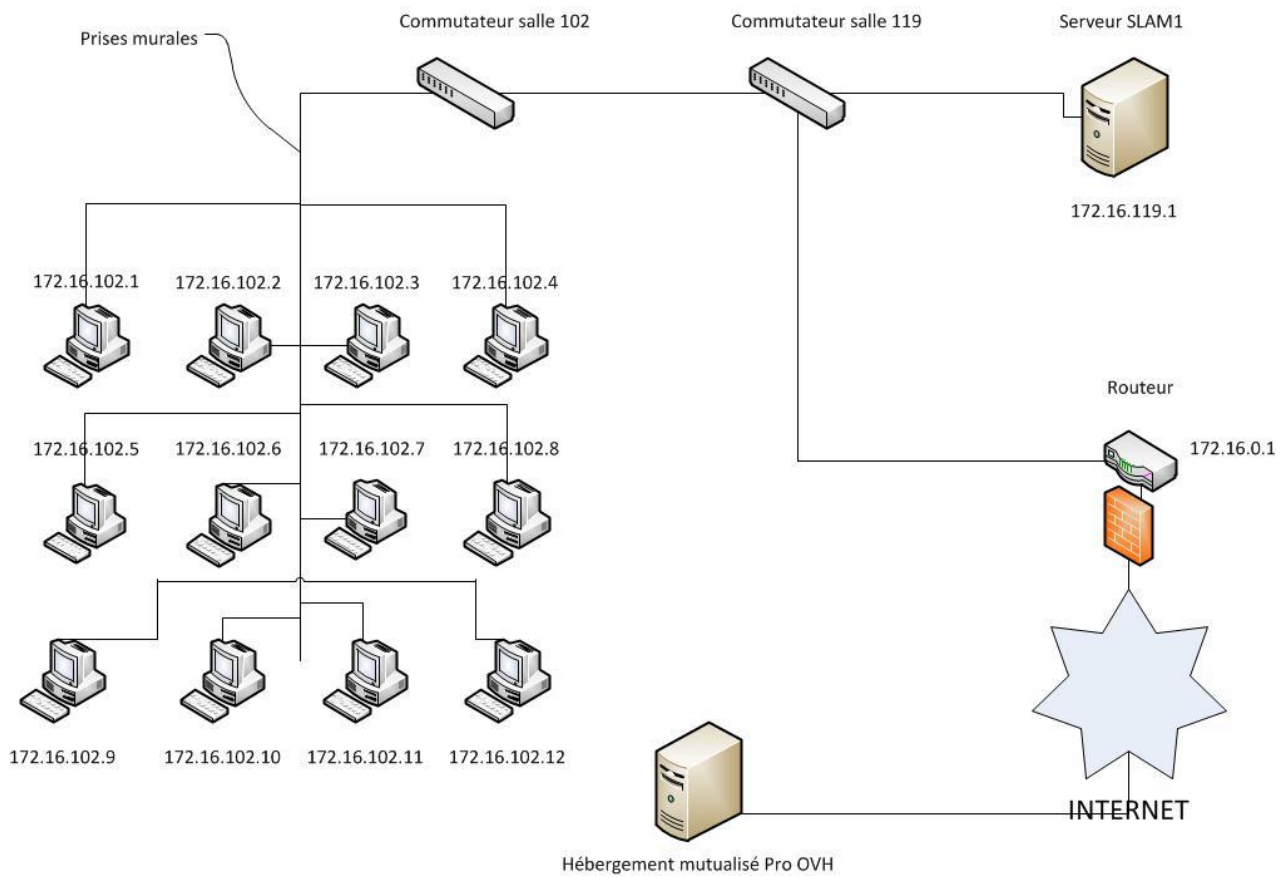


```

    ✓ tmpl
      ✓ affectation
        🐘 default.php
      ✓ Affectation_e
        🐘 edit.php
      ✓ affectations
        🐘 default_items.php
        🐘 default.php
        📄 default.xml
      > agent
      > agent_e
      > Agents
      > event
      > event_e
      > event_user_e
      > events
      > Incident
      > incident_e
      > incidents
      > prise_en_charge
      > prise_en_charge_e
      > prise_en_charges
✓ View
  ✓ Affectation
    🐘 HtmlView.php
  ✓ Affectation_e
    🐘 HtmlView.php
  ✓ Affectations
    🐘 HtmlView.php
  > Agent
  > Agent_e
  > Agents
  > Event
  > Event_e
  > Event_user_e
  > Events
  > Incident
  > Incident_e
  > Incidents
  > Prise_en_charge
  > Prise_en_charge_e
  > Prise_en_charges
```


Architecture matérielle

L'infrastructure réseau mise en œuvre pour le développement de GILLA est la suivante :



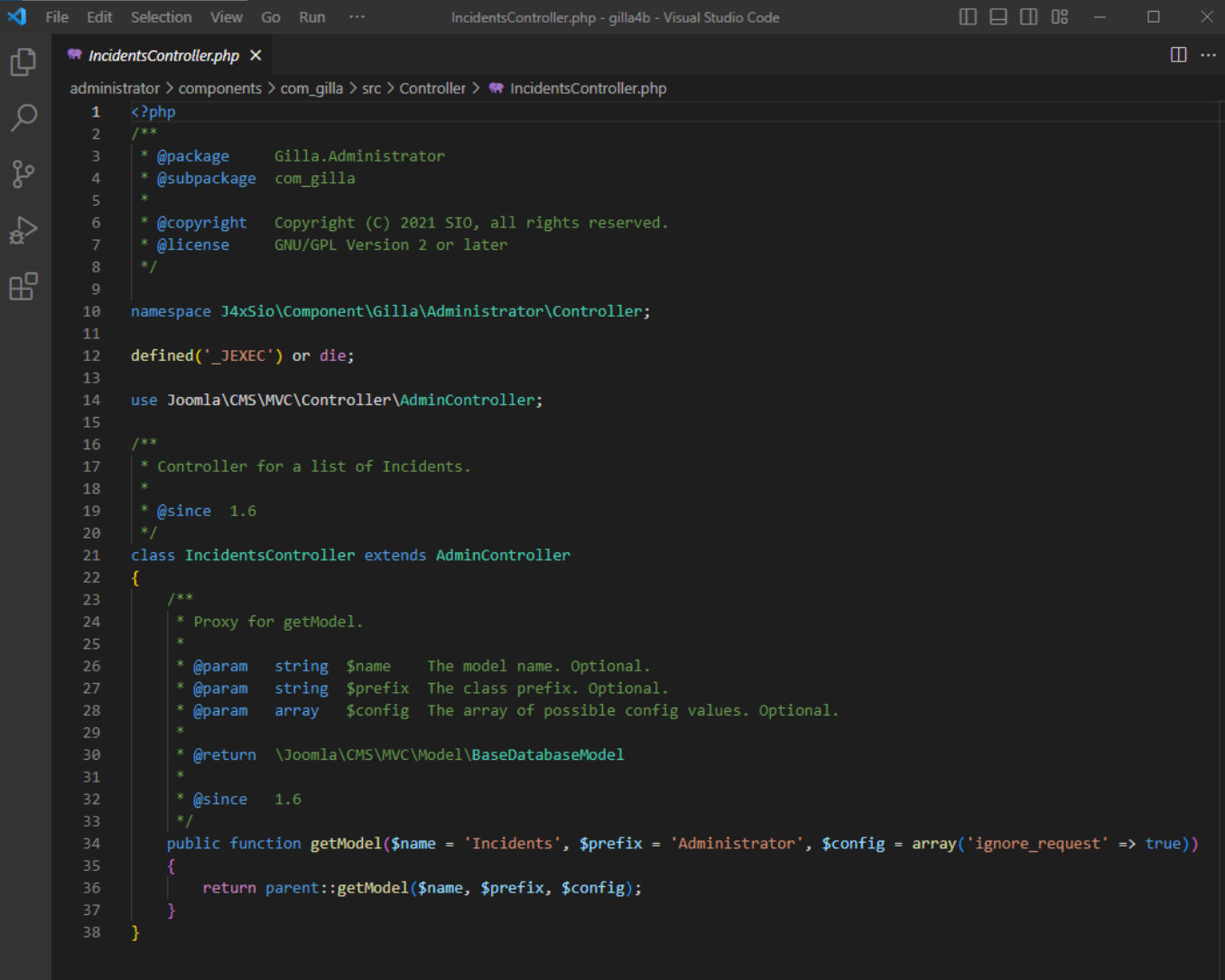
Codage du composant com_gilla

Codage MVC du backend de com_gilla

Les contrôleurs, modèles et vues constituant le code PHP du composant com_gilla selon l'arborescence et les règles de nommage des fichiers définis plus-haut sont conçus comme des classes PHP héritant des classes correspondantes du CMS Joomla.

Par exemple, la vue **liste des incidents** est constituée des fichiers de code suivants :
(#08) Julian

Contrôleur « IncidentsController.php » :

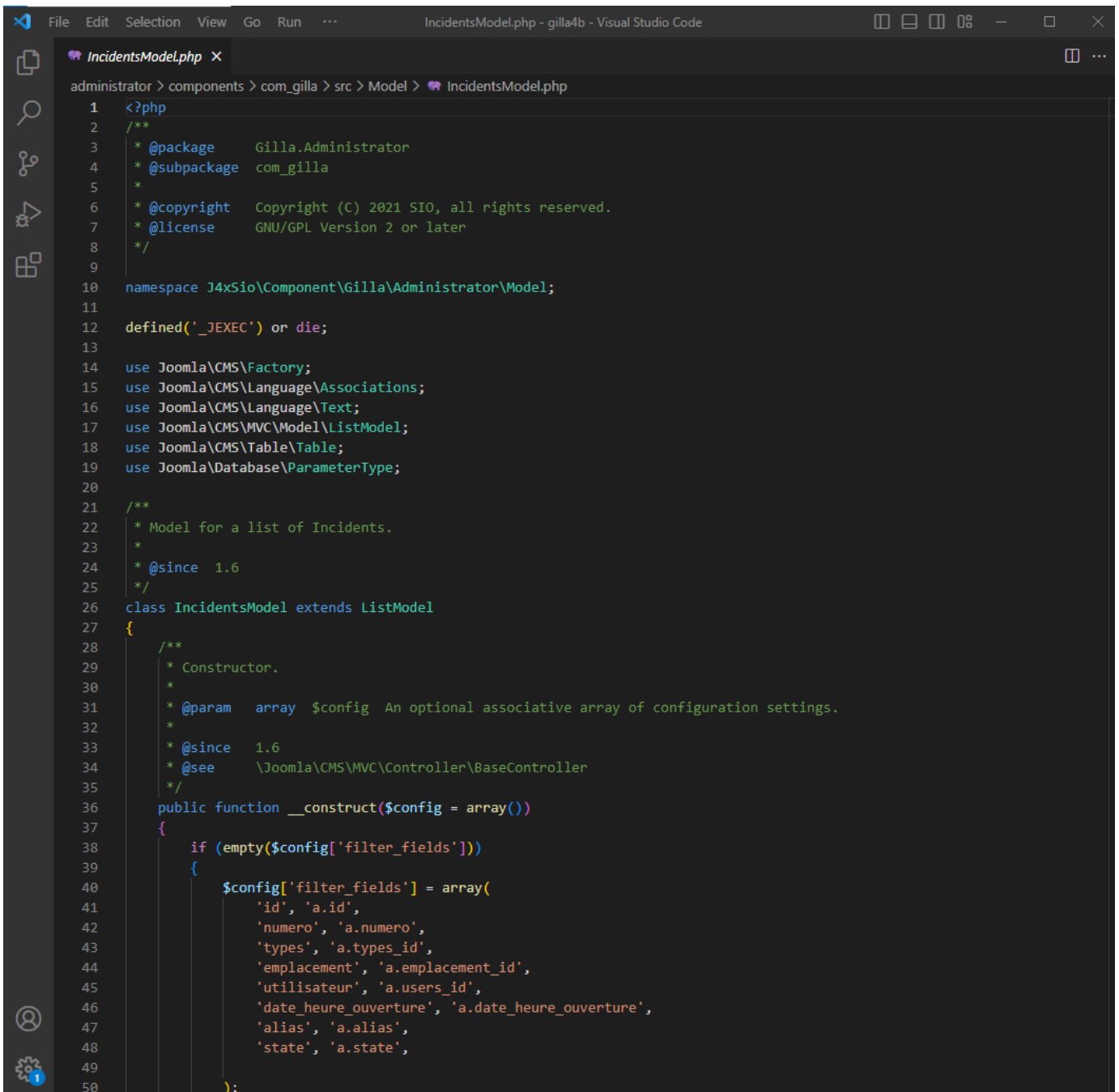


```

1  <?php
2  /**
3   * @package      Gilla.Administrator
4   * @subpackage    com_gilla
5   *
6   * @copyright     Copyright (C) 2021 SIO, all rights reserved.
7   * @license       GNU/GPL Version 2 or later
8   */
9
10 namespace J4xSio\Component\Gilla\Administrator\Controller;
11
12 defined('_JEXEC') or die;
13
14 use Joomla\CMS\MVC\Controller\AdminController;
15
16 /**
17  * Controller for a list of Incidents.
18  *
19  * @since 1.6
20  */
21 class IncidentsController extends AdminController
22 {
23     /**
24      * Proxy for getModel.
25      *
26      * @param string $name The model name. Optional.
27      * @param string $prefix The class prefix. Optional.
28      * @param array $config The array of possible config values. Optional.
29      *
30      * @return \Joomla\CMS\MVC\Model\BaseDatabaseModel
31      *
32      * @since 1.6
33      */
34     public function getModel($name = 'Incidents', $prefix = 'Administrator', $config = array('ignore_request' => true))
35     {
36         return parent::getModel($name, $prefix, $config);
37     }
38 }

```

Modèle « IncidentsModel.php » :



```

1  <?php
2  /**
3   * @package      Gilla.Administrator
4   * @subpackage    com_gilla
5   *
6   * @copyright     Copyright (C) 2021 SIO, all rights reserved.
7   * @license       GNU/GPL Version 2 or later
8   */
9
10 namespace J4xSio\Component\Gilla\Administrator\Model;
11
12 defined('_JEXEC') or die;
13
14 use Joomla\CMS\Factory;
15 use Joomla\CMS\Language\Associations;
16 use Joomla\CMS\Language\Text;
17 use Joomla\CMS\MVC\Model\ListModel;
18 use Joomla\CMS\Table\Table;
19 use Joomla\Database\ParameterType;
20
21 /**
22  * Model for a list of Incidents.
23  *
24  * @since 1.6
25  */
26 class IncidentsModel extends ListModel
27 {
28     /**
29      * Constructor.
30      *
31      * @param array $config An optional associative array of configuration settings.
32      *
33      * @since 1.6
34      * @see \Joomla\CMS\MVC\Controller\BaseController
35      */
36     public function __construct($config = array())
37     {
38         if (empty($config['filter_fields']))
39         {
40             $config['filter_fields'] = array(
41                 'id', 'a.id',
42                 'numero', 'a.numero',
43                 'types', 'a.types_id',
44                 'emplacement', 'a.emplacement_id',
45                 'utilisateur', 'a.users_id',
46                 'date_heure_ouverture', 'a.date_heure_ouverture',
47                 'alias', 'a.alias',
48                 'state', 'a.state',
49             );
50         }
51     }
52 }

```

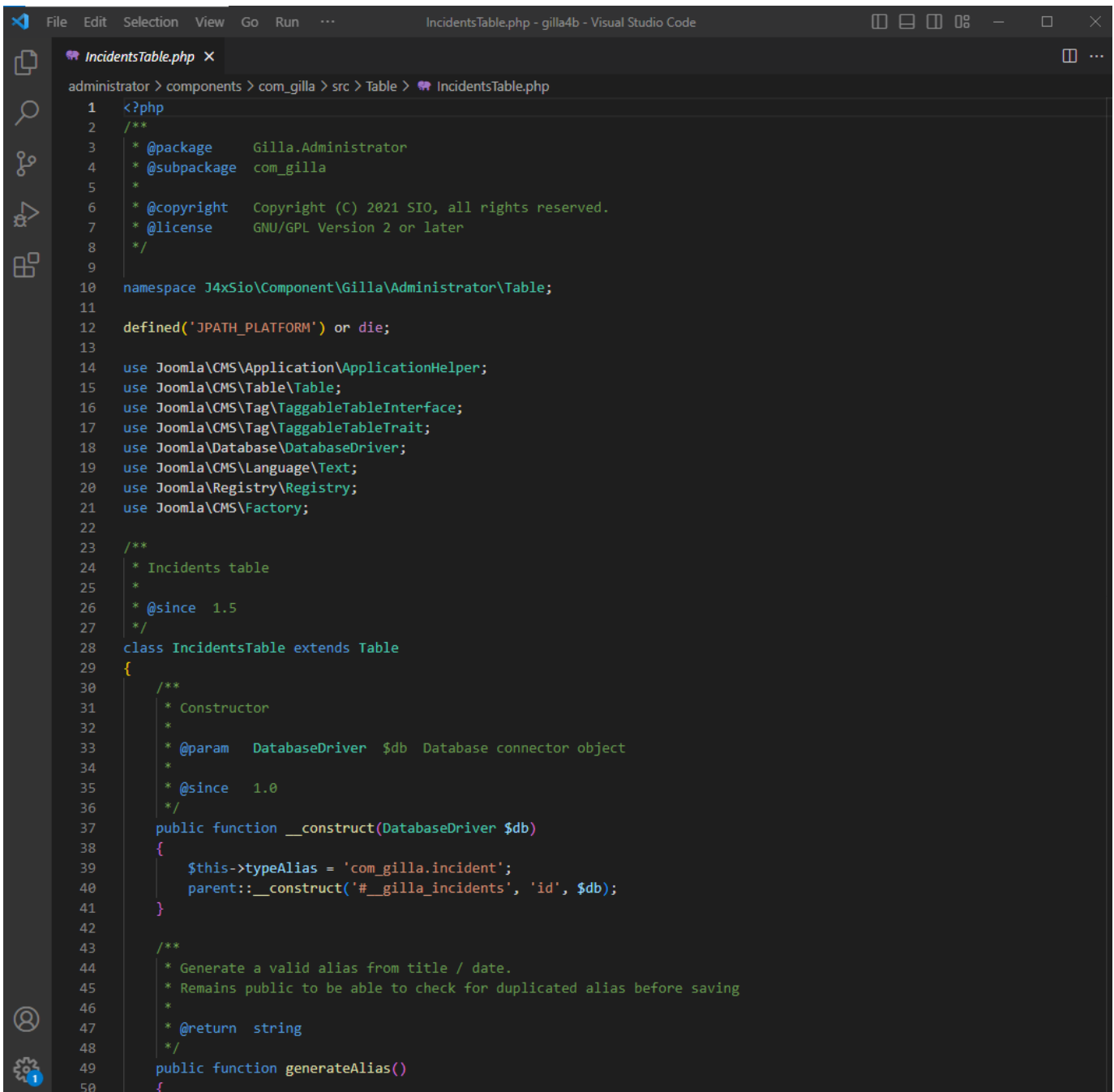
```

IncidentsModel.php x
administrator > components > com_gilla > src > Model > IncidentsModel.php

109     $query = $db->getQuery(true);
110
111     // Select the required fields from the table.
112     $query->select(
113         $this->getState(
114             'list.select',
115             'a.*'
116         )
117     );
118     $query->from('#__gilla_incidents AS a');
119     $query->select('t.type')->join('LEFT', '#__gilla_types AS t ON a.types_id = t.id');
120     $query->select('u.name')->join('LEFT', '#__users AS u ON a.users_id = u.id');
121     $query->select('e.emplacement')->join('LEFT', '#__gilla_emplacements AS e ON a.emplacements_id = e.id');
122
123
124     // Filter by published state.
125     $published = (string) $this->getState('filter.published');
126
127     if (is_numeric($published))
128     {
129         $query->where($db->quoteName('a.state') . ' = :published');
130         $query->bind(':published', $published, ParameterType::INTEGER);
131     }
132     elseif ($published === '')
133     {
134         $query->where('(' . $db->quoteName('a.state') . ' = 0 OR ' . $db->quoteName('a.state') . ' = 1)');
135     }
136
137     // Filter by search in title or description.
138     $search = $this->getState('filter.search');
139
140     if (!empty($search))
141     {
142         $search = $db->quote('%' . str_replace(' ', '%', $db->escape(trim($search), true) . '%'));
143         $searches = array();
144         $searches[] = 'a.id LIKE '.$search;
145         $query->where('(' . implode(' OR ', $searches) . ')');
146     }
147
148     // Add the list ordering clause.
149     $orderCol = $this->state->get('list.ordering', 'a.id');
150     $orderDirn = $this->state->get('list.direction', 'ASC');
151
152     $query->order($db->escape($orderCol) . ' ' . $db->escape($orderDirn));
153     //echo nl2br(str_replace('#__', 'gilla4b_', $query)); // TEST/DEBUG
154     return $query;
155 }
156
157 /**
158  * Method to get a list of Incidents

```

Table « IncidentsTable.php » :

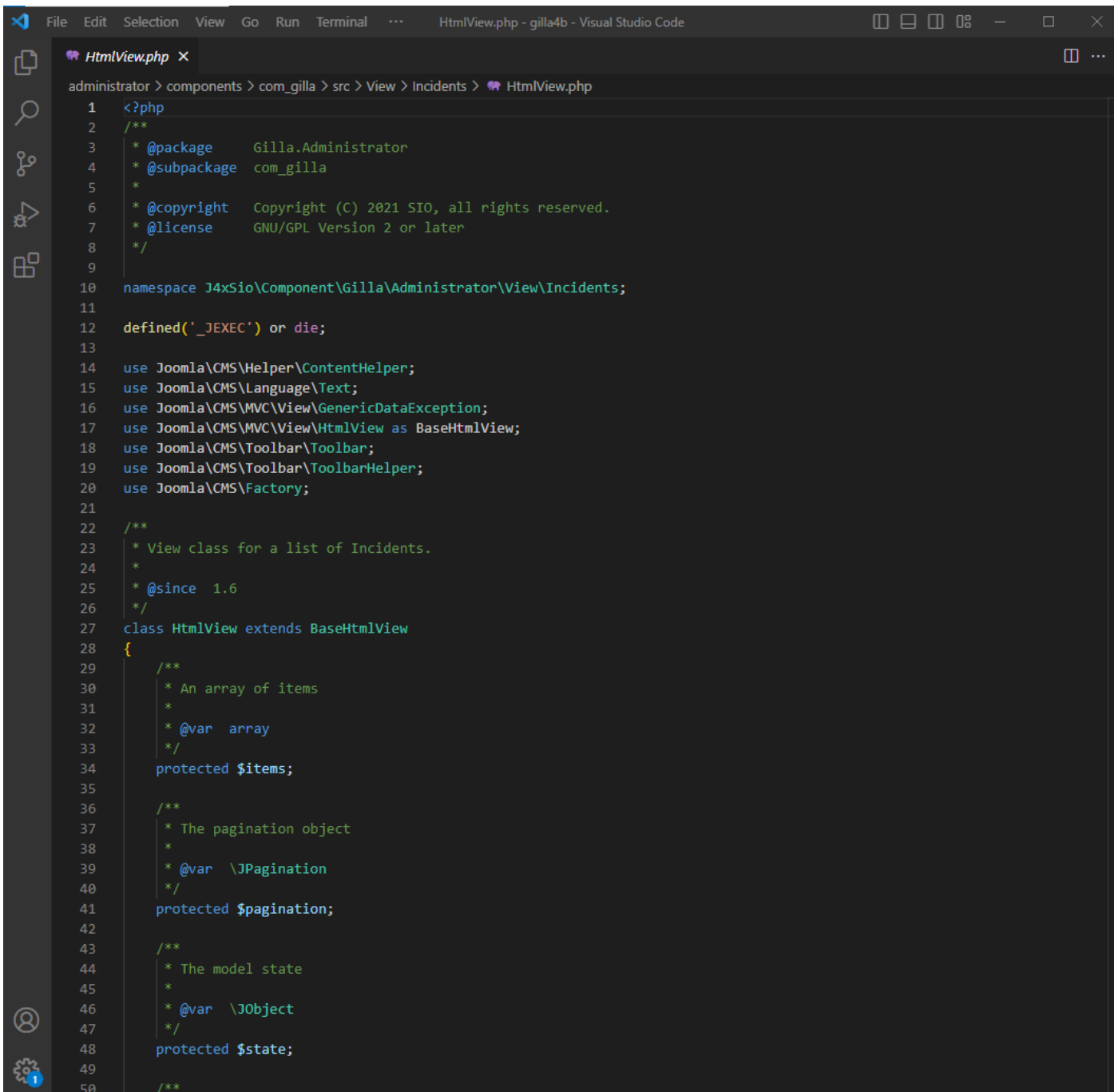


```

1  <?php
2  /**
3   * @package      Gilla.Administrator
4   * @subpackage    com_gilla
5   *
6   * @copyright     Copyright (C) 2021 SIO, all rights reserved.
7   * @license       GNU/GPL Version 2 or later
8   */
9
10 namespace J4xSio\Component\Gilla\Administrator\Table;
11
12 defined('JPATH_PLATFORM') or die;
13
14 use Joomla\CMS\Application\ApplicationHelper;
15 use Joomla\CMS\Table\Table;
16 use Joomla\CMS\Tag\TaggableTableInterface;
17 use Joomla\CMS\Tag\TaggableTableTrait;
18 use Joomla\Database\DatabaseDriver;
19 use Joomla\CMS\Language\Text;
20 use Joomla\Registry\Registry;
21 use Joomla\CMS\Factory;
22
23 /**
24  * Incidents table
25  *
26  * @since 1.5
27  */
28 class IncidentsTable extends Table
29 {
30     /**
31      * Constructor
32      *
33      * @param DatabaseDriver $db Database connector object
34      *
35      * @since 1.0
36      */
37     public function __construct(DatabaseDriver $db)
38     {
39         $this->typeAlias = 'com_gilla.incident';
40         parent::__construct('#__gilla_incidents', 'id', $db);
41     }
42
43     /**
44      * Generate a valid alias from title / date.
45      * Remains public to be able to check for duplicated alias before saving
46      *
47      * @return string
48      */
49     public function generateAlias()
50     {

```

Vue « Incidents/HtmlView.php » :



```

1  <?php
2  /**
3   * @package      Gilla.Administrator
4   * @subpackage    com_gilla
5   *
6   * @copyright     Copyright (C) 2021 SIO, all rights reserved.
7   * @license       GNU/GPL Version 2 or later
8   */
9
10 namespace J4xSio\Component\Gilla\Administrator\View\Incidents;
11
12 defined('_JEXEC') or die;
13
14 use Joomla\CMS\Helper\ContentHelper;
15 use Joomla\CMS\Language\Text;
16 use Joomla\CMS\MVC\View\GenericDataException;
17 use Joomla\CMS\MVC\View\HtmlView as BaseHtmlView;
18 use Joomla\CMS\Toolbar\Toolbar;
19 use Joomla\CMS\Toolbar\ToolbarHelper;
20 use Joomla\CMS\Factory;
21
22 /**
23  * View class for a list of Incidents.
24  *
25  * @since 1.6
26  */
27 class HtmlView extends BaseHtmlView
28 {
29     /**
30      * An array of items
31      *
32      * @var array
33      */
34     protected $items;
35
36     /**
37      * The pagination object
38      *
39      * @var \JPagination
40      */
41     protected $pagination;
42
43     /**
44      * The model state
45      *
46      * @var \JObject
47      */
48     protected $state;
49
50     /**

```

```

51  * Form object for search filters
52  *
53  * @var \JForm
54  */
55  public $filterForm;
56
57  /**
58  * The active search filters
59  *
60  * @var array
61  */
62  public $activeFilters;
63
64  /**
65  * Display the view.
66  *
67  * @param string $tpl The name of the template file to parse; automatically searches through the template paths.
68  *
69  * @return mixed A string if successful, otherwise an Error object.
70  */
71  public function display($tpl = null)
72  {
73      $this->items      = $this->get('Items');
74      $this->pagination = $this->get('Pagination');
75      $this->state      = $this->get('State');
76      $this->filterForm = $this->get('FilterForm');
77      $this->activeFilters = $this->get('ActiveFilters');
78
79      // Check for errors.
80      if (count($errors = $this->get('Errors')))
81      {
82          throw new GenericDataException(implode("\n", $errors), 500);
83      }
84
85      $this->addToolBar();
86
87      return parent::display($tpl);
88  }
89

```

Template « incidents/default.php » :

```

File Edit Selection View Go Run Terminal Help default.php - gilla4b - Visual Studio Code
default.php x
administrator > components > com_gilla > tmpl > incidents > default.php
1 <?php
2 /**
3  * @package      Gilla.Administrator
4  * @subpackage    com_gilla
5  *
6  * @copyright     Copyright (C) 2021 SIO, all rights reserved.
7  * @license       GNU/GPL Version 2 or later
8  */
9
10 defined('_JEXEC') or die;
11
12 use Joomla\CMS\HTML\HTMLHelper;
13 use Joomla\CMS\Language\Text;
14 use Joomla\CMS\Layout\LayoutHelper;
15 use Joomla\CMS\Router\Route;
16 use Joomla\CMS>Date\Date;
17
18 HTMLHelper::_('behavior.multiselect');
19
20 $listOrder = $this->escape($this->state->get('list.ordering'));
21 $listDirn = $this->escape($this->state->get('list.direction'));
22 $states = array (
23     '0' => Text::_('JUNPUBLISHED'),
24     '1' => Text::_('JPUBLISHED'),
25     '2' => Text::_('JARCHIVED'),
26     '-2' => Text::_('JTRASHED')
27 );
28 $editIcon = '<span class="fa fa-pen-square me-2" aria-hidden="true"></span>';
29 ?>
30 <form action="<?php echo Route::_('index.php?option=com_gilla&view=incidents'); ?>" method="post" name="adminForm" id="adm
31 <div class="row">
32 <div class="col-md-12">
33 <div id="j-main-container" class="j-main-container">
34 <?php echo LayoutHelper::render('joomla.searchtools.default', array('view' => $this)); ?>
35 <?php if (empty($this->items)) : ?>
36 <div class="alert alert-info">
37 <span class="fa fa-info-circle" aria-hidden="true"></span><span class="sr-only"><?php echo Text::
38 <?php echo Text::_('JGLOBAL_NO_MATCHING_RESULTS'); ?>
39 </div>
40 <?php else : ?>
41 <table class="table" id="IncidentsList">
42 <thead>
43 <tr>
44 <th style="width:1%" class="text-center">
45 <?php echo HTMLHelper::_('grid.checkall'); ?>
46 </th>
47
48 <th scope="col" class="w-1 text-center">
49 <?php echo HTMLHelper::_('searchtools.sort', 'JSTATUS', 'a.published', $listDirn, $lis
50 </th>

```



```

File Edit Selection View Go Run Terminal Help default.php - gilla4b - Visual Studio Code
default.php X
administrator > components > com_gilla > tmpl > incidents > default.php
69 </tr>
70 <tbody>
71 <tr>
72 <td>
73 <div>
74 <div>
75 <div>
76 <div>
77 <div>
78 <div>
79 <div>
80 <div>
81 <div>
82 <div>
83 <div>
84 <div>
85 <div>
86 <div>
87 <div>
88 <div>
89 <div>
90 <div>
91 <div>
92 <div>
93 <div>
94 <div>
95 <div>
96 <div>
97 <div>
98 <div>
99 <div>
100 <div>
101 <div>
102 <div>
103 <div>
104 <div>
105 <div>
106 <div>
107 <div>
108 <div>
109 <div>
110 <div>
111 <div>
112 <div>
113 <div>
114 <div>
115 <div>
116 <div>
117 <div>
118 <div>

```

Codage MVC du frontend de com_gilla

Les contrôleurs, modèles et vues constituant le code PHP du composant com_gilla selon l'arborescence et les règles de nommage des fichiers définis plus-haut sont conçus comme des classes PHP héritant des classes correspondantes du CMS Joomla.

Par exemple, la vue d'**édition d'un incident** est constituée des fichiers de code suivants :
(#09) Melvyn

Formulaire « incident.xml » :

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <form>
3      <fieldset>
4          <field
5              name="id"
6              type="number"
7              label="JGLOBAL_FIELD_ID_LABEL"
8              default="0"
9              class="readonly"
10             readonly="true"
11         />
12
13         <field
14             name="numero"
15             type="text"
16             label="COM_GILLA_INCIDENTS_NUMERO"
17             size="40"
18             required="true"
19         />
20
21         <field
22             name="date_heure_ouverture"
23             type="calendar"
24             label="COM_GILLA_INCIDENTS_DATE_HEURE_OUVERTURE"
25             translateformat="true"
26             showtime="true"
27             size="22"
28             filter="user_utc"
29             required="true"
30         />
31
32         <field
33             name="description"
34             type="textarea"
35             label="COM_GILLA_INCIDENTS_DESCRIPTION"
36             rows="5"
37             cols="40"
38             required="true"
39         />
40
41         <field
42             name="date_heure_fermeture"
43             type="calendar"
44             label="COM_GILLA_INCIDENTS_DATE_HEURE_FERMETURE"
45             translateformat="true"
46             showtime="true"
47             size="22"
48             filter="user_utc"
49             required="false"

```

Contrôleur « IncidentController.php » :

```

1  <?php
2  /**
3   * @package      Gilla.Site
4   * @subpackage    com_gilla
5   *
6   * @copyright     Copyright (C) 2021 SIO, all rights reserved.
7   * @license       GNU/GPL Version 2 or later
8   */
9
10 namespace J4xSio\Component\Gilla\Site\Controller;
11
12 \defined('_JEXEC') or die;
13
14 use Joomla\CMS\Factory;
15 use Joomla\CMS\MVC\Controller\FormController;
16 use Joomla\CMS\Router\Route;
17 use Joomla\CMS\Uri\Uri;
18 use Joomla\Utilities\ArrayHelper;
19
20 /**
21  * Controller for single foo view
22  *
23  * @since 4.0.0
24  */
25 class IncidentController extends FormController
26 {
27     /**
28      * The URL view item variable.
29      *
30      * @var string
31      * @since 4.0.0
32      */
33     protected $view_item = 'incident_e';
34
35     /**
36      * Method to get a model object, loading it if required.
37      *
38      * @param string $name The model name. Optional.
39      * @param string $prefix The class prefix. Optional.
40      * @param array $config Configuration array for model. Optional.
41      *
42      * @return \Joomla\CMS\MVC\Model\BaseDatabaseModel The model.
43      *
44      * @since 4.0.0
45      */
46     public function getModel($name = 'incident_e', $prefix = '', $config = ['ignore_request' => true])
47     {
48         return parent::getModel($name, $prefix, ['ignore_request' => false]);
49     }
50
51     /**
52      * Method override to check if you can add a new record.
53      *

```

Modèle « Incident_eModel.php » :

```

1  <?php
2  /**
3   * @package      Gilla.Site
4   * @subpackage   com_gilla
5   *
6   * @copyright    Copyright (C) 2021 SIO, all rights reserved.
7   * @license      GNU/GPL Version 2 or later
8   */
9
10 namespace J4xSio\Component\Gilla\Site\Model;
11
12 defined('_JEXEC') or die;
13
14 use Joomla\CMS\Factory;
15 use Joomla\CMS\Form\Form;
16 use Joomla\CMS\Language\Associations;
17 use Joomla\CMS\Language\Multilanguage;
18 use Joomla\Registry\Registry;
19 use Joomla\Utilities\ArrayHelper;
20
21 /**
22  * Incident Component Incident Model
23  *
24  * @since 4.0.0
25  */
26 class Incident_eModel extends \J4xSio\Component\Gilla\Administrator\Model\IncidentModel
27 {
28     /**
29      * Model typeAlias string. Used for version history.
30      *
31      * @var string
32      * @since 4.0.0
33      */
34     public $typeAlias = 'com_gilla.incident';
35
36     /**
37      * Name of the form
38      *
39      * @var string
40      * @since 4.0.0
41      */
42     protected $formName = 'incident_e';
43
44     /**
45      * Method to get the row form.
46      *
47      * @param array $data Data for the form.
48      * @param boolean $loadData True if the form is to load its own data (default case), false if not.
49      *
50      * @return \JForm|boolean A \JForm object on success, false on failure
51      *
52      * @since 4.0.0
53      */

```

Vue « Incident_e/HtmlView.php » :

```

1  <?php
2  /**
3   * @package      Gilla.Site
4   * @subpackage   com_gilla
5   *
6   * @copyright    Copyright (C) 2021 SIO, all rights reserved.
7   * @license      GNU/GPL Version 2 or later
8   */
9
10 namespace J4xSio\Component\Gilla\Site\View\Incident_e;
11
12 defined('_JEXEC') or die;
13
14 use Joomla\CMS\Factory;
15 use Joomla\CMS\Language\Multilanguage;
16 use Joomla\CMS\Language\Text;
17 use Joomla\CMS\MVC\View\HtmlView as BaseHtmlView;
18 use J4xSio\Component\Gilla\Administrator\Helper\GillaHelper;
19
20 /**
21  * HTML Incident_e View class for the Incident component
22  *
23  * @since 4.0.0
24  */
25 class HtmlView extends BaseHtmlView
26 {
27     /**
28      * @var      \Joomla\CMS\Form\Form
29      * @since 4.0.0
30      */
31     protected $form;
32
33     /**
34      * @var      object
35      * @since 4.0.0
36      */
37     protected $item;
38
39     /**
40      * @var      string
41      * @since 4.0.0
42      */
43     protected $return_page;
44
45     /**
46      * @var      string
47      * @since 4.0.0
48      */
49     protected $pageclass_sfx;
50
51     /**
52      * @var      \Joomla\Registry\Registry
53      * @since 4.0.0

```

Template « incident_e/edit.php » :

```

1  <?php
2  /**
3   * @package      Gilla.Site
4   * @subpackage    com_gilla
5   *
6   * @copyright     Copyright (C) 2021 SIO, all rights reserved.
7   * @license       GNU/GPL Version 2 or later
8   */
9
10 defined('_JEXEC') or die;
11
12 use Joomla\CMS\HTML\HTMLHelper;
13 use Joomla\CMS\Language\Multilanguage;
14 use Joomla\CMS\Language\Text;
15 use Joomla\CMS\Layout\LayoutHelper;
16 use Joomla\CMS\Router\Route;
17 use J4xSio\Component\Gilla\Site\Helper\RouteHelper as GillaHelperRoute;
18
19 HTMLHelper::_('behavior.Keepalive');
20 HTMLHelper::_('behavior.formvalidator');
21
22 $this->tab_name = 'com-gilla-incident_e';
23 $this->ignore_fieldsets = ['publishing', 'jmetadata'];
24 $this->useCoreUI = true;
25 ?>
26 <form action="<?php echo Route::_('index.php?option=com_gilla&id=' . (int) $this->item->id); ?>"
27 method="post" name="adminForm" id="adminForm" class="form-validate form-vertical">
28     <fieldset>
29         <?php echo HTMLHelper::_('uitab.startTabSet', $this->tab_name, ['active' => 'details']); ?>
30
31         <?php echo HTMLHelper::_('uitab.addTab', $this->tab_name, 'details', Text::_
32 ('COM_GILLA_INCIDENT')); ?>
33         <?php echo $this->form->renderField('numero'); ?>
34         <?php echo $this->form->renderField('date_heure_ouverture'); ?>
35         <?php echo $this->form->renderField('emplacements_id'); ?>
36         <?php echo $this->form->renderField('types_id'); ?>
37         <?php echo $this->form->renderField('description'); ?>
38         <?php echo $this->form->renderField('date_heure_fermeture'); ?>
39         <?php echo $this->form->renderField('commentaire_agent'); ?>
40         <?php echo $this->form->renderField('users_id'); ?>
41         <?php echo HTMLHelper::_('uitab.endTab'); ?>
42
43         <?php echo HTMLHelper::_('uitab.endTabSet'); ?>
44
45         <input type="hidden" name="task" value="">
46         <input type="hidden" name="return" value="<?php echo $this->return_page; ?>">
47         <?php echo HTMLHelper::_('form.token'); ?>
48     </fieldset>
49 </div class="mb-2">

```

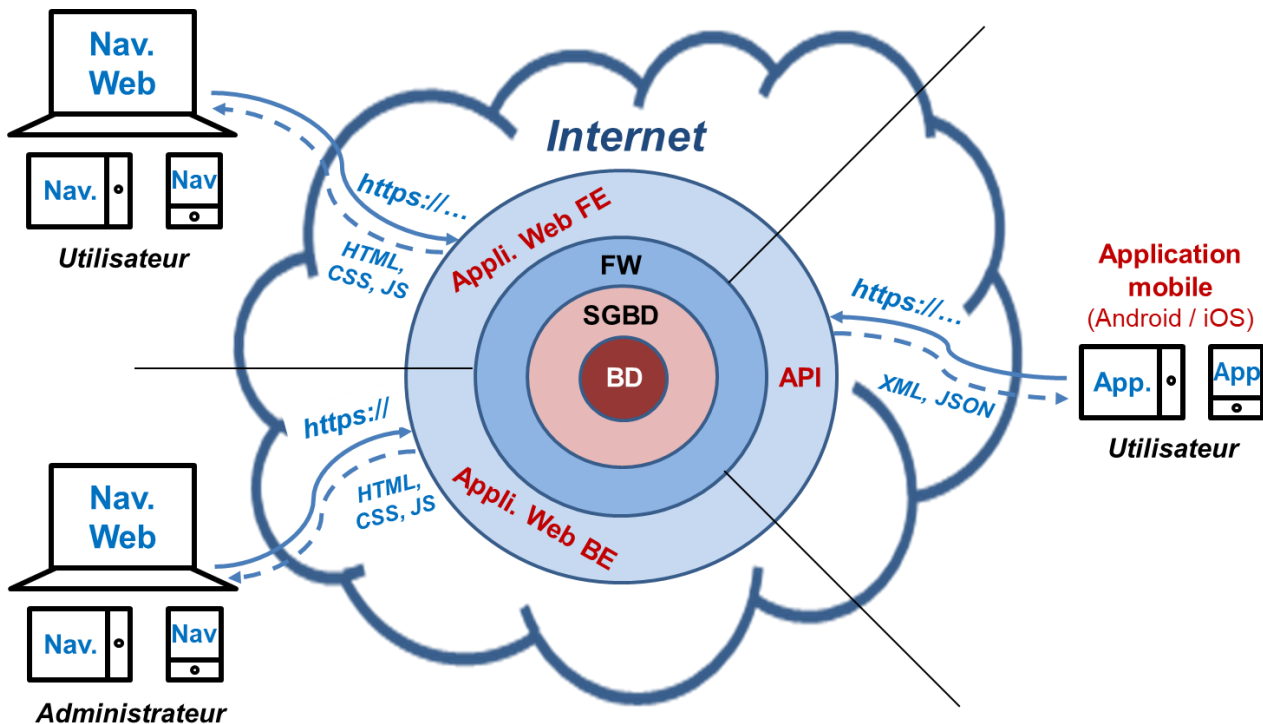
Application AndroidGilla

Architecture de la solution

Côté serveur, une API de test a été spécialement développée en PHP pour pouvoir consulter la base de données GILLA à travers le framework Joomla par le protocole http.

Côté client, l'application Android a été développée en Java pour permettre à l'utilisateur « client » disposant d'une tablette ou d'un smartphone Android de se connecter à l'API par le protocole http et consulter ensuite les offres et ses candidatures.

En environnement de développement, serveur et terminal Android sont connectés au même réseau local.



L'API du site GILLA dispose de cinq points d'entrée (*entry points*) de type CRUD (avec deux *Read* : liste et unitaire) par table (incidents ici) :

- | | |
|---|---|
| • POST gilla/incidents (+ body) : | création d'un incident. |
| • GET gilla/incidents : | consultation de la liste des incidents. |
| • GET gilla/incidents /{incident_id} : | consultation de l'incident <i>incident_id</i> . |
| • PATCH gilla/incidents /{incident_id} (+ body) : | modification de l'incident <i>incident_id</i> . |
| • DEL gilla/incidents /{incident_id} : | suppression de l'incident <i>incident_id</i> . |

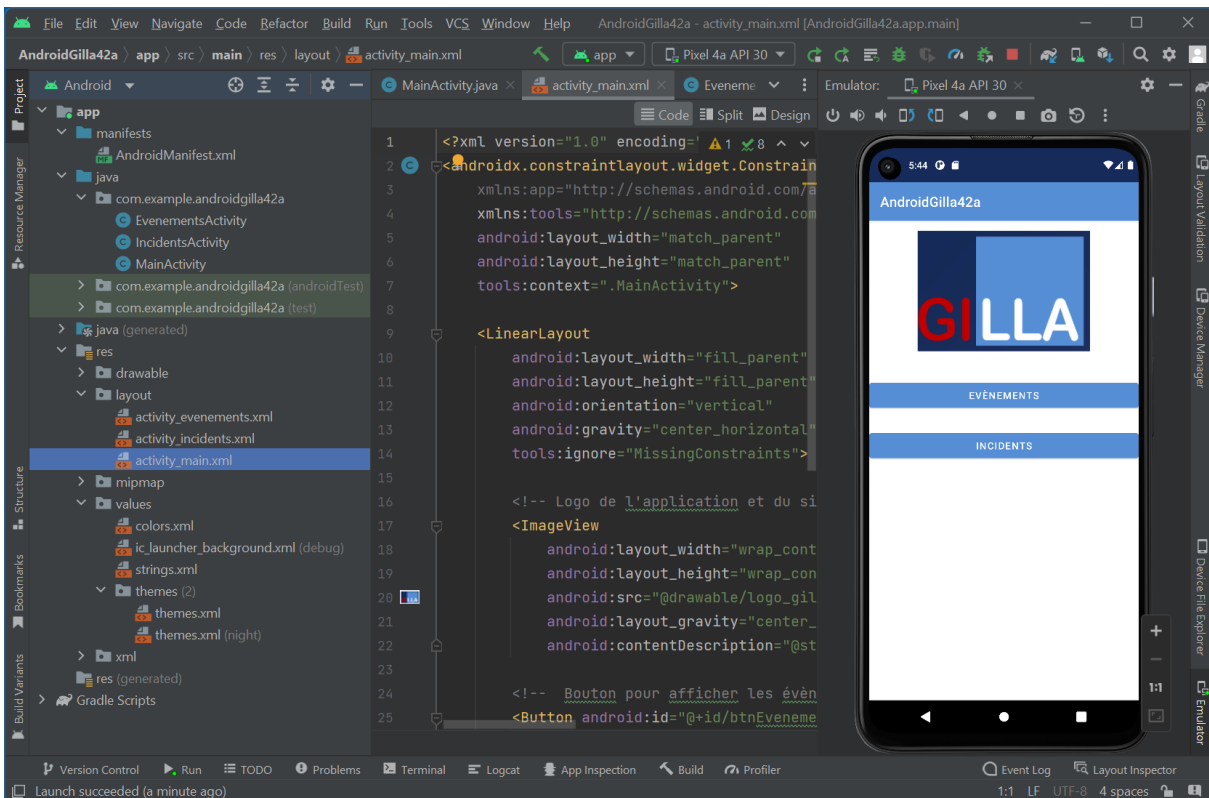
Android Studio

Conformément aux dernières recommandations de Google, l'environnement de développement Android Studio a été choisi pour concevoir et tester l'application AndroidGilla.

Android Studio est un IDE qui permet de développer des applications Android en Java, il bénéficie aussi de la technologie WYSIWYG (What You See Is What You Get) qui permet de voir et gérer la mise en page à côté de l'XML.

Structure et code de l'application

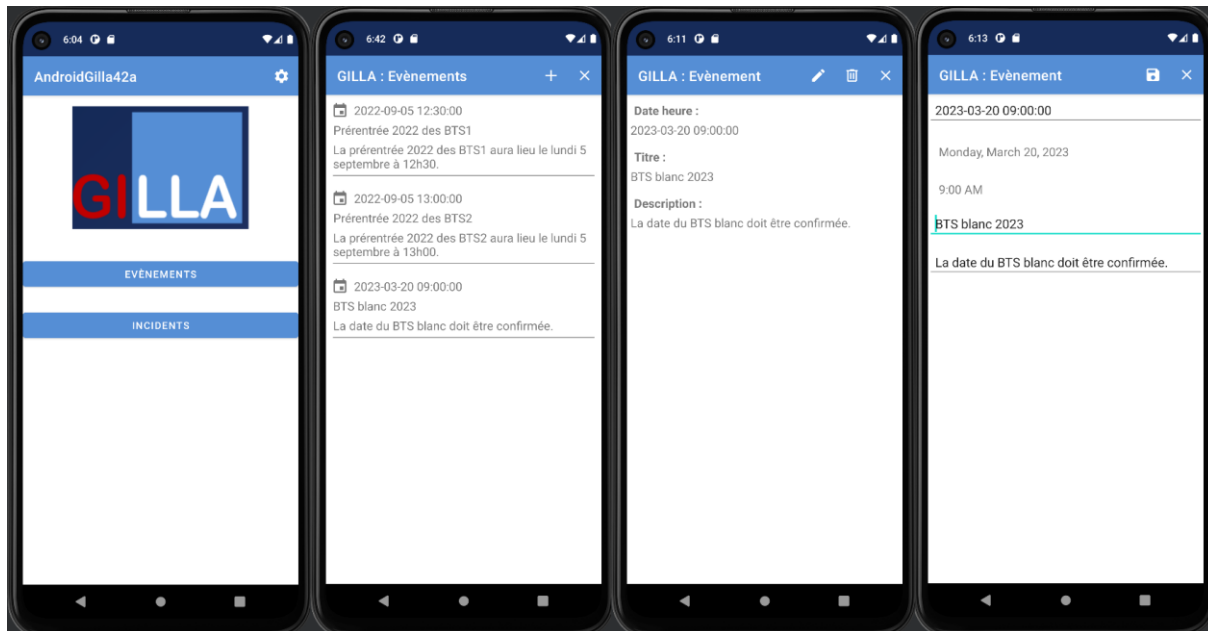
La structure de l'application AndroidGilla est la suivante :



L'arborescence « app » contient les dossiers « manifests », « java » qui contient les activités (contrôleurs et modèles codés en JAVA) et « res » qui contient notamment le dossier « layout » des écrans codés en XML (vues).

Tests de l'application

Une fois l'application lancée, la sélection dans le **menu principal** de « Evènements » affiche la **vue liste** et les **vues de détail** du 3^e évènement comme suit (de gauche à droite) :



Intégration et tests d'intégration du site GILLA

Création du composant packagé com_gilla.zip

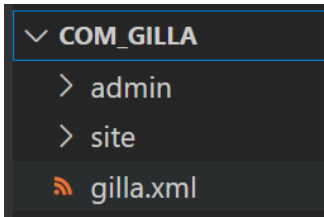
Il s'agit de créer un composant packagé **com_gilla.zip** installable depuis le CMS Joomla 4.2. Le **manifeste gilla.xml** décrit les attributs et les dossiers et fichiers d'installation des tables et des arborescences du frontend et du backend du composant comme suit :

```

administrator > components > com_gilla > gilla.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <extension type="component" method="upgrade">
3      <name>com_gilla</name>
4      <creationDate>Oct 2022</creationDate>
5      <author>SIO</author>
6      <copyright>Copyright (C) 2021-2022 SIO, all rights reserved.</copyright>
7      <license>GNU/GPL Version 2 or later - http://www.gnu.org/licenses/gpl-2.0.html</license>
8      <version>0.6.1</version>
9      <description>COM_GILLA_DESCRIPTION</description>
10     <namespace path="src">J4xSio\Component\Gilla</namespace>
11
12 >     <!-- Runs on install -->...
16 </install>
17 >     <!-- Runs on uninstall -->...
21 </uninstall>
22
23     <!-- Frontend Site Section -->
24     <files folder="site">
25         <folder>forms</folder>
26         <folder>src</folder>
27         <folder>tmpl</folder>
28     </files>
29     <languages folder="site">
30         <language tag="en-GB">language/en-GB/com_gilla.ini</language>
31         <language tag="fr-FR">language/fr-FR/com_gilla.ini</language>
32     </languages>
33
34     <!-- Backend Site Section -->
35     <administration>
36         <menu img="class:default" link="option=com_gilla">com_gilla</menu>
37 >         <submenu>...
47 </submenu>
48 >         <files folder="admin">...
56 </files>
57 >         <languages folder="admin">...
62 </languages>
63 </administration>
64
65     <!-- API Section -->
66     <api>
67         <files folder="api">
68             <folder>src</folder>
69         </files>
70     </api>
71 </extension>

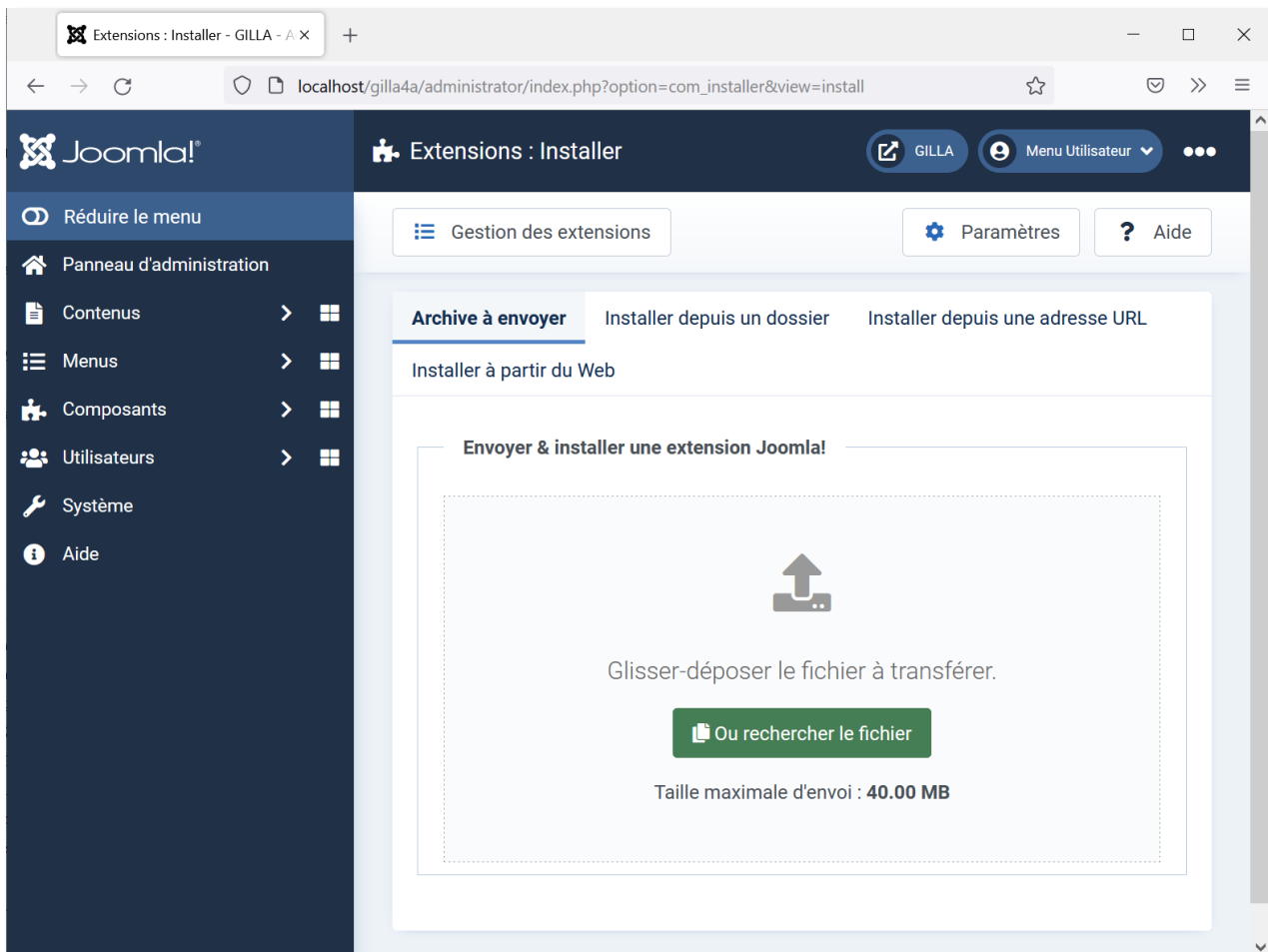
```

Le dossier compressé contient ce fichier manifeste et les deux arborescence sources :



Test d'installation du composant packagé com_gilla.zip

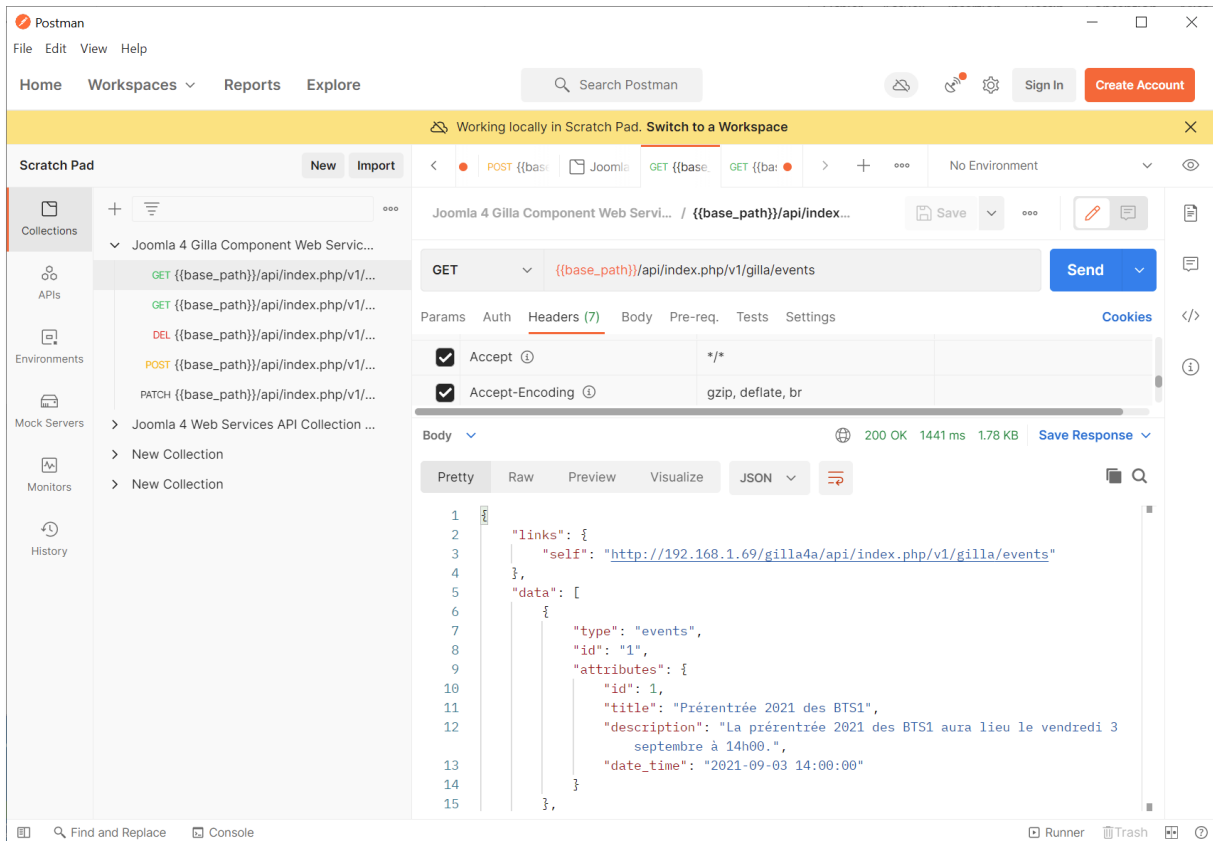
Il s'agit de vérifier la bonne installation du composant **com_gilla.zip** à partir du menu du backend de Joomla : Système / Installation / Extensions. Un glisser-déposer du fichier com_gilla.zip dans le cadre centre ou une recherche du fichier sur le système déclenche l'installation du composant. La bonne installation est confirmée par un message sur fond vert : « Installation du composant réussie ».



Test de l'API

Le test de l'API du composant com_gilla se fait avec le logiciel **Postman** et une collection de requêtes développée au format JSON puis importée dans ce logiciel.

Après paramétrage du Token obtenu dans le backend de GILLA (onglet « Jeton Joomla » du super utilisateur SLAM) et du chemin « base_path » définissant la racine du site, il est possible d'exécuter les requêtes prévues dans la collection :



On peut ainsi analyser les réponses JSON envoyées par l'API du site GILLA.

Tests et recette fonctionnelle

Tests du backend GILLA

Le cahier de tests du backend GILLA comprend pour chaque cas d'utilisation identifié lors des spécifications une fiche de test selon modèle suivant :

(#01 : BE com_gilla – Consulter Prises en charge)

Tests du frontend GILLA

Les tests du frontend GILLA se sont déroulés selon le même principe que ceux du backend, à partir d'un cahier de tests composé de fiches de tests associées aux cas d'utilisation spécifiés dans le frontend, comme par exemple :

(#02 : FE com_gilla – Consulter mes incidents)

Tests de l'application AndroidGilla

Les tests de l'application AndroidGilla ont consisté à vérifier le bon affichage d'un certain nombre d'écrans comme par exemple :

(#03)

Déploiement

Le déploiement du site GILLA s'est fait sur une plateforme mutualisée LAMP hébergée chez OVH, après achat du nom de domaine disponible lla-sio.fr et de l'hébergement Pro, de la façon suivante :

1. Préparation des fichiers à télécharger :
 - Export de la base de données GILLA complète au format SQL (CMS Joomla + extension com_gilla).
 - Copie du fichier GILLA.sql à la racine de l'arborescence des fichiers du site.
2. Préparation de la plateforme d'hébergement :
 - Création du sous domaine GILLA dans www.la-sio.fr.
 - Création du répertoire GILLA à la racine www du site (correspondant au sous-domaine).
 - Création de la base de données sous MySQL.
3. Téléchargement des fichiers :
 - Téléchargement de l'arborescence des fichiers par FTP dans le répertoire www du serveur OVH.
4. Installation de la base de données :
 - Import du fichier GILLA.sql (structure et contenu de la base de données) avec phpMyAdmin (serveur OVH).
5. Tests :
 - Test du site public : <https://gilla4a.sio.louis-armand.paris> (ou ...gilla4b...)
 - Test du site d'administration : <https://gilla4a.sio.louis-armand.paris/administrator> (ou ...gilla4b...)