VIGNON CHAUDEY
Clément

Rapport individuel SAE 3.01c

Synthèse :

Le but de notre projet MobiList est de répertorier l'inventaire des objets mobiliers couverts par une assurance dans un domicile. Cet inventaire est essentiel pour la déclaration de sinistre auprès de l'assurance en cas de catastrophe naturelle entraînant des dégâts matériels. Le but est d'automatiser la mise à jour de l'inventaire lors d'une nouvelle acquisition et de conserver électroniquement les justificatifs d'achat, les factures et les photos. L'objectif est d'évaluer un état financier des pertes potentielles, pièce par pièce, en appliquant une vétusté basée sur le prix d'achat et la date du sinistre.
Pour cette première semaine, nous devions faire le document de suivi de projet présentant la liste des tâches à effectuer et la répartition des tâches pour cette semaine, faire le site web côté Client et le doss
Le plus dur dans cette 2ème semaine a forcément été dans le code mais notamment le code dans le views.py. Il a fallu réussir à bien lier le front-end et le back-end c'est-à-dire la BD et les pages webs. Par exemple pour supprimer un logement, au départ je supprimais juste le logement sans rien supprimer d'autre et je ne coier de spécifications fonctionnelles. Nous avons tous fait du code durant les 4 premiers jours puis nous nous sommes occupés du rapport collectif. De mon côté je me suis surtout occupé de coder les pages qui parle qui parle du logement, des pièces et des paramètres.

Analyse :

mprenais pas pourquoi ça ne marchait pas. En utilisant le debugguage, j'ai eu un message d'erreur qui disait que la contrainte « NOT NULL » n'était pas respecté si je supprimais le logement. Ainsi, avant de supprimer le logement j'ai supprimé tout ce qui était lié au logement donc tous les biens, pièces et sinistres lié à ce logement.

Avant :

```python
@app.route('/logement/<int:id>/delete', methods=['POST'])
@login_required
def delete_logement(id):
    logement = Logement.query.get_or_404(id)
    try:
        db.session.delete(logement)
        db.session.commit()
        flash('Logement et toutes ses pièces et biens supprimés.', 'success')
    except Exception as e:
        db.session.rollback()
        print("Erreur suppression :", e)
        flash(f'Erreur lors de la suppression : {e}', 'danger')
    return redirect(url_for('mes_logements'))
```

VIGNON CHAUDEY
Clément

Après :

```python
@app.route('/logement/<int:id>/delete', methods=['POST'])
@login_required
def delete_logement(id):
    logement = Logement.query.get_or_404(id)
    try:
        for sinistre in logement.sinistres:
            db.session.delete(sinistre)
        for piece in logement.pieces:
            for bien in piece.biens:
                db.session.delete(bien)
            db.session.delete(piece)
        db.session.delete(logement)
        db.session.commit()
        flash('Logement et toutes ses pièces et biens supprimés.', 'success')
    except Exception as e:
        db.session.rollback()
        print("Erreur suppression :", e)
        flash(f'Erreur lors de la suppression : {e}', 'danger')
    return redirect(url_for('mes_logements'))
```

J’ai pu énormément consolider mes connaissances en Flask dans cette 2ème semaine étant donné que j’ai quasi exclusivement fait que de ça. Cela inclut surtout des compétences en BD, Python et Dev Web.

Démonstration de compétences :

AC21.01 | Élaborer et implémenter les spécifications fonctionnelles et non fonctionnelles à partir des exigences

- Analyse des besoins : gestion des logements, pièces, biens, utilisateurs, sinistres.
- Implémentation des fonctionnalités selon les exigences du sujet.
- Prise en compte des contraintes : sécurité, validation des formulaires, gestion des erreurs.

AC21.02 | Appliquer des principes d’accessibilité et d’ergonomie

- Utilisation de labels explicites pour chaque champ de formulaire.
- Boutons d’action clairs et accessibles.
- Messages d’erreur et de succès affichés en couleur et accessibles.
- Navigation cohérente et structurée dans toutes les pages.

AC21.03 | Adopter de bonnes pratiques de conception et de programmation

- Respect des conventions PEP8 pour le code Python.

VIGNON CHAUDEY
Clément

- Organisation du projet en modules : views.py, forms.py, database/, templates/, static/.
- Utilisation de l'ORM SQLAlchemy pour la gestion des données.
- Validation des données côté serveur avec WTForms.

AC22.01 | Choisir des structures de données complexes adaptées au problème

- Utilisation de modèles relationnels pour représenter les liens entre utilisateurs, logements, pièces, biens, sinistres.
- Relations SQLAlchemy : 1-1, 1-N, N-N.
- Utilisation de listes et dictionnaires pour transmettre les données aux templates.

AC23.01 | Concevoir et développer des applications communicantes

- Application web communicante via le protocole HTTP.
- Utilisation de Flask pour gérer les routes et les échanges entre client et serveur.
- Gestion des formulaires et des fichiers.

AC23.02 | Utiliser des serveurs et des services réseaux virtualisés

- Déploiement et exécution de l'application sur un serveur local Linux.
- Utilisation de Flask comme serveur applicatif.

AC24.03 | Organiser la restitution de données à travers la programmation et la visualisation

- Affichage des statistiques dans le tableau de bord.
- Utilisation de tableaux et de cartes pour visualiser les données.
- Restitution claire et synthétique des informations dans les templates HTML.

AC24.04 | Manipuler des données hétérogènes

- Gestion de différents types de données : chaînes, dates, fichiers, nombres.
- Manipulation de données issues de plusieurs modèles .
- Conversion et validation des formats.

AC25.02 | Formaliser les besoins du client et de l'utilisateur

- Rédaction des spécifications fonctionnelles.
- Traduction des besoins en fonctionnalités concrètes dans l'application.

AC25.03 | Identifier les critères de faisabilité d'un projet informatique

- Analyse de la faisabilité technique : choix de Flask, SQLAlchemy, WTForms.
- Prise en compte des contraintes de sécurité, d'ergonomie et de performance.

AC25.04 | Définir et mettre en œuvre une démarche de suivi de projet

VIGNON CHAUDEY
Clément

- Organisation du code en modules et fichiers pour faciliter le suivi.
- Utilisation de messages flash et de logs pour le suivi des actions utilisateur.
- Tests unitaires présents dans le dossier tests/.

AC26.02 | Appliquer une démarche pour intégrer une équipe informatique au sein d’une organisation

- Respect des conventions de développement pour faciliter le travail en équipe.
- Documentation du code et des fonctionnalités.
- Utilisation de Git.

AC26.03 | Mobiliser les compétences interpersonnelles pour travailler dans une équipe informatique

- Communication des choix techniques et des solutions aux membres de l’équipe.
- Collaboration sur la conception et la validation des fonctionnalités.

AC26.04 | Rendre compte de son activité professionnelle

- Rédaction de rapports et de documentation sur l’avancement du projet.
- Présentation des fonctionnalités réalisées et des difficultés rencontrées.
- Utilisation de messages flash et de logs pour tracer les actions.