

DOSSIER DE PROJET

Date de naissance : 04/06/1993
Nom d'usage : RIVIÈRE
Prénom : Adrien
Adresse : 121 rue de Vaugirard
Code postal : 75015 Paris

Pour le passage du titre RNCP

Concepteur Développeur d'Application (CDA)

Titre du projet

Conception d'une application de gestion de contacts

Lieu de réalisation

Centre de formation

*WebForce3 (WF3) Paris 14^e +
Grenoble Ecole de Management (GEM) Campus Paris*

Table des matières

Liste des compétences couvertes par le projet	4
Résumé en anglais du projet	5
Expression des besoins	6
Gestion de projet	8
Spécifications fonctionnelles du projet	10
Wireframes du projet	12
Spécifications techniques	16
Spécifications de sécurité	20
Réalisation de la partie commune de l'application.....	22
Réalisation de la partie connexion de l'application	25
Réalisation de la partie contacts de l'application	29
Gestion de la persistance des données	33
Présentation de la fonctionnalité à tester	34
Jeu d'essai et résultats	36
Veille sur les vulnérabilités de sécurité	44
Situation de travail ayant nécessité une recherche	48
Liste des recherches	50
Annexes	51

Liste des compétences couvertes par le projet

Les compétences non couvertes par le projet sont barrées

Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité :

- Maquetter une application
- Développer une interface utilisateur de type desktop
- Développer des composants d'accès aux données
- Développer la partie front-end d'une interface utilisateur web
- Développer la partie back-end d'une interface utilisateur web

Concevoir et développer la persistance des données en intégrant les recommandations de sécurité :

- Concevoir une base de données
- Mettre en place une base de données
- Développer des composants dans le langage d'une base de données

Concevoir et développer une application multicouche répartie en intégrant les recommandations de sécurité :

- Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement
- Concevoir une application
- Développer des composants métier
- Construire une application organisée en couches
- ~~- Développer une application mobile~~
- Préparer et exécuter les plans de tests d'une application
- Préparer et exécuter le déploiement d'une application

Compétences transversales :

- Utiliser l'anglais dans son activité professionnelle en conception et développement d'applications
- Actualiser et partager ses compétences en conception et développement d'applications

Résumé en anglais du projet

The goal is to create a project in order to validate the qualifications which haven't been acquired during the working periods of my work-study contract in SAP.

I decided to make an application for managing his contacts (that means personal and professional relationships). I thought it covers well the asked abilities for the RNCP certification and it fits my aptitudes.

I decided to program with python language, which is a popular and simple language, and use the django framework that is adapted for my goal. I decided to use VS Code as Integrated Development Environment (IDE). The deployment on internet was made on Heroku website because it was free and suitable for beginners programmers.

The main part of the application displays the contacts table, the social networks table or the parties table depending on the click on the navigation bar's links. The tables are interactive and allow the user to add, update or delete an element by filling forms.

The user need to connect to his account for accessing the application. The administrations pages are ruled by the django framework and some personal settings.

I had planned a calendar page for managing events, but I didn't make it because I needed time to make the documents for the title rncp obtaining.

I worked with a TDD (Test-Driven Development) method in order to test continuously the functionalities and detect malfunctions as soon as possible.

Expression des besoins

L'application a été réalisée en centre de formation, mais l'expression des besoins cherche à mimer une situation en entreprise.

Une entreprise aimerait mettre à disposition des employés une application pour gérer leurs contacts (personnels ou professionnels).

Ces derniers travaillent souvent sur leur ordinateur et il leur est plus facile de gérer leurs contacts par ordinateur. L'application serait utilisable par internet ou téléchargeable en tant qu'application bureau.

L'application doit être souple et facile d'utilisation, et permettre de renseigner les réseaux sociaux et les fêtes d'un contact et aussi les événements (au moyen d'un calendrier interactif par exemple).

Le temps est limité et il est possible que certaines fonctionnalités ne puissent pas être développées dans l'immédiat. Mais les employés exigent une application « avec une belle esthétique ».

Fidèle à l'environnement international de la société (des permanents et consultants de la société sont d'origines étrangères), l'application devra être en anglais afin d'être aisément utilisable par tous. La traduction du navigateur sera désactivée au travers des « templates ».

L'application sera responsive et s'adaptera aux tablettes et aux smartphones.

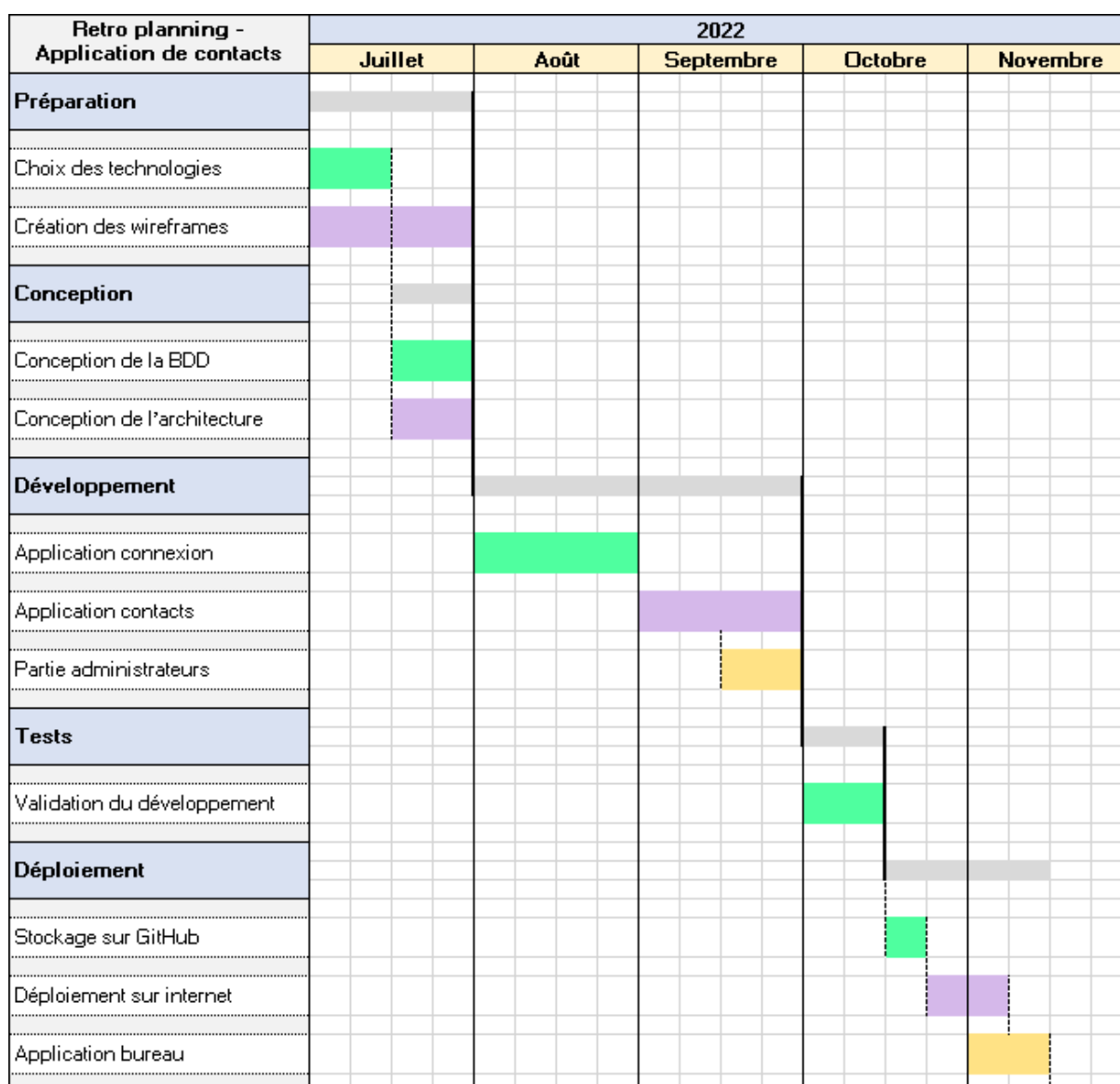
Cahier des charges

Voici le cahier des charges pour réaliser l'application web de gestion des contacts. Les colonnes de gauche à droite représentent respectivement : la fonction auquel doit répondre l'application, les critères à respecter dans la réalisation de cette fonction, les conditions ou valeurs à respecter pour respecter le(s) critère(s) indiqué(s), et la tolérance envisagée concernant le respect de ce(s) critère(s).

Fonctions	Critères	Niveau	Tolérance
FP1 : Afficher les contacts	Les contacts sont tous visibles	Prénom, nom, téléphone et e-mail	C'est le minimum
FP2 : Afficher les réseaux sociaux et les fêtes	Avoir le nom du réseau social	-	-
FP3 : Gérer les contacts	Gestion simple et facile	-	-
FP4 : Gérer les réseaux sociaux et les fêtes	Gestion simple et facile	-	-
FP5 : Gérer les événements	Un moyen simple et facile de gérer les événements	-	-
<i>FC1</i> : Les informations personnelles doivent être sécurisées	Respecter les recommandations de la CNIL	Tous les critères de la CNIL doivent être respectés	La sécurité des informations doit être fiable
<i>FC2</i> : Le site web doit être sécurisé	Respecter les recommandations de la CNIL	Tous les critères de la CNIL doivent être respectés	Le site doit être difficilement piratable
<i>FC3</i> : L'application est responsive	S'adapte à la taille de l'écran	Ordinateurs et tablettes	L'adaptation aux écrans de téléphone n'est pas nécessaire
<i>FC4</i> : L'application est esthétique	L'aspect doit plaire aux employés et faire sérieux	L'aspect est validé par les employés et les managers consultés	100 % des managers consultés, 90% des employés consultés
<i>FC5</i> : Avoir la partie administrateur	Pouvoir gérer la base de données	Gérer tous les modèles	Gestion simple et de qualité

Planning Gantt

L'application est développée selon une méthode de TDD (Test-Driven Development) : des tests sont effectués tout le long du développement et du déploiement.



Environnement technique

Système d'exploitation : Windows 11

Environnement de développement (IDE) : VS Code

Langage de programmation : python 3.9.13

Framework : Django 4.0.6

Base de données : SQLite 3.37.2 géré par l'ORM de Django

Plateforme de déploiement : Heroku

Contenu de requirementx.txt définissant les modules nécessaires au projet et leur version :

```
asgiref==3.5.2
dj-database-url==1.0.0
Django==4.1.2
django-heroku==0.3.1
gunicorn==20.1.0
psycopg2==2.9.4
sqlparse==0.4.3
whitenoise==6.2.0

django-cleanup==6.0.0
```

```
django-dbconn-retry==0.1.7
django-enviro==0.9.0
django-heroku==0.3.1
django-jquery==3.1.0
django-phonenumber-field==7.0.0
django-registration==3.3
django-smtplib==1.0

Pillow==9.0.0
```

Objectifs qualité

L'application doit être simple et facile d'utilisation.

L'application doit être esthétique et adaptée aux handicaps visuels.

Elle doit être utilisable sur internet avec la gestion d'un compte.

Elle doit être téléchargeable en tant qu'application bureau.

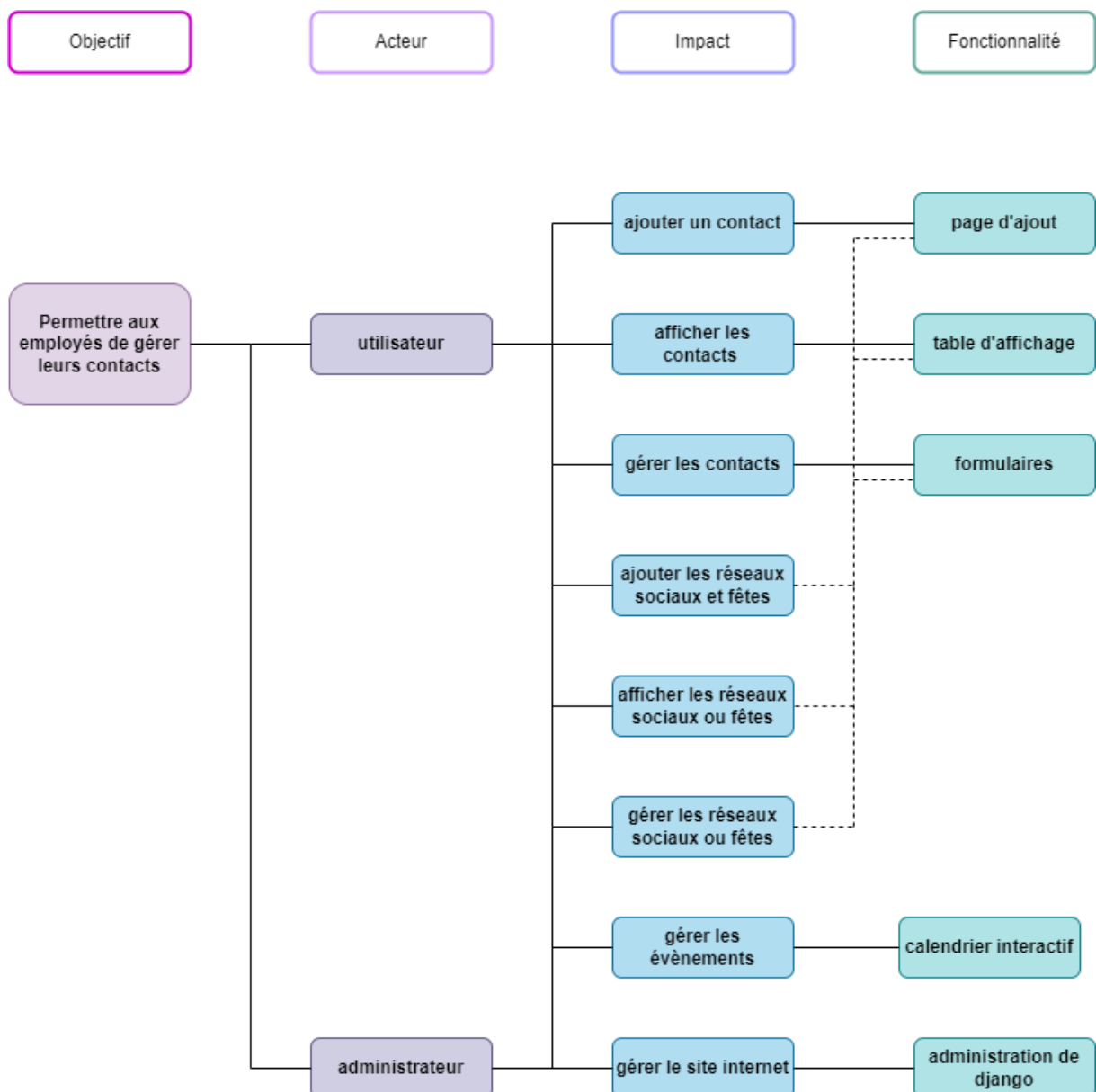
Il est probable que certaines fonctionnalités ne puissent pas être développées.

Le tout doit être livré sans dysfonctionnement.

Spécifications fonctionnelles du projet

Impact mapping

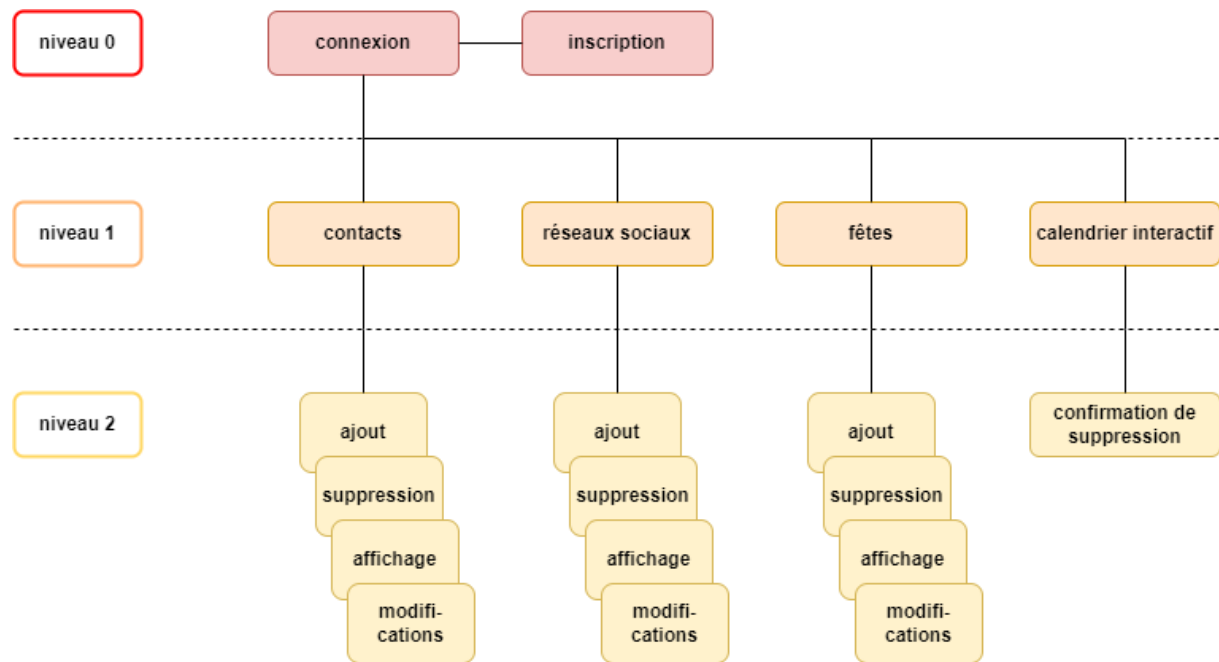
Je n'ai pas mis dans les acteurs le nouvel utilisateur, car cela implique une inscription, ce qui ne remplit pas l'objectif de la gestion des contacts.



Arborescence du projet

Les parties administrateur et de réinitialisation du mot de passe ne sont pas représentées car elles sont gérées par Django.

Un compte utilisateur est nécessaire pour gérer la séparation et la confidentialité des données.



Wireframes du projet

Cette partie présente les templates planifiés au début du projet. La partie administrateur et les pages de réinitialisation du mot de passe possèdent les templates par défaut de Django.

Login page

user name :

password :

[subscribe](#)

[reset password](#)

Sign up page

user name :

First name:

Last name :

E-mail :

password :

password confirmation :

Contacts, networks and parties pages

contacts	networks	parties	calendar
----------	----------	---------	----------

nom	e-mail	telephone n°1	telephone n°2	boutons
martin dupont	martin.matin@hotmail.fr	06 18 15 00 00	02 00 00 06 60	see edit del
pierre leblanc	leblanc06@gmail.com	06 20 20 80 95	04 60 70 00 10	see edit del
helene dubois	helene3@live.com	07 45 00 00 59	02 36 95 69 00	see edit del
marie leblanc	marie.leblanc@gmail.com	07 10 90 00 09	04 36 18 18 10	see edit del
jeanne dupont	jeanne20@hotmail.fr	06 18 81 02 02	02 70 12 70 12	see edit del

Calendar page

contacts	networks	parties	calendar	
month 2022				
1	2	3	4	5
6	7 anne birthday	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24 CE 17h	25
26 canteen closed	27	28	29	30
31	1	2	3	4

Add form page

contacts	networks	parties	calendar
----------	----------	---------	----------

add

nom	actions
martin du	edit del
pierre lebl	edit del
helene du	edit del
marie lebl	edit del
jeanne du	edit del

First name:

Last name :

E-mail :

Téléphone n° 1 :

Téléphone n° 2 :

add

picture

picture

Delete form page

contacts

networks

parties

calendar

add

Are you sure you want to delete luc leblanc ?

canceldelete

nom	e-m	2	boutons
martin dupont	mar	0	see edit del
pierre leblanc	lebl	0	see edit del
helene dubois	hele	0	see edit del
marie leblanc	mar	0	see edit del
jeanne dupont	jeanne20@hotmail.fr	06 18 81 02 02	02 70 12 70 12 see edit del

<

1

2

3

>

Details form page

contacts

networks

parties


calendar

add

First name :
Last name :
E-mail :
Téléphone n° 1 :
Téléphone n° 2 :

Timothée
Leblanc
timothée.leblanc@live.com
Téléphone1
Téléphone2

x



nom	boutons
martin dupont	see edit del
pierre leblanc	see edit del
helene dubois	see edit del
marie leblanc	see edit del
jeanne dupont	see edit del

<

1

2

3

>

Update form page

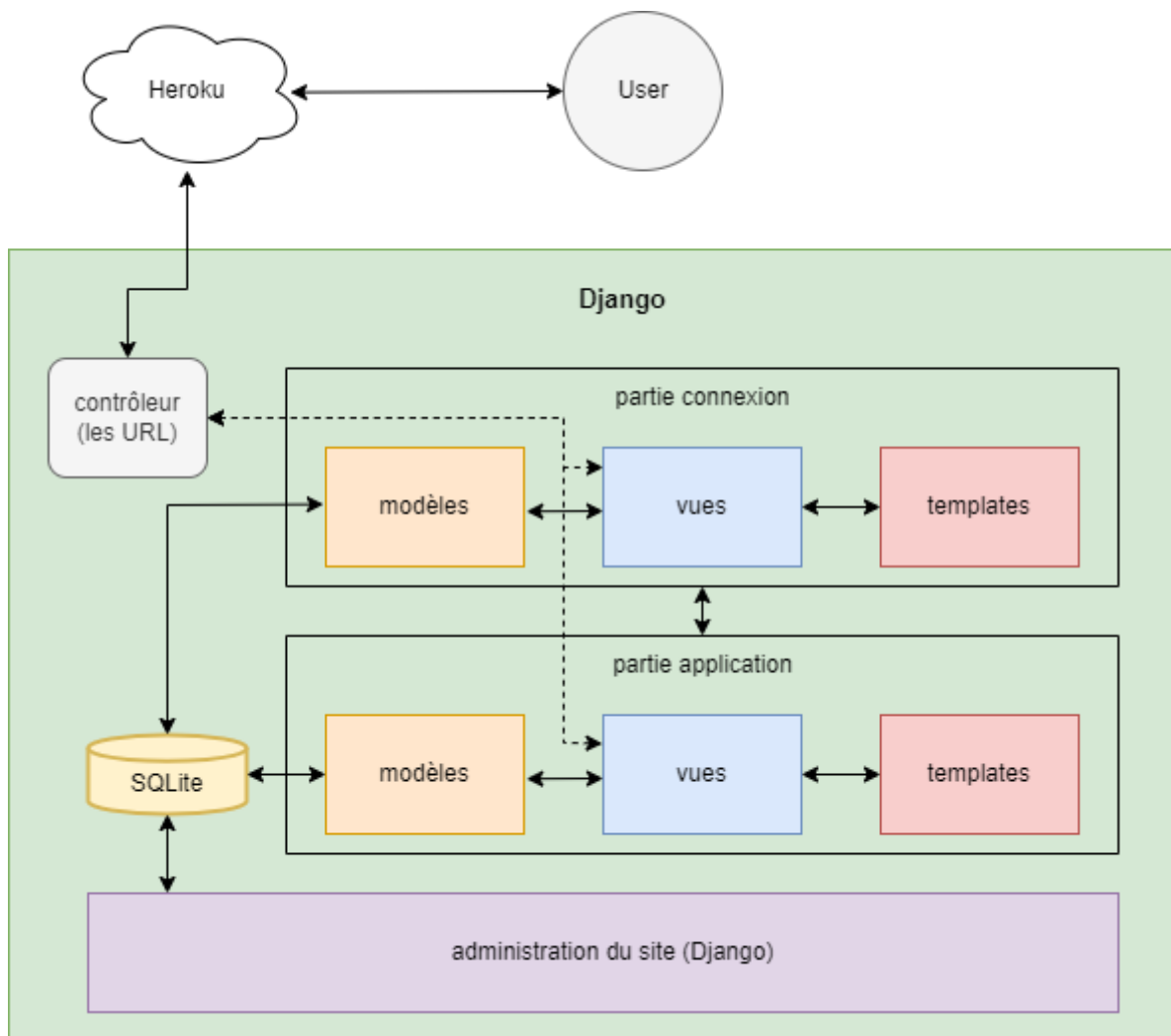
The screenshot shows a web application interface for managing contacts. The top navigation bar has four tabs: 'contacts' (highlighted), 'networks', 'parties', and 'calendar'. On the left side, there is a table of contacts with a search bar and an 'add' button. The table has a header 'nom' and lists several contacts. A modal form is open in the center for updating a contact. The form fields are: First name (Timothée), Last name (Leblanc), E-mail (timothée.leblanc@live.com), Téléphone n° 1 (06 35 65 00 10), and Téléphone n° 2 (02 69 69 20 10). There is a 'picture' button and an 'update' button. On the right side, there is a table of buttons for each contact, with 'edit' and 'del' options. At the bottom, there are pagination controls with arrows and numbers 1, 2, 3.

Divergences par rapport aux templates planifiés au début du projet :

- Les tableaux des modèles : Abandon de la pagination pour une **barre de défilement** et ajout d'une **barre de recherche** pour filtrer les données.
- Les pages de formulaires : Abandon des pages en pop-up pour des **pages entières**.
- Page du calendrier : **Je n'ai pas réalisé le calendrier** car je devais réaliser les dossiers du projet.
- Ensemble des pages : Ajout d'un **en tête** « Contacts Booklet » et amélioration de la **mise en forme**.

Spécifications techniques

Schéma de l'application internet



Modèle Logique des Données (MLD)

Ce diagramme montre la modélisation de la BDD envisagée au début du projet. Des relations d'un à plusieurs ont été établies entre les contacts, les réseaux sociaux, les fêtes et le modèle User.

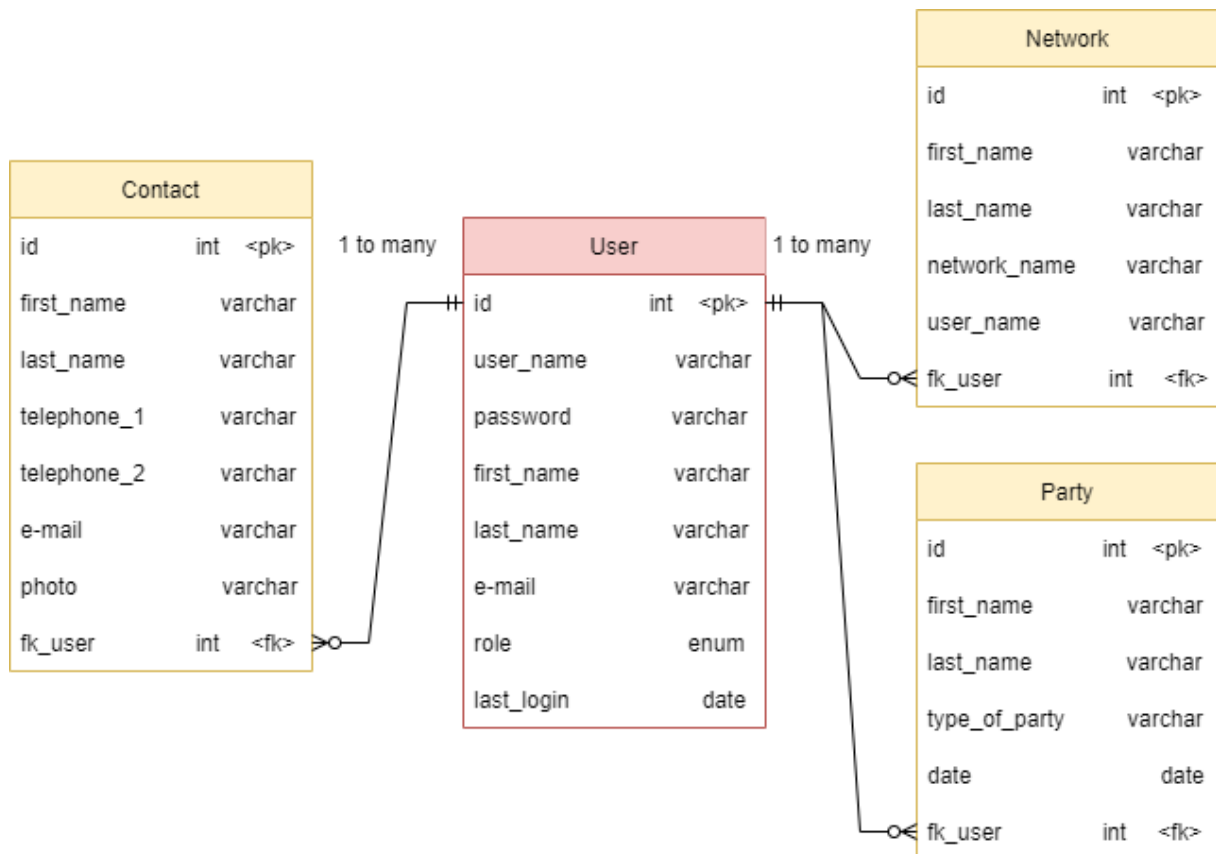
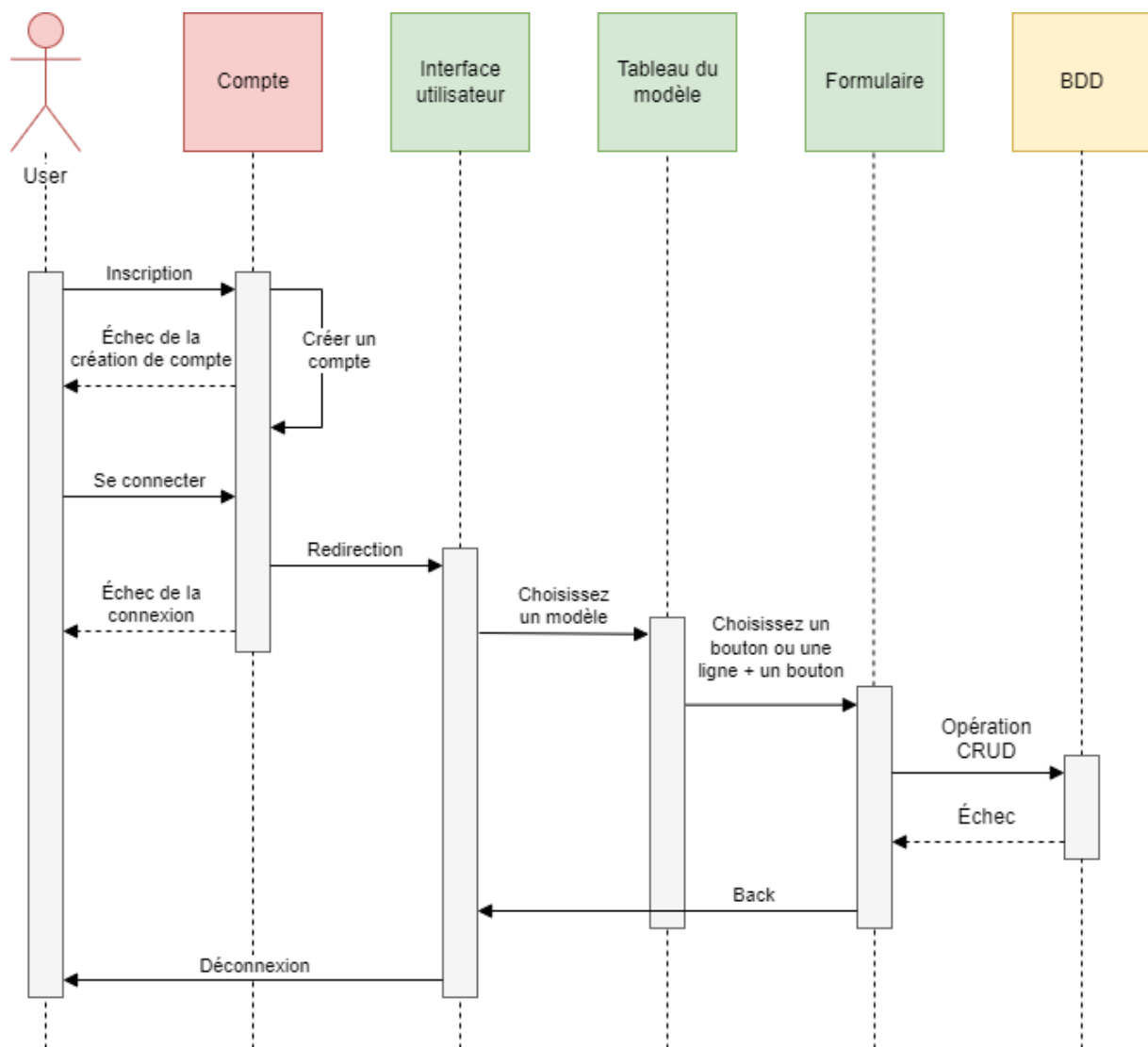


Diagramme de séquence

Après s'être connecté, l'utilisateur peut visualiser les éléments de la BDD dans des tables et faire des opérations CRUD au moyen de formulaires.

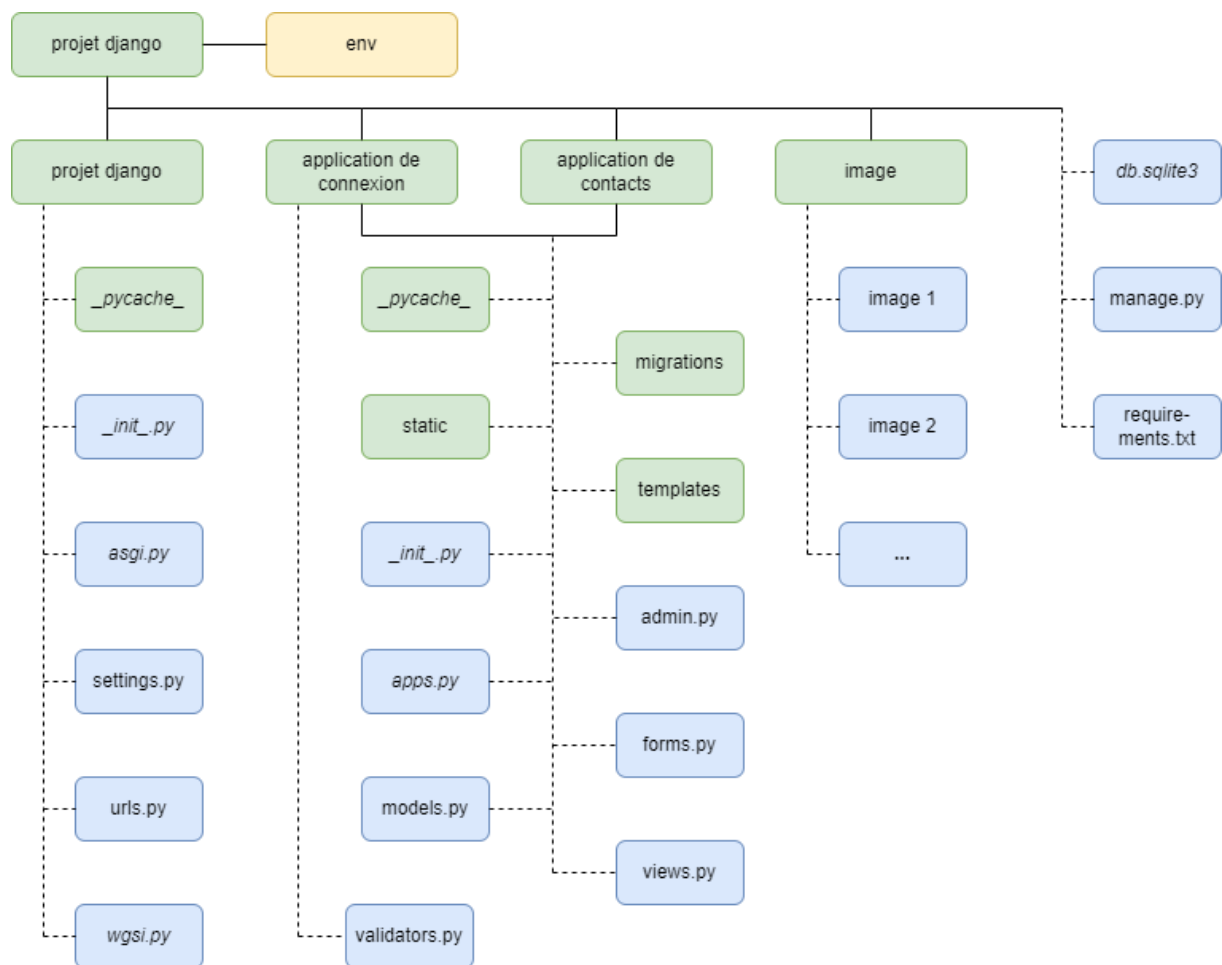
Les opérations CRUD (create, read, update, delete) sont les opérations de création, lecture, modification et suppression d'un élément d'une base de données.



Structures des fichiers du projet

Les répertoires sont de couleur verte (excepté pour l'environnement virtuel « env ») et les fichiers sont de couleur bleue.

Fichier	Rôle
env	Environnement virtuel pour le langage python
settings.py	La configuration du projet
urls.py	Les routes du projet
migrations	Répertoire les modifications des modèles et de la BDD
templates	Répertoire des templates html
admin.py	Indique les modèles gérés par l'administration de l'application web
forms.py	Les formulaires utilisés dans les templates html
models.py	Les différents modèles (liés à la BDD SQLite)
views.py	Fait le lien entre les modèles et les templates html
manage.py	Gère les commandes faites dans le terminal concernant le projet
requirements.txt	Définit les modules nécessaires au projet et leur version



Spécifications de sécurité

Authentification

Complexité du mot de passe :

Le mot de passe d'un utilisateur doit avoir au moins 8 caractères, dont un non-numérique. L'utilisateur peut réinitialiser son mot de passe au moyen d'un lien reçu par e-mail.

Réinitialisation du mot de passe :

Cet envoi se fait par une messagerie utilisant un protocole de sécurité (TLS par exemple) et protégé au moyen d'un code assurant un haut niveau de sécurité.

Autorisations

Droits d'accès :

Tout utilisateur peut accéder à l'application, à condition de d'avoir créé un compte et de s'être connecté. La partie administrateur est réservée aux « super-utilisateurs » qui sont les seuls à pouvoir créer des comptes de « super-utilisateur ». Les « super-utilisateurs » peuvent aisément supprimer un compte ou modifier ses droits d'accès.

Session utilisateur :

La session d'un utilisateur expire à la fermeture de son navigateur. Après la fermeture de son navigateur, il devra se reconnecter.

Protection

Injection SQL :

Django est automatiquement protégé contre les injections SQL qui consiste à rentrer du code SQL dans le navigateur (URL, champ de saisie, fichiers sources).

Cross Site Request Forgery (CSRF) :

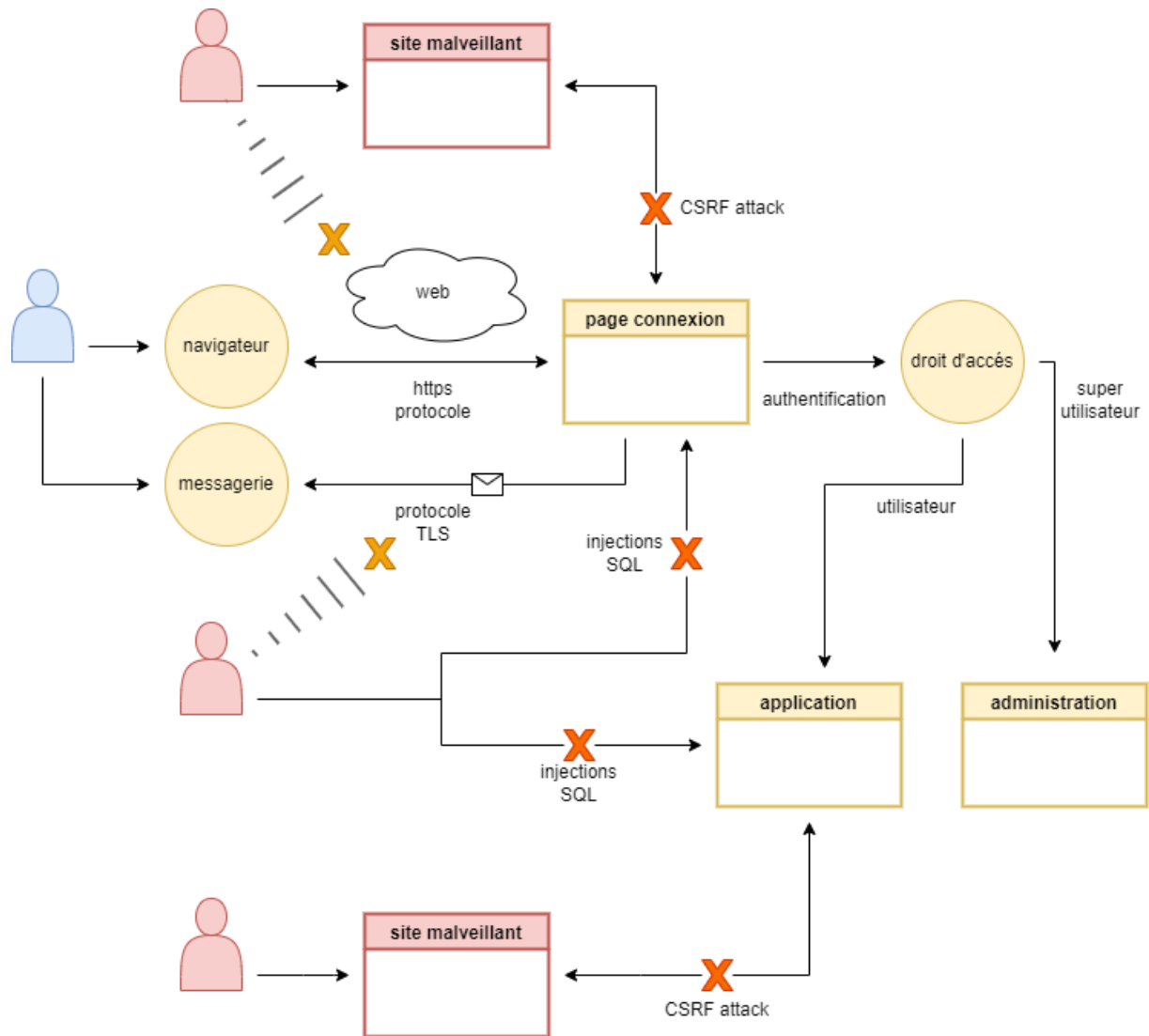
Les formulaires pour les opérations CRUD contiennent une balise « {% csrf_token %} » spécifique à Django qui protège l'application contre ce type de piratage.

Confidentialité

La protection contre l'analyse du trafic :

Le déploiement de l'application est prévu sur Heroku. La plateforme Heroku met en place le protocole « https » pour les sites hébergés, assurant le chiffrement des données entre le navigateur de l'internaute et le site web.

Schéma des spécifications de sécurité

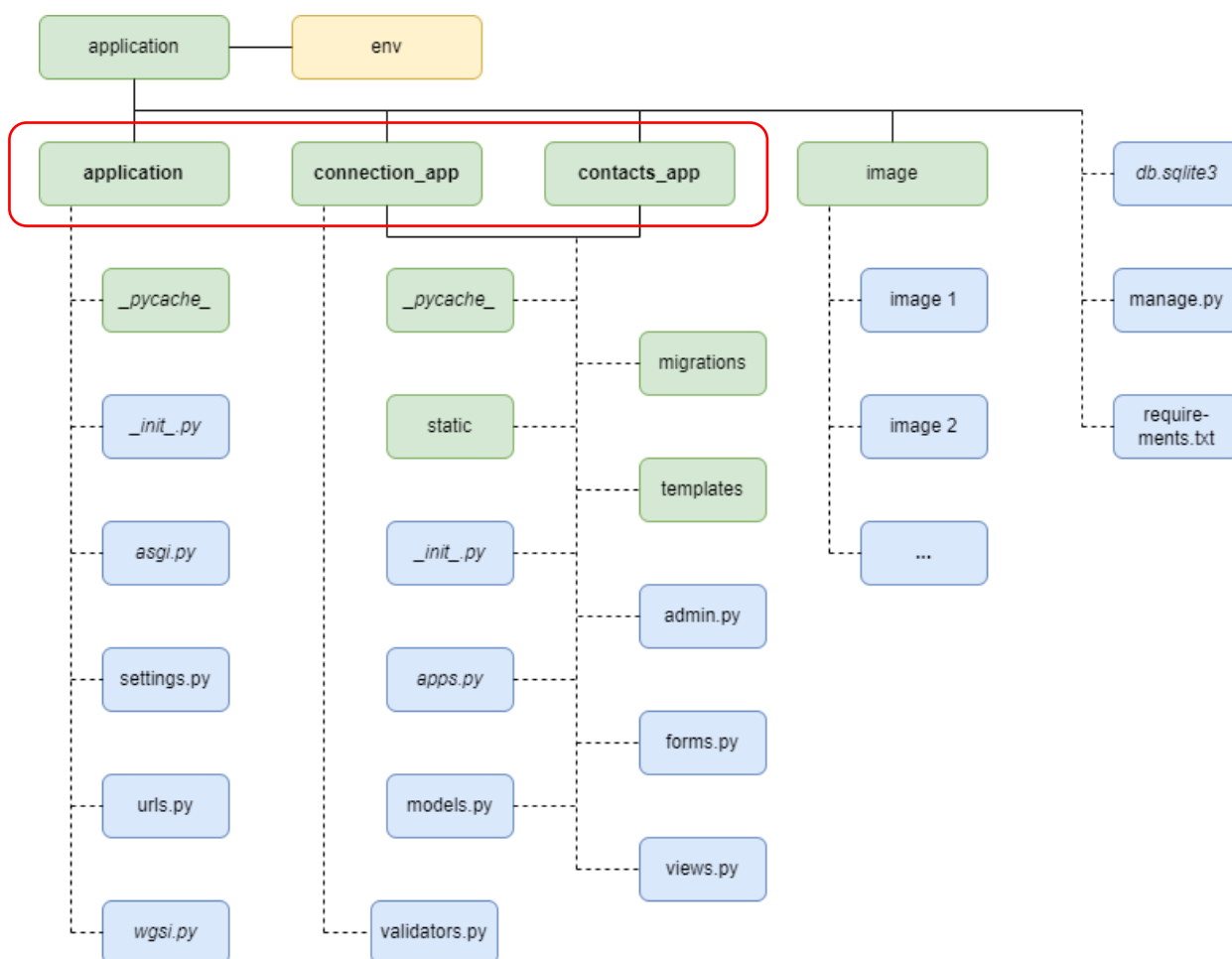


Réalisation de la partie commune de l'application

Les « (...) » surlignés en jaune symbolisent les parties du code non représentées dans ce dossier. Les extraits de code sont mis en couleur pour une meilleure lecture.

Structure générale

Le dossier du projet comporte trois grands répertoires : `application`, `connection_app`, et `contacts_app`. C'est le répertoire `application` qui est abordé dans cette partie. Il contient les fichiers communs aux parties connexion et contacts de l'application.



settings.py :

Le fichier `settings.py` contient la configuration du projet. La partie en vert possède la configuration pour la connexion et la déconnexion. La partie colorée en bleu contient celle pour l'envoi d'un email (réinitialiser le mot de passe).

La variable « `AUTH_PASSWORD_VALIDATORS` » définit une obligation de mot de passe d'au moins 8 caractères et avec un caractère non-numérique.

La ligne « `SESSION_EXPIRE_AT_BROWSER_CLOSE = True` » permet de fermer une session lors de la fermeture du navigateur.

L'envoi des e-mails est sécurisé par « `EMAIL_USE_TLS = True` ».

```
(...)  
AUTH_PASSWORD_VALIDATORS = [  
    {  
        'NAME': 'django.contrib.auth.password_validation.  
                MinimumLengthValidator',  
        'OPTIONS': {  
            'min_length': 8,  
        },  
    },  
    {  
        'NAME': 'django.contrib.auth.password_validation.  
                NumericPasswordValidator',  
    },  
]
```

```
(...)  
  
AUTH_USER_MODEL = 'connection_app.User'  
  
LOGIN_URL = 'login'  
  
LOGIN_REDIRECT_URL = '/contacts'  
LOGOUT_REDIRECT_URL = 'login'  
  
SESSION_EXPIRE_AT_BROWSER_CLOSE = True
```

```
(...)  
  
EMAIL_HOST = 'smtp.gmail.com'  
EMAIL_PORT = 587
```

```
(...)  
EMAIL_USE_TLS = True
```

```
(...)
```

urls.py :

Il définit l'ensemble des routes du projet et leur donne une référence.

```
(...)  
import connection_app.views  
import contacts_app.views  
  
from django.conf import settings  
from django.conf.urls.static import static  
  
urlpatterns = [  
    path('admin/', admin.site.urls),  
    path('', connection_app.views.login_page, name='login'),  
    (...)  
    path('reset_password/', auth_views.PasswordResetView.as_view(),  
        name="reset_password"),  
    (...)  
    path('contacts/', contacts_app.views.contacts, name="contacts"),  
    path('contacts/add/', contacts_app.views.addContact, name="addContact"),  
    (...)  
]  
(...)
```


Réalisation de la partie connexion de l'application

La partie connexion est définie dans le répertoire `connection_app` du projet. La page de connexion est la première page à laquelle accède l'utilisateur. Les liens « [Sign-up now !](#) » et « [Reset the password !](#) » permettent d'accéder à la page de création de compte et aux pages de réinitialisation de mot de passe. La page de connexion est rendue par le fichier `login_page.html`.

Contacts Booklet

User name:

Password:

Log in

Not a member ? [Sign-up now !](#)

forgotten password ? [Reset the password !](#)

models.py :

`models.py` permet de créer le modèle `User` correspondant au compte de l'utilisateur. Ce modèle hérite de `AbstractUser` qui possède déjà un attribut `username`, `password` et `email`. Django possède un modèle utilisateur par défaut, mais dans notre cas le modèle `User` sera défini comme modèle utilisateur dans `settings.py`.

```
from django.contrib.auth.models import AbstractUser
from django.db import models
from datetime import datetime

class User(AbstractUser):
    (...)

    first_name = models.CharField(max_length=254)
    last_name = models.CharField(max_length=254)

    profile_photo = models.ImageField(verbose_name='Photo de profil',
    upload_to='image/', blank=True)

    role = models.CharField(max_length=30, choices=ROLE_CHOICES,
    verbose_name='Rôle', default='Abonné')

    REQUIRED_FIELDS = ['first_name', 'last_name']

    (...)

```

forms.py :

forms.py permet de créer les formulaires qui seront utilisés pour les opérations CRUD, dont le formulaire LoginForm.

```
from django import forms
from django.contrib.auth import get_user_model
from django.contrib.auth.forms import UserCreationForm

class LoginForm(forms.Form):
    username = forms.CharField(max_length=63, label='User name', required=True)
    password = forms.CharField(max_length=63, widget=forms.PasswordInput,
                              label='Password', required=True)

( ... )
```

urls.py :

urls.py du répertoire application fait le lien entre views.py et les urls associés.

```
from django.contrib import admin
from django.contrib.auth import views as auth_views
from django.urls import path

import connection_app.views
import contacts_app.views

from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', connection_app.views.login_page, name='login'),
    path('logout', connection_app.views.logout_user, name='logout'),
    path('signup', connection_app.views.signup_page, name='signup'),
    path('home', connection_app.views.home, name='home'),

( ... )

]
```

views.py :

views.py fait le lien entre urls.py et les templates html. Il importe les objets de type Form de forms.py, dont LoginForm. Il n'a pas besoin d'importer le modèle User car celui-ci est défini comme modèle utilisateur dans settings.py.

```
from django.conf import settings

(...)

from . import forms

(...)

def login_page(request):
    form = forms.LoginForm()
    message = ''
    if request.method == 'POST':
        form = forms.LoginForm(request.POST)
        if form.is_valid():
            user = authenticate(
                username=form.cleaned_data['username'],
                password=form.cleaned_data['password'],
            )
            if user is not None:
                login(request, user)
                return redirect(settings.LOGIN_REDIRECT_URL)
            else:
                message = 'Identifiants invalides.'
    return render(request, 'connection_app/login_page.html',
        context={'form': form, 'message': message})

(...)
```

base.html :

Le contenu de base.html sera hérité par login_page.html.

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>base_connection_page</title>
    <link rel="stylesheet" href="{% static 'style.css' %}">
</head>
<body class = "nottranslate">
    <h1>Contacts Booklet</h1>
    {% block content %}{% endblock content %}
    {% if user.is_authenticated %}
        <p>Your are connected as {{ request.user }}.
        <a href="{% url 'logout' %}">Log out</a></p>
    {% endif %}
</body>
</html>
```

login_page.html :

login_page.html hérite de base.html au moyen des balises `{% extends 'base.html' %}` et `{% block content %}`. La balise `<form>` permet de créer un formulaire à partir de LoginForm. La balise `{% csrf_token %}` sert à protéger le formulaire contre les attaques de piratage CSRF.

```
{% extends 'base.html' %}
{% block content %}
    <p id="message">{{ message }}</p>
    <form method="post">
        {{ form.as_p }}
        {% csrf_token %}
        <button type="submit" >Log in</button>
    </form>
    <p>Not a member ? <a href="{% url 'signup' %}">Sign-up now !</a></p>
    <p>forgotten password ? <a href="{% url 'reset_password' %}">Reset the
password !</a></p>
{% endblock content %}
```

Réalisation de la partie contacts de l'application

La partie contacts se trouve dans le dossier `contacts_app` du projet. C'est dans cette partie que l'utilisateur visualise et gère ses contacts, avec les réseaux sociaux et les fêtes associés. La barre de navigation permet de choisir le modèle et les boutons (add, details, edit, delete) permettent la gestion des éléments du modèle.

La partie contacts est construite de la même manière que `connexion_app`, excepté que les modèles `Contact`, `Network` et `Party` sont importés dans `forms.py` et `views.py`. La partie calendrier n'a pas été réalisée afin de rédiger le dossier de projet et le dossier professionnel.

Contacts Booklet

menu: **contacts** networks parties calendar log out

add + number of contacts: 16

First Name	Last Name	E-mail	Telephone n°1	Management
Adrien	Fxxxx	fxxxxx@gmail.com	07 00 00 00 00	details edit delete
Auguste	Cxxxx	Cxxxxx@hotmail.com	06 00 00 00 00	details edit delete
Chaima	Bxxxx	bxxxxx@hotmail.fr	06 00 00 00 00	details edit delete
Céline	-	jxxxxx@hotmail.fr		details edit delete
David	Fxxxx	dxxxxx@live.com	06 00 00 00 00	details edit delete
Elodie	Sxxxx		07 00 00 00 00	details edit delete
François	Lxxxx	fxxxxx@gmail.com	07 00 00 00 00	details edit delete
Gérard	Rxxxx	gxxxxx@gmail.com	07 00 00 00 00	details edit delete
Harmonie	Txxxx	hxxxxx@live.com	07 00 00 00 00	details edit delete
Hilbert	Hxxxx	hxxxxx@live.com	07 00 00 00 00	details edit delete

forms.py :

```
from django import forms
from contacts_app.models import Contact, Network, Party

class addContactForm(forms.ModelForm):
    class Meta:
        model = Contact
        fields = ('first_name', 'last_name', 'email', 'telephone1',
                  'telephone2', 'profile_photo')
    (...)
```

views.py :

La partie contacts sera construite de la même manière que la partie connexion, excepté que les modèles seront importés dans forms.py et views.py. L'élément `@login_required` permet d'autoriser la page contacts uniquement aux utilisateurs connectés.

```
from django.shortcuts import render, redirect
from django.contrib.auth.decorators import login_required
from contacts_app.models import Contact, Network, Party

from .forms import addContactForm, editContactForm
from .forms import addNetworkForm, editNetworkForm
from .forms import addPartyForm, editPartyForm

# Create your views here
@login_required
def contacts(request):
    contacts = Contact.objects.filter(fk_user=request.user.id)
        .order_by('first_name', 'last_name')
    count = contacts.count()
    return render(request, 'contacts_app/contacts.html',
        context={'contacts' : contacts, 'count': count})

@login_required
def addContact(request):
    form = addContactForm()
    if request.method == "POST":
        form = addContactForm(request.POST, request.FILES)
        if form.is_valid():
            contact = form.save(commit=False)
            contact.fk_user = request.user
            contact.save()
            return redirect('contacts')
    return render(request, 'contactsForms/addContact.html',
        context={'form' : form})
```

(...)

contacts.html :

C'est le template pour la page des contacts.

```
{% extends 'base2.html' %}
{% load static %}

{% block css %}
    <link rel="stylesheet" href="{% static 'contacts.css' %}" />
{% endblock %}

{% block content %}
    (...)
    <table class="table-striped table-bordered" style="width:100%">
        <thead>
            <tr>
                <th>First Name</th>
                <th>Last Name</th>
            </tr>
        </thead>
        <tbody id="table">
            {% for line in contacts %}
            <tr>
                <td>{{ line.first_name }}</td>
                <td>{{ line.last_name }}</td>
            </tr>
            {% endfor %}
        </tbody>
    </table>
    (...)
{% endblock content %}


{% block script %}
    <script src="{% static 'contacts.js' %}"></script>
{% endblock script %}
```

Formulaire pour ajouter les contacts :

Contacts Booklet

[back](#)

Add a contact



First name:

Last name:

E-mail:

Telephone n°1:

Telephone n°2:

Picture: Aucun...tionné

AddContact.html :

```
(...)  
  
{% block content %}  
(...)  
    <div class="row">  
        <div class="column">  
              
        </div>  
        <div class="column_form">  
            <form method="post" enctype="multipart/form-data">  
                {{ form.as_p }}  
                {% csrf_token %}  
                <button type="submit" >add the contact</button>  
            </form>  
        </div>  
        <div class="column">  
        </div>  
    </div>  
{% endblock content %}  
  
(...)
```


Gestion de la persistance des données

La persistance des données sous Django :

Les requêtes SQL avec la BDD SQLite sont gérées par l'ORM (Object Relational-Mapping) de Django. Elles sont utilisées lors des migrations (création des tables) et lors des opérations CRUD.

SQLite est un programme contenant un SGBDR (Système de Gestion de Base de Données Relationnelles) et une BDD intégrée directement dans le programme qui l'utilise, dans notre cas Django.

Liste des programmes utilisés par Django pour gérer les données :

- ORM : opération CRUD
- SQLite : SGBD + BDD

Exemple de requête faite par l'ORM de Django :

Ci-dessous une requête qui sélectionne dans la table `contacts_app_network` les contacts possédés par l'utilisateur possédant l'identifiant « 2 ». L'ensemble des attributs indiqués (`id`, `first_name`, `last_name`, `network_name`, `user_name`, `fk_user_id`) sont sélectionnés.

```
SELECT
"contacts_app_network"."id", "contacts_app_network"."first_name",
"contacts_app_network"."last_name",
"contacts_app_network"."network_name", "contacts_app_network"."user_name",
"contacts_app_network"."fk_user_id"
FROM "contacts_app_network"
WHERE "contacts_app_network"."fk_user_id" = 2
```

Ci-dessous les instructions pour renvoyer le code SQL.

```
from connection_app.models import User
from contacts_app.models import Network
from django.db import connection
print(Network.objects.filter(fk_user_id=2).query)
```

Les `querysets` sont les requêtes lancées par Django lors d'une opération CRUD. Ces objets possèdent une méthode `query` qui renvoie le code SQL de la requête.

Présentation de la fonctionnalité à tester

Cette partie consiste à présenter la fonctionnalité du projet jugée la plus représentative dans le but de la soumettre à un jeu d'essai (tests). Cette partie abordera la fonctionnalité d'ajout d'un contact par un utilisateur. Sur la page des contacts, l'utilisateur clique sur le bouton d'ajout.

Contacts Booklet

menu: **contacts** networks parties calendar log out

add + number of contacts: 16


First Name	Last Name	E-mail	Telephone n°1	Management
Adrien	Fxxxx	fxxxxx@gmail.com	07 00 00 00 00	details edit delete
Auguste	Cxxxx	Cxxxxx@hotmail.com	06 00 00 00 00	details edit delete
Chalma	Bxxxx	bxxxxx@hotmail.fr	06 00 00 00 00	details edit delete
Céline	-	jxxxxx@hotmail.fr		details edit delete
David	Fxxxx	dxxxxx@live.com	06 00 00 00 00	details edit delete
Elodie	Sxxxx		07 00 00 00 00	details edit delete
François	Lxxxx	fxxxxx@gmail.com	07 00 00 00 00	details edit delete
Gérard	Rxxxx	gxxxxx@gmail.com	07 00 00 00 00	details edit delete
Harmonie	Txxxx	hxxxxx@live.com	07 00 00 00 00	details edit delete
Hibert	Hxxxx	hxxxxx@live.com	07 00 00 00 00	details edit delete

Puis l'utilisateur remplit le formulaire et ajoute éventuellement une photo de la personne, ici son cher collègue « Jean-Stéphane ». Après cela, il clique sur le bouton « add the contact ».

Contacts Booklet

[back](#)

Add a contact



First name:

Last name:

E-mail:

Telephone n°1:

Telephone n°2:

Picture:

hacker.jpg

add the contact

L'utilisateur peut voir son ajout dans le tableau du modèle.

Contacts Booklet

menu:

contacts

networks

parties

calendar

log out

add +

number of contacts: 17

First Name	Last Name	E-mail	Telephone n°1	Management		
Gérard	Rxxxxx	gxxxxx@gmail.com	07 00 00 00 00	details	edit	delete
Harmonie	Txxxxx	hxxxxx@live.com	07 00 00 00 00	details	edit	delete
Hibert	Hxxxxx	hxxxxx@live.com	07 00 00 00 00	details	edit	delete
Jean-stéphane	Lapon	jean.lapon.64@live.com	06 37 26 26 36	details	edit	delete
Lorela	Lxxxxx	lxxxxx@live.com		details	edit	delete
Luc	Hxxxxx	lxxxxx@hotmail.fr	06 00 00 00 00	details	edit	delete
Marion	Nxxxxx	mxxxxx@live.com	06 00 00 00 00	details	edit	delete
Martin	Dxxxxx	mxxxxx@gmail.com	07 00 00 00 00	details	edit	delete
Patrick	-	pxxxxx@live.com		details	edit	delete
Raphael	Bxxxxx	bxxxxx@live.com	07 00 00 00 00	details	edit	delete

Jeu d'essai et résultats

Cette partie en huit pages consiste à présenter un jeu d'essai (tests) sur la fonctionnalité d'ajout de contact (présentée dans la partie précédente), afin de s'assurer de son bon fonctionnement. Le jeu d'essai présenté consiste en une série de tests fonctionnels effectués manuellement. L'ensemble des tests sera effectué.

Date : 09/11/2022

Nom du testeur : Adrien RIVIERE, Développeur full stack

Test	Description	Etape	Résultat attendu	Résultat réel	Validité du test	Commentaires
001	Fonctionnement du bouton « add »	Cliquer sur le bouton « add ».	Le formulaire d'ajout de contact s'affiche	Le formulaire d'ajout de contact s'affiche	OK	
002	Fonctionnement du bouton « back »	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Cliquer sur le bouton « back ».	La page contacts s'affiche	La page contacts s'affiche	OK	
003	La page du formulaire est sécurisée	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Vérifier que l'url ne donne pas d'informations utilisateur	L'url ne donne pas l'identifiant utilisateur	http://127.0.0.1:8000/contacts/add/	OK	
		Vérifier sur Heroku que l'url est protégée contre l'analyse de trafics	L'url est en « https » sur Heroku	L'url est en « https » sur Heroku	OK	
		Vérifier que le formulaire n'est pas accessible aux utilisateurs non connectés	La page n'est pas directement accessible via son url	Lorsque l'on tente l'accès par l'url, pas de redirection	OK	http://127.0.0.1:8000/?next=/contacts/add/

Date : 09/11/2022

Nom du testeur : Adrien RIVIERE, Développeur full stack

Test	Description	Etape	Résultat attendu	Résultat réel	Validité du test	Commentaires
004	Le formulaire limite le nombre de caractères saisis	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Champ First name : « first name more than thirty characters »	On ne peut saisir que « first name more than thirty c »	« first name more than thirty c »	OK	
		Champ Last name : « last name more than thirty characters »	On ne peut saisir que « last name more than thirty ch »	« last name more than thirty ch »	OK	
		Champ Telephone n°1 « 123456789_123456789_12345789_123456789_ »	On ne peut saisir que « 123456789_123456789_12345789_ »	« 123456789_123456789_12345789_ »	OK	
		Champ Telephone n°2 « 123456789_123456789_12345789_123456789_ »	On ne peut saisir que « 123456789_123456789_12345789_ »	« 123456789_123456789_12345789_ »	OK	
		Champ E-mail « a.very.long.email.with@more.than.fifty.characters.fr »	On ne peut saisir que « a.very.long.email.with@more.than.fifty.characters.»	« a.very.long.email.with@more.than.fifty.characters. »	OK	

Date : 09/11/2022

Nom du testeur : Adrien RIVIERE, Développeur full stack

Test	Description	Etape	Résultat attendu	Résultat réel	Validité du test	Commentaires
005	Seul les champs « First name » et « Last name » sont requis	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Remplir le champ Last name avec « l_test »			OK	
		Cliquer sur le bouton « add the contact »	Une erreur s'affiche au niveau du champ « First name »	Un petit message : « Veuillez remplir ce champ »	OK	
		Effacer le contenu du champ « Last name »			OK	
		Remplir le champ First name avec « f_test »			OK	
		Cliquer sur le bouton « add the contact »	Une erreur s'affiche au niveau du champ « first name »	Un petit message : « Veuillez remplir ce champ »	OK	
		Remplir le champ Last name « l_test »			OK	
		Cliquer sur le bouton « add the contact »	Retour sur la page contacts	Retour sur la page contacts	OK	
		Vérifier que le nouveau contact s'affiche dans la table	Le contact est visible avec ses attributs	Le contact est visible avec ses attributs	OK	

Date : 09/11/2022

Nom du testeur : Adrien RIVIERE, Développeur full stack

Test	Description	Etape	Résultat attendu	Résultat réel	Validité du test	Commentaires
006	Le champ d'e-mail n'accepte que le format « xxxx@xx xx.xx »	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Remplir le champ First name avec « f_test »			OK	
		Remplir le champ Last name avec « l_test »			OK	
		Remplir le champ E-mail avec « test@mail »			OK	
		Cliquer sur le bouton « add the contact »	Une erreur s'affiche au niveau du champ email	Deux erreurs s'affichent : « Enter a valid email address »	NON	Deux messages d'erreurs mal positionnées avec une mauvaise mise en forme
		Effacer le contenu du champ E-mail			OK	
		Remplir le champ E-mail avec « test.fr »			OK	
		Cliquer sur le bouton « add the contact »	Une erreur s'affiche au niveau du champ de saisie email	Un message demandant un « @ »	OK	Le message est satisfaisant
		Remplir le champ E-mail avec « test »			OK	
			Une erreur s'affiche au niveau du champ email	Un message demandant un « @ »	OK	Le message est satisfaisant

Date : 09/11/2022

Nom du testeur : Adrien RIVIERE, Développeur full stack

Test	Description	Etape	Résultat attendu	Résultat réel	Validité du test	Commentaires
007	Cas du test : L'utilisateur peut ajouter un contact	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Remplir le champ First name avec « f_test »			OK	
		Remplir le champ Last name avec « l_test »			OK	
		Remplir le champ E-mail avec test@mail.ex			OK	
		Remplir le champ Telephone 1 avec « 06 00 00 00 00 »			OK	
		Remplir le champ Telephone 2 avec « 02 00 00 00 00 »			OK	
		Cliquer sur le bouton de validation « add the contact »			OK	
		Vérifier que le nouveau contact s'affiche dans la table	Le contact est visible avec ses attributs	Le contact est visible avec ses attributs	OK	
		Vérifier dans l'administration les données du contact	Le contact est présent avec ses attributs	Le contact est présent avec ses attributs	OK	

Date : 09/11/2022

Nom du testeur : Adrien RIVIERE, Développeur full stack

Test	Description	Etape	Résultat attendu	Résultat réel	Validité du test	Commentaires
008	Bonne gestion de l'image par défaut	<i>Si nécessaire, exécuter le test 007</i>			OK	
		Vérifier dans le code que l'image par défaut n'est qu'en un seul exemplaire	Il n'y a qu'une image dont le nom contient « no-image »	Il n'y a qu'une image dont le nom contient « no-image »	OK	
009	Le champ Picture interdit l'import de fichier n'étant pas une image	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Tenter l'import d'une vidéo	Pas d'import	Pas accessible	OK	
		Tenter l'import d'une musique	Pas d'import	Pas accessible	OK	
		Tenter l'import d'un GIF	Pas d'import	Pas accessible	OK	
		Tenter l'import d'un fichier .docx	Pas d'import	Pas accessible	OK	
010	Le champ Picture permet l'import d'une image	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Tenter l'import de l'image « i_test1.png »	L'image est importée	Le titre est visible près de Picture	OK	
		Vérifier que l'image est présente sur l'écran	L'image est visible à gauche avec de bonnes dimensions	L'image est visible à gauche avec de bonnes dimensions	OK	

Date : 09/11/2022

Nom du testeur : Adrien RIVIERE, Développeur full stack

Test	Description	Etape	Résultat attendu	Résultat réel	Validité du test	Commentaires
011	L'image est correctement gérée pour un compte	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Mettre dans le champ first name « f_test »			OK	
		Mettre dans le champ last name « l_test »			OK	
		Cliquer sur le bouton de validation « add the contact »	La page contacts s'affiche	La page contacts s'affiche	OK	
		Vérifier dans l'administration du site que l'image est présente	Le nom de l'image est accessible et commence par « img_test1 »	Le nom de l'image est accessible et commence par « img_test1 »	OK	
		Vérifier dans le code qu'une nouvelle image est produite.	Deux images contiennent dans leur nom « img_test1 »	Deux images avec dans leur nom « img_test1 »	OK	
		Vérifier dans le code que les deux images ont des noms différents	La nouvelle image à un nom plus long	La nouvelle image a un nom avec une extension aléatoire	OK	

Date : 10/11/2022

Nom du testeur : Adrien RIVIERE, Développeur full stack

Test	Description	Etape	Résultat attendu	Résultat réel	Validité du test	Commentaires
012	L'image est correctement gérée entre deux comptes	<i>Si nécessaire, exécuter le test 001</i>			OK	
		Mettre dans le champ First name « f_test »			OK	
		Mettre dans le champ Last name « l_test »			OK	
		Cliquer sur le bouton « add the contact »	La page contacts s'affiche	La page contacts s'affiche	OK	
		Vérifier dans l'administration du site que l'image est présente	Le nom de l'image est accessible et commence par « img_test2 ».	Le nom de l'image est accessible et commence par « img_test2 »	OK	
		Vérifier dans le code qu'une nouvelle image s'est créée.	Deux images contiennent dans leur nom « img_test2 ».	Deux images avec dans leur nom « img_test2 »	OK	
		Vérifier dans le code que les deux images ont un nom différent	La nouvelle image à un nom plus long	La nouvelle image a un nom avec une extension aléatoire	OK	

Conclusion des tests :

Les tests sont très concluants. Seul le test 006 présente une difficulté au niveau de la 5^e étape. Le message d'erreur lors de la saisie de « test@mail » est en double, mal positionné et avec un mauvais format. L'idéal serait d'avoir un message d'erreur avec un format identique aux autres messages.

Veille sur les vulnérabilités de sécurité

Les problèmes courants

Source : Site internet de la CNIL

Url : <https://www.cnil.fr/fr/securite-des-sites-web-les-5-problemes-les-plus-souvent-constates>

Liste des problèmes courants identifiés par le CNIL :

- 1 : Une authentification par un mot de passe trop souple
- 2 : L'absence de règles d'authentification à un compte (url accessible sans connexion)
- 3 : Rendre un compte client accessible depuis une URL incrémentale
- 4 : L'absence de chiffrement des données (données utilisateurs non chiffrées)
- 5 : L'indexation des données dans un moteur de recherche (accès à un fichier)

Information	Exigence(s)	Mise en œuvre
1	Règles pour le mot de passe nécessaires	Mot de passe avec 8 caractères et au moins une lettre
2	L'application accessible qu'aux utilisateurs connectés	Utilisation du tag « @login_required » pour limiter l'accès aux pages aux utilisateurs connectés
3	A partir d'un compte, un utilisateur ne peut accéder pas aux données des autres comptes	Redirection vers la page contacts lorsque l'identifiant de l'url ne correspond pas à un contact du compte utilisé.
4	Aucune	Non
5	Non utile : Pas de fichier	-

Les risques informatiques gérés par Django

Source : Documentation Django

Url : <https://docs.djangoproject.com/fr/4.1/topics/security/>

Liste des risques gérés par Django :

- 1 : Cross site scripting (XSS) : injection de scripts
- 2 : Cross site request forgery (CSRF) : attaque par le compte d'un utilisateur
- 3 : Injection SQL : Injecter des requêtes SQL
- 4 : Détournement de clique : Contenir un site dans un autre
- 5 : Protocole SSL (Secure Sockets Layer) : Sécurisation des échanges dans un réseau informatique
- 6 : Protocole HTTPS : Protocole http sécurisé (SSL, TLS ...) contre l'écoute de trafic

Information	Exigence(s)	Mise en œuvre
1	Protection contre le XSS	Django utilise des gabarits protégeant contre la majorité des attaques XSS
2	Protection des formulaires contre le CSRF	Utilisation du tag « {% csrf_token %} »
3	Protection contre les injections SQL	Django est naturellement protégé contre les requêtes SQL
4	Empêcher le site d'être affiché dans un cadre	Définir « XFrameOptions » dans MIDDLEWARE de « settings.py » (présent par défaut)
5	Protéger l'envoi de l'email pour la réinitialisation du mot de passe	Mettre « EMAIL_USE_TLS = True » dans « settings.py »
6	Mettre le protocole « https »	Naturellement mis en place par Heroku

Les bonnes pratiques pour la sécurité des sites web

Source : Site internet de la CNIL

Url : <https://www.cnil.fr/fr/securite-securiser-les-sites-web>

Les précautions élémentaires identifiées par le CNIL :

- 1 : Utiliser le protocole TLS (ou SSL) : les protocoles « https » et « TLS » sont utilisés
- 2 : Utiliser le TLS pour toutes les pages d'authentification : le protocole « https » est utilisé
- 3 : Limiter les ports de communication entre le client et le serveur
- 4 : Limiter l'accès aux outils et interfaces d'administration aux seules personnes habilitées
- 5 : Recueillir le consentement de l'internaute pour les cookies
- 6 : Limiter le nombre de composants mis en œuvre

Information	Exigence(s)	Mise en œuvre
1	Non utile : les protocoles TLS pour l'envoi d'email et « https » pour les pages web sont utilisés	-
2	Non utile : le protocole « https » est utilisé	-
3	Non utile : peu de ports	-
4	L'administration est réservée aux administrateurs de l'application web	Des utilisateurs ayant accès à l'application seule, et des super utilisateurs ayant accès à l'application et à l'administration du site
5	Non utile : pas de cookies	-
6	Non utile : peu de composants dans l'application	-

Ce qu'il ne faut pas faire selon le CNIL :

- 1 : Faire transiter des données à caractère personnel dans une URL
- 2 : Utiliser des services non sécurisés (authentification en clair, flux en clair, etc.)
- 3 : Utiliser les serveurs comme des postes de travail
- 4 : Placer les bases de données sur un serveur directement accessible depuis Internet
- 5 : Utiliser des comptes utilisateurs génériques (c'est-à-dire partagés entre plusieurs utilisateurs).

Information	Exigence(s)	Mise en œuvre
1	Pas de données personnelles dans l'url	-
2	La messagerie doit utiliser un protocole sécurisé (SSL, TLS ...)	Mettre « EMAIL_USE_TLS = True » dans settings.py
3	Non utile : serveur Heroku	-
4	La BDD n'est pas accessible directement	La BDD est prise en charge par Heroku
5	Non utile : pas de comptes génériques	-

Danger(s) identifié(s) :

L'identifiant de l'élément du modèle est présent dans l'url des pages « details » et « edit ».

Dangers identifiés concernant l'application

Un point est à améliorer concernant la sécurité de l'application.

Problème	Description	Risque
2	L'identifiant de l'élément du modèle est présent dans l'url des pages « details » et « edit »	Faible

Situation de travail ayant nécessité une recherche

Cette partie aborde la création d'une table filtrante pour l'affichage des données d'un modèle (contacts, réseaux sociaux ou fêtes).

Ressource utilisée

Navigateur :	Chrome
Type de ressource :	https://www.w3schools.com/
Barre de recherche :	« w3school html table filter »
Nom du lien hypertexte :	« How To Create a Filter/Search Table - W3Schools »
Url de la page web :	https://www.w3schools.com/howto/howto_js_filter_table.asp

Contenu de la recherche

```
<input type="text" id="myInput" onkeyup="myFunction()" placeholder="Search for names.." title="Type in a name">
```

```
<script>
function myFunction() {
  var input, filter, table, tr, td, i, txtValue;
  input = document.getElementById("myInput");
  filter = input.value.toUpperCase();
  table = document.getElementById("myTable");
  tr = table.getElementsByTagName("tr");
  for (i = 0; i < tr.length; i++) {
    td = tr[i].getElementsByTagName("td")[0];
    if (td) {
      txtValue = td.textContent || td.innerText;
      if (txtValue.toUpperCase().indexOf(filter) > -1) {
        tr[i].style.display = "";
      } else {
        tr[i].style.display = "none";
      }
    }
  }
}
</script>
```


Mise en œuvre :

Un élément « input » est ajouté au-dessus de la table de la page html. Un script est appelé dans la page. Il est utilisé pour permettre la modification de la table selon le contenu de cet élément.

contacts.html

(...)

```
<input type="text" id="search" onkeyup="filter()" placeholder="Search for first names or last names.." title="Type in a name">
```

(...)

```
{% block script %}
    <script src="{% static 'contacts.js' %}"></script>
{% endblock script %}
```

contacts.js

(...)

```
function filter() {
    var input, filter, table, tr, td, i, txtValue;
    input = document.getElementById("search");
    filter = input.value.toUpperCase();
    table = document.getElementById("table");
    tr = table.getElementsByTagName("tr");
    for (i = 0; i < tr.length; i++) {
        td = tr[i].getElementsByTagName("td")[0];
        td2 = tr[i].getElementsByTagName("td")[1];
        if (td || td2) {
            txtValue = td.textContent || td.innerHTML;
            txtValue2 = td2.textContent || td2.innerHTML;
            if (txtValue.toUpperCase().indexOf(filter) > -1) {
                tr[i].style.display = "";
            }
            else if (txtValue2.toUpperCase().indexOf(filter) > -1){
                tr[i].style.display = "";
            }
            else {
                tr[i].style.display = "none";
            }
        }
    }
}
```

Liste des recherches et des sources

Cette partie est non exhaustive, les recherches et les sources sur le web étant très nombreuses.

1. Créer une page de connexion et d'inscription | *OpenClassrooms*
2. Mot de passe oublié | *Youtube*
3. Trier un tableau | *CSEStack.org*
4. Sélectionner une ligne et la mettre en couleur une fois sélectionnée | *Stack Overflow*
5. Faire une relation d'un à plusieurs | *Documentation officielle de Django*
6. Faire une table filtrante | *W3Schools*
7. Faire une table avec une barre de défilement verticale | *Stack Overflow*
8. Envoi d'un e-mail pour la réinitialisation des mots de passe | *Support Microsoft + SitePoint*
9. Définir la valeur d'un champ de saisie d'un formulaire Django | *Stack Overflow*
10. Empêcher la traduction du navigateur | *1formatik.com + Stack Overflow*
11. Problème de retour à la ligne pour les « spans » des pages « details » | *Prograide*
12. Mettre sur github | *Javatpoint*
13. Avoir le « return » d'une fonction sur plusieurs lignes | *Stack Overflow*
14. Déployer sur Heroku | *Youtube*

Annexes

Site internet : <https://contactsbooklet.herokuapp.com/>

Dépôt GitHub : https://github.com/adrien2riviere/contacts_application