

# A platform designed to address language disorders.

*The art of speaking well is the art of practicing well.*



Rechercher

# SOMMAIRE

CONTEXT .....	3
PERSONA AND TRELLO .....	4
MOCKUP & WIREFRAME .....	5
PHYSICAL DATA MODEL .....	6
ARCHITECTURE & PLAN DE TEST .....	7
DJANGO templates .....	8
App Profiles .....	9
unit and integration test .....	10
functional test .....	11
App Exercises template .....	12
views et template .....	13
unit and integration test .....	14
functional test .....	15
SHELL_PLUS .....	16
COVERAGE test to 93% .....	17
DATA MIGRATION .....	18
DEPLOYMENT .....	19
setting prod.py .....	19
heroku .....	20
deployment plateform .....	21

## Context

Many parents feel powerless when their children suffer from language disorders. It is imperative to address this issue and provide aid to parents and children alike. A platform that offers a personalised and fun course to improve communication skills can be a solution.

The platform should allow parents to practice with their children outside of sessions and help them improve their language skills on a daily basis. Moreover, speech and language therapists need an efficient tool to produce patient files and reports, and organize sessions.

# L'art de bien parler...

## C'est l'art de bien pratiquer

Apprendre en s'amusant pour lutter contre les troubles du langage avec l'aide d'un orthophoniste !

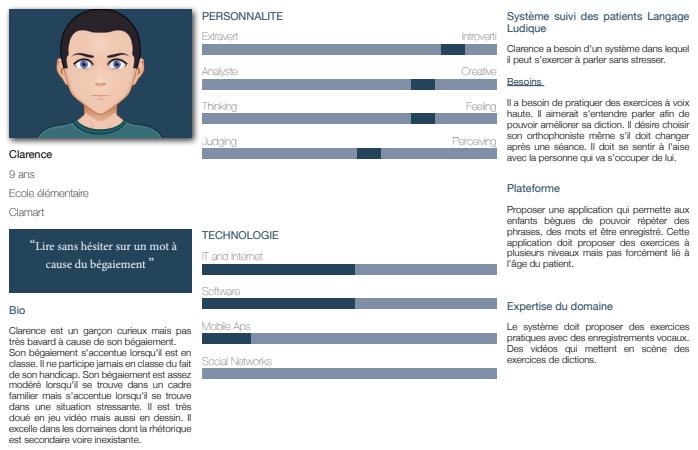
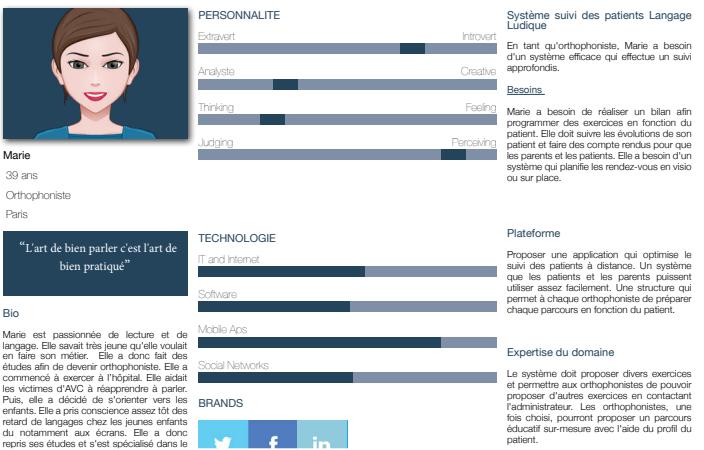
Où ?

# Persona et Trello

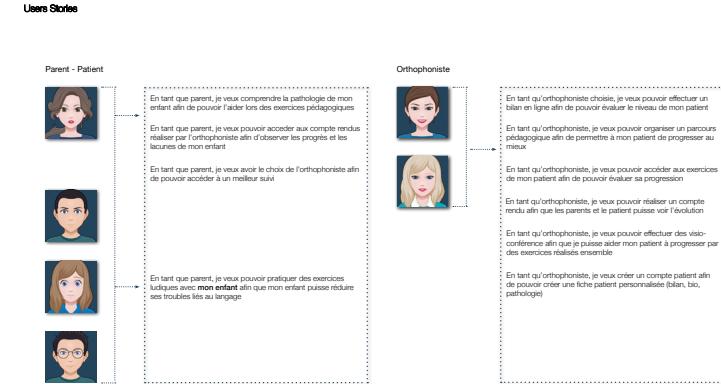
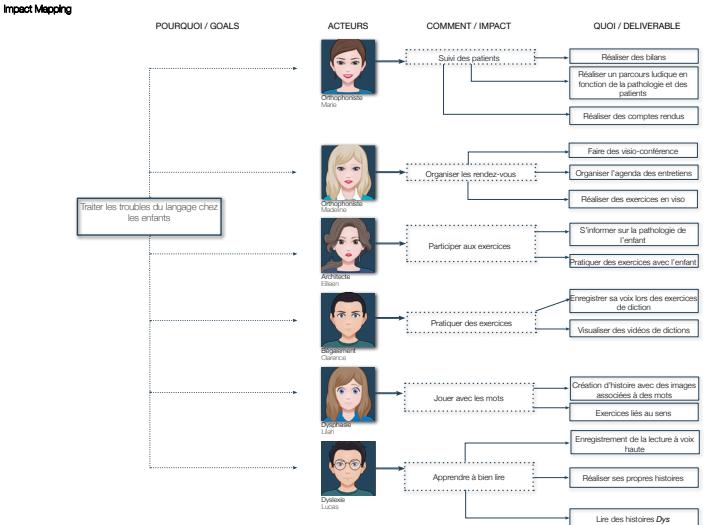
## Persona

Identify user profiles

- Creation of personas
  - Patient / Parents
  - Speech Therapist



## Impact Mapping et Users Stories



**Langage Ludique**

**Backlog**

- Définir les fonctionnalités
- Choix de l'orthophoniste
- Modèles + homepage
- modèles + sign in + speech therapist
- speech therapist index + patient's form (create patient forms)
- list of patient + tests
- Continuer avec la création des workshops
- Faire les corrections sur les tests
- Commencer la création des workshops
- Terminer la template de l'application (speech-therapist) commencez l'app exercices
- app exercices continuer
- creation ludic journey + finir partie profil patient
- Ajouter une carte

**Sujets de la prochaine session**

- Maquette + Diagramme de Classe
- Modèles + homepage
- modèles + sign in + speech therapist + patients
- speech\_therapist index + patient's form ( create patient forms)

**A faire**

- Ajouter une carte
- Ajouter une carte

**Langage Ludique**

**Backlog**

- Admin functionality
- Création du profil orthophoniste
- speech therapist functionality
- Ludic Journey par l'orthophoniste
- parent functionality
- Choix de l'orthophoniste
- speech therapist functionality
- patient functionality
- parent functionality
- Compte rendu
- speech therapist functionality
- Effectuer un bilan en ligne

**Sujets de la prochaine session**

- Définir les fonctionnalités en fonction des users stories
- Maquette + Diagramme de Classe
- Modèles + homepage
- modèles + sign in + speech therapist + patients
- speech\_therapist index + patient's form ( create patient forms)

**A faire**

- Ajouter une carte
- Ajouter une carte

## Trello

Backlog's files with les users stories

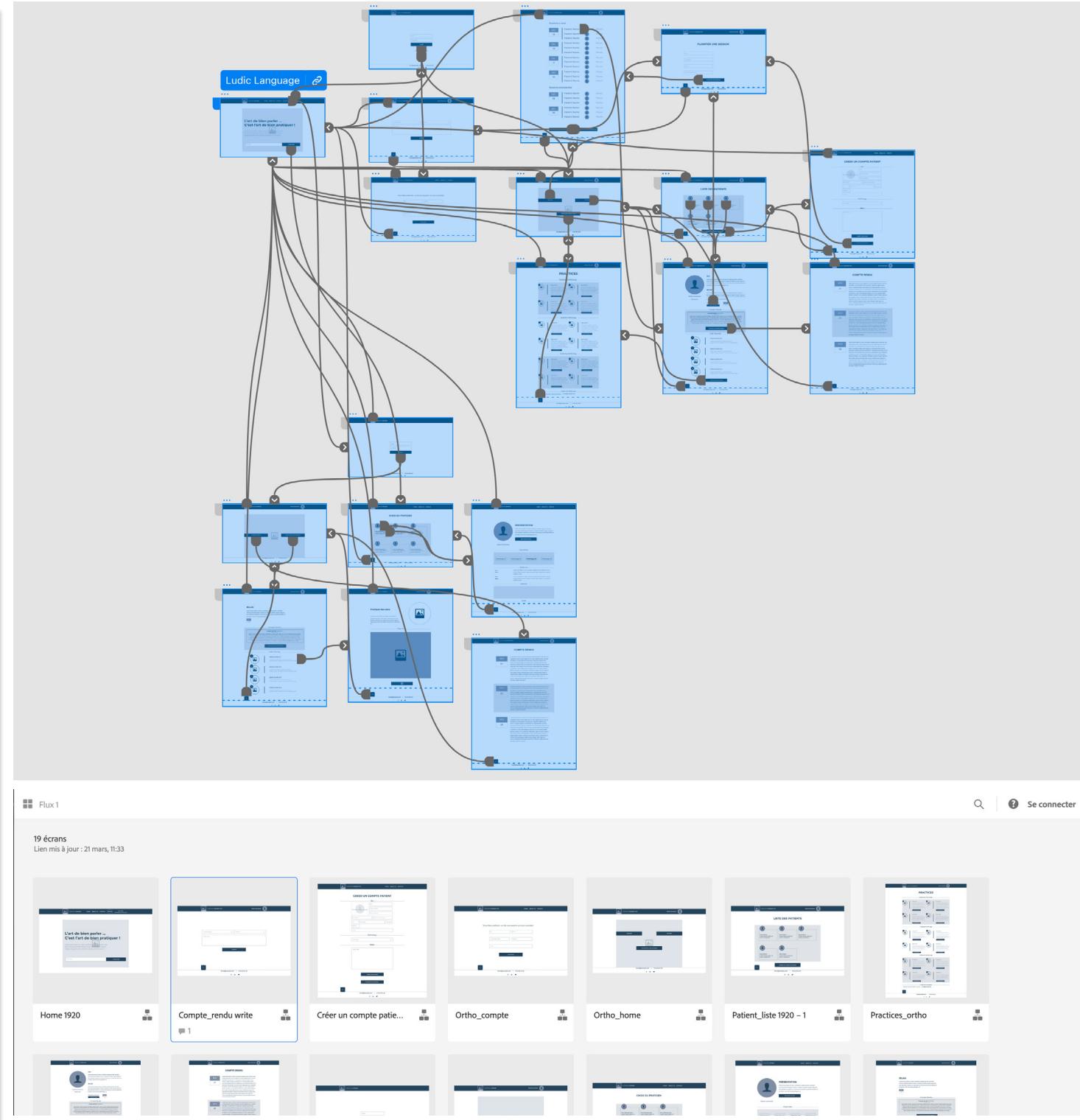
Ludic Language

# Mock-up & Wireframe

## XD Adobe

Creating a mock-up on paper to get ideas off the ground.

Creating a mock-up --> Wireframe on Adobe XD



# Physical Data Model

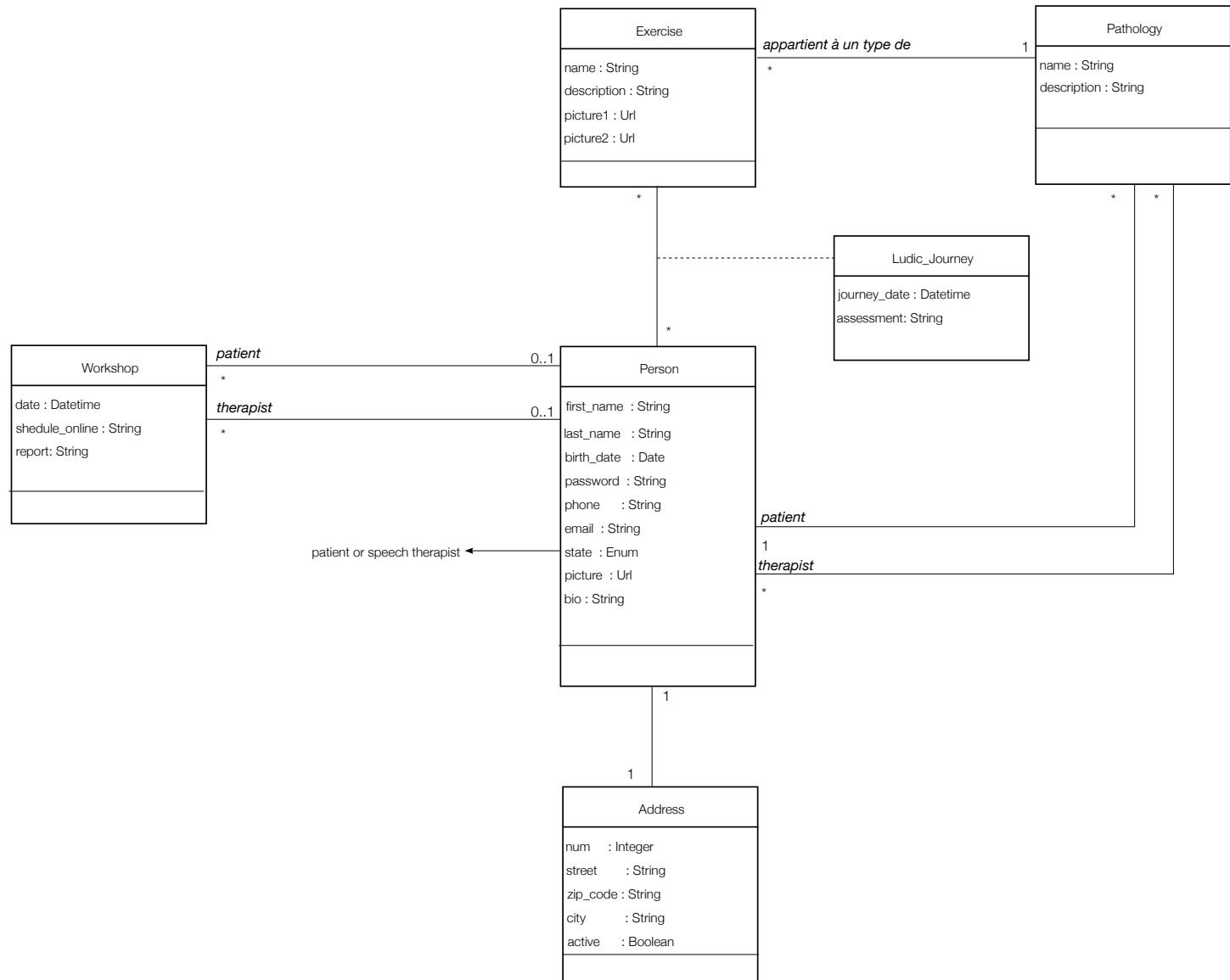
## physical data model

Producing the Model in InDesign

Postico

Viewing tables

	Name
	SQL Query
	auth_group
	auth_group_permissions
	auth_permission
	auth_user
	auth_user_groups
	auth_user_user_permissions
	django_admin_log
	django_content_type
	django_migrations
	django_session
	exercises_exercise
	exercises_ludicjourney
	exercises_pathology
	profiles_address
	profiles_profile
	workshops_workshop
>	pg_catalog
>	information_schema



## Architecture and test plan

### Architecture

Xtreme Programming

### Test plan

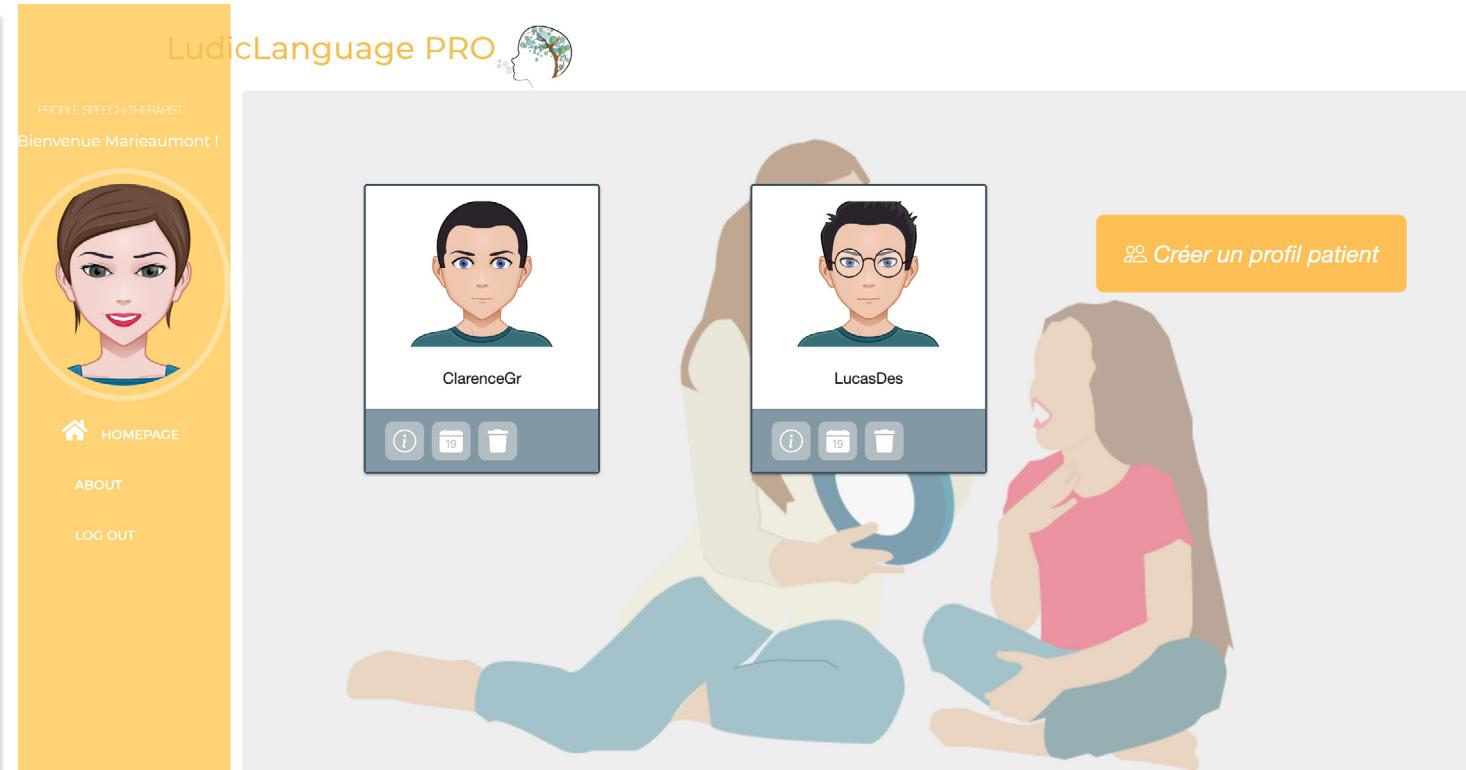
```
language_project/
  |- ludic_language/
    |- __init__.py
    |- asgi.py
    |- urls.py
    |- wsgi.py
    |- settings.py
    |- base/
      |- __init__.py
      |- admin.py
      |- views.py
      |- templates/
        |- index.html
        |- legal_notice.html
        |- aboutus.html
      |- statics/
        |- css/
          |- styles.css
        |- js/
          |- script.js
        |- assets/
          |- favicon.ico
          |- img/
    |- profiles/
      |- __init__.py
      |- admin.py
      |- views.py
      |- models.py
      |- forms.py
      |- templates/
        |- login.html
        |- index_patient.html
        |- index_speech.html
        |- patient_list.html
        |- speech_list.html
        |- form_patient.html
        |- patient_confirm_delete.html
      |- tests/
        |- test_views.py
        |- test_forms.py
        |- test_models.py
        |- tests_functional.py
    |- statics/
      |- profiles/
        |- css/
          |- styles.css
        |- js/
          |- script.js
        |- assets/
          |- img/
```

# DJANGO

## App Profiles

Patient form

- Templates



The screenshot shows a registration form for a new user. The title "LudicLanguage PRO" and brain icon are at the top. Below, it says "PROFILE SPEECH-THERAPIST Bienvenue Marieaumont !". The form fields include:

- First name :
- Last name :
- Email :
- Nom d'utilisateur :  (Requis. 150 caractères maximum. Uniquement des lettres, nombres et les caractères « @, ., \*, +, - » et « \_ ».)
- Mot de passe :

Below these fields are several bullet points regarding password requirements:

- Votre mot de passe ne peut pas trop ressembler à vos autres informations personnelles.
- Votre mot de passe doit contenir au minimum 8 caractères.
- Votre mot de passe ne peut pas être un mot de passe couramment utilisé.
- Votre mot de passe ne peut pas être entièrement numérique.

Further down, there are fields for "Confirmation du mot de passe" (Confirm password), "Birth date" (Birth date), and "Bio" (Bio), each with its own input field. The "Confirmation du mot de passe" field includes a note: "Saisissez le même mot de passe que précédemment, pour vérification." (Enter the same password as previously, for verification.)

## App Profiles

### Patient form

- Template html
- forms.py
- views.py

enctype=>multipart/form-data --> upload de données

This function retrieves the ID of the user who is currently logged in and then passes it as an argument in a specific form.

```
{% extends 'index_speech.html' %}  
{% block content%}  
<div class=>content-container>  
  <div class=>content-container>  
    <section class=>jumbotron>  
      <form method=>POST</form> enctype=>multipart/form-data>  
        {% csrf_token %}  
        {{ form.as_p }}  
        <button class=>submit>CREER UN PATIENT</button>  
      </form>  
    </section>  
  </div>  
</div>  
{% endblock content %}  
template --> form_patient.html
```

```
class PatientAddView(LoginRequiredMixin, CreateView):  
    form_class = UserProfileForm  
    model = User  
    template_name = 'form_patient.html'  
  
    def get_form_kwargs(self):  
        kwargs = super().get_form_kwargs()  
        kwargs['therapist'] = self.request.user.profile  
        return kwargs  
  
    def get_success_url(self):  
        return reverse('index_speech')
```

views.py --> CreateView

```
class UserProfileForm(UserCreationForm):  
    first_name = forms.CharField()  
    last_name = forms.CharField()  
    email = forms.EmailField(required=True)  
    birth_date = forms.DateField()  
    bio = forms.CharField(widget=forms.Textarea)  
    review = forms.CharField(widget=forms.Textarea)  
    profile_pic = forms.ImageField(required=False)  
    pathology = forms.ModelChoiceField(queryset=Pathology.objects.all())  
    num = forms.IntegerField()  
    street = forms.CharField()  
    zip_code = forms.CharField()  
    city = forms.CharField()  
  
    class Meta:  
        model = User  
        fields = ['first_name', 'last_name', 'email',  
                  'username', 'password1', 'password2']  
  
    def __init__(self, *args, therapist=None, **kwargs):  
        super().__init__(*args, **kwargs)  
        self.therapist = therapist  
  
    def save(self, *args, **kwargs):  
        user = super().save(*args, **kwargs)  
        address = Address.objects.create(  
            profile=user.profile, num=self.cleaned_data['num'], street=self.cleaned_data['street'],  
            zip_code=self.cleaned_data['zip_code'], city=self.cleaned_data['city'])  
        user.profile.birth_date = self.cleaned_data['birth_date']  
        user.profile.therapist = self.therapist  
        user.profile.bio = self.cleaned_data['bio']  
        user.profile.review = self.cleaned_data['review']  
        user.profile.profile_pic = self.cleaned_data['profile_pic']  
        user.profile.pathology = self.cleaned_data['pathology']  
        user.profile.save()  
        user.profile.address.save()  
        return user
```

forms.py -->UserCreationForm

## Integration and unit test

Testing patient form

```
class UserProfileFormTest(TestCase):
    def setUp(self):
        self.pathology_id = Pathology.objects.create(name='Dyslexie').pk
        self.therapist = User.objects.create_user(
            username='Marieaumont', password='Therapist@25').pk
        return super().setUp()

    def test_userprofile_valid(self):
        form = UserProfileForm(data={
            'first_name': 'Lucas',
            'last_name': 'Desmarais',
            'email': 'helenedesmarais@email.com',
            'username': 'LucasD',
            'password1': '12test12',
            'password2': '12test12',
            'birth_date': '2013-05-12',
            'bio': 'Lucas est atteint de .....',
            'review': 'Bilan du patient .....',
            'profile_pic': 'lucasdesmarais.png',
            'pathology': self.pathology_id,
            'num': 10,
            'street': 'rue de la paix',
            'zip_code': 75015,
            'city': 'Paris',
        }, therapist=self.therapist)
        self.assertTrue(form.is_valid())

    def test_userprofile_missing_first_name(self):
        form = UserProfileForm(data={
            'first_name': '',
            'last_name': 'Desmarais',
            'email': 'helenedesmarais@email.com',
            'username': 'LucasD',
            'password1': '12test12',
            'password2': '12test12',
            'birth_date': '2013-05-12',
            'bio': 'Lucas est atteint de .....',
            'profile_pic': 'lucasdesmarais.png',
            'pathology': self.pathology_id,
            'num': 10,
            'street': 'rue de la paix',
            'zip_code': 75015,
            'city': 'Paris',
        }, therapist=self.therapist)
        self.assertFalse(form.is_valid())
        self.assertIn('first_name', form.errors)
```

test\_forms.py

```
class UserProfileViewTest(TestCase):
    def setUp(self):
        self.user = get_user_model().objects.create_user(
            username='Marieaumont', password='Therapist@25').pk
        self.patient = User.objects.create_user(
            username='ClarenceBr', password='12test12',
            email='test@email.com')
        self.pathology_id = Pathology.objects.create(name='Dyslexie').pk
        self.form_url = reverse('form_patient')
        self.client = Client()
        self.patient = {
            'first_name': 'Clarence',
            'last_name': 'Brault',
            'username': 'ClarenceBr',
            'email': 'test@email.com',
            'password1': '12test12',
            'password2': '12test12',
            'birth_date': '12/05/2013',
            'bio': 'Clarence est atteint de .....',
            'profile_pic': '',
            'pathology': self.pathology_id,
            'num': '10',
            'street': 'rue de la paix',
            'zip_code': '75015',
            'city': 'Paris',
            'therapist': self.user
        }
        return super().setUp()

    def test_view_ok(self):
        login = self.client.login(
            username='Marieaumont', password='Therapist@25')
        self.assertTrue(login)
        response = self.client.get(self.form_url)
        self.assertEqual(response.status_code, 200)

    def test_form_ok(self):
        login = self.client.login(
            username='Marieaumont', password='Therapist@25')
        self.assertTrue(login)
        response = self.client.post(self.form_url, self.patient)
        self.assertEqual(response.status_code, 200)
        patient = User.objects.get(email='test@email.com')
        patient.pathology_id = self.pathology_id
        patient.first_name = 'Clarence'
```

test\_views.py

## Functional test

Testing patient form with Selenium

```
class LudicLanguageProfileTest(StaticLiveServerTestCase):

    def setUp(self):
        self.driver = webdriver.Firefox(options=firefox_options)
        self.user = get_user_model().objects.create_user(
            username='Marieaumont', first_name='Marie', password='12test12', email='test@email.com')
        self.user_id = self.user.pk
        self.user.save()
        self.therapist = Profile(user=self.user, birth_date='1975-05-12', state=2,
                                bio='Marie est spécialisée .....', profile_pic='marieaumont.png'
                                ).save()
        self.address = Address.objects.create(
            num=6, street='rue de ...', zip_code=92500, city='Paris', profile_id=self.user_id)
        self.patient = User.objects.create_user(
            username='LucasD', password='12test12', email='test@email.com')
        self.pathology_id = Pathology.objects.create(name='Dyslexie').pk
        Profile(user=self.patient, birth_date='2013-05-12', state=1,
                bio='Lucas est atteint de .....', profile_pic='lucasdesmarais.png',
                pathology_id=self.pathology_id, therapist_id=self.therapist).save()
        return super().setUp()

    def test_form(self):
        driver = self.driver
        driver.implicitly_wait(10)
        driver.get("%s%s" % (self.live_server_url, '/login/'))
        driver.find_element_by_id('id_username').send_keys('Marieaumont')
        driver.find_element_by_name('password').send_keys('12test12')
        driver.find_element_by_css_selector('.submit').click()
        driver.find_element_by_css_selector('.btn').click()
        driver.get("%s%s" % (self.live_server_url, '/form_patient/'))
        driver.find_element_by_name('first_name').send_keys('Clarence')
        driver.find_element_by_name('last_name').send_keys('Grey')
        driver.find_element_by_name('email').send_keys('helenegrey@email.com')
        driver.find_element_by_name('username').send_keys('ClarenceGr')
        driver.find_element_by_name('password1').send_keys('12test12')
        driver.find_element_by_name('password2').send_keys('12test12')
        driver.find_element_by_name('birth_date').send_keys('12/05/2015')
        driver.find_element_by_name('bio').send_keys('Clarence est un ....')
        driver.find_element_by_name('review').send_keys(
            'Clarence a besoin de ....')
        driver.find_element_by_name('pathology').send_keys('Dyslexie')
        driver.find_element_by_name('num').send_keys('10')
        driver.find_element_by_name('street').send_keys('rue de ....')
        driver.find_element_by_name('zip_code').send_keys(92140)
        driver.find_element_by_name('city').send_keys('Clamart')
        driver.find_element_by_css_selector('.submit').click()
        driver.find_element_by_name('patient').click()
```

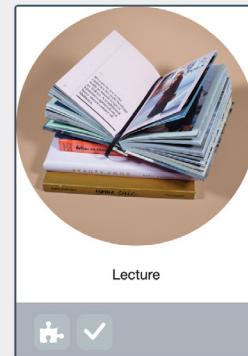
## App Exercises

List of exercises categorized by the disease

- Templates



## Dysphasie



## Dyslexie

## App Exercises

List of exercises categorized by the disease

- Template html
- views.py

```
{% for pathology in pathology_list %}
    <div class=>row<>
        <div class=> d-flex justify-content-center>
            <h3>{{pathology.name}}</h3>
        </div>
        <hr>
        {% for exercise in pathology.exercise_set.all %}
            <div class=>col-lg-4 col-md-6 col-sm-12 d-flex justify-
            'content-center'>
                <div class='card exercise-card d-flex flex-column'>
                    {% if exercise.picture1 %}
                        <img src=>{{exercise.picture1.url}}</img>
                    <div class='card-img rounded d-block align-m'>
                        {% endif %}
                    <p class='card-title'>{{exercise.name}}</p>
                    <div class='card-footer d-flex exercise align-content-around
                    mt-auto'>
                        <a class=>d-flex justify-content-center align-items-
                        'center btn btn-detail'> type=>button < href=>{% url 'form_ludicjourney
                        'exercise.id %}>
                            data-toggle=>tooltip data-placement=>bottom
                        <span>Ajouter l'exercice au parcours ludique </span>
                        <i class=>fas fa-puzzle-piece</i></a>
                        <a class=>d-flex justify-content-center align-items-center btn btn-
                        'detail'> type=>button < href=>{% url 'exercise_therapist' %}>
                            data-toggle=>tooltip data-placement=>bottom
                        <span>Liste des exercices</span>
                        <i class=>bi bi-check-lg</i> </a>
                    </div>
                </div>
            </div>
        {% endfor %}
    </div>
    {% endfor %}
```

template --> exercise\_list.html

```
class ExerciseListView(LoginRequiredMixin,
ListView):
    template_name = 'exercise_list.html'
    context_object_name = 'pathology_list'
    model = Pathology

    def get_queryset(self):
        qs = super().get_queryset()
        qs.select_related('exercise')
        return qs
```

views.py --> ListView Version Django

```
class ExerciseListView(LoginRequiredMixin,
ListView):
    template_name = 'exercise_list.html'
    context_object_name = 'exercise_list'
    model = Exercise

    def get_context_data(self, **kwargs):
        context = super().get_context_data()
        exercises = context['exercise_list']
        pathologies = defaultdict(list)
        for exercise in exercises:
            pathologies[exercise.pathology.name].append(exercise)
        path = sorted(pathologies.items())
        context['pathologies'] = path
        return context
```

views.py --> ListView Version Python

## Integration and unit test

This is a test to display the list of exercises.

```
class BaseTest(TestCase):
    def setUp(self):
        self.client = Client()
        self.factory = RequestFactory()
        self.user = get_user_model().objects.create_user(
            username='LucasD', first_name='Lucas', password='12test12', email='test@email.com')
        self.user_id = self.user.pk
        self.user.save()
        self.therapist = User.objects.create_user(
            username='Marieaumont', password='Therapist@25').pk

        self.pathology_id = Pathology.objects.create(
            name='Dyslexie', description='la pathologie .....').pk
        self.profile = Profile(user=self.user, birth_date='2013-05-12', state=1,
                              bio='Lucas est atteint de .....', profile_pic='lucasdesmarais.png',
                              pathology_id=self.pathology_id, therapist_id=self.therapist).save()

        self.exercise = Exercise.objects.create(name='Améliorer ....', description='L'aider ...', picture1='Winter.png',
                                                picture2='puzzle.png', pathology_id=self.pathology_id, therapist_id=self.therapist,
                                                description_game='Rassemble ...', title_game='Puzzle').pk
    return super().setUp()

class ExerciseListTest(BaseTest):
    def setUp(self):
        self.list_url = reverse('exercise_list')
        return super().setUp()

    def test_view_ok(self):
        login = self.client.login(
            username='Marieaumont', password='Therapist@25')
        self.assertTrue(login)

        response = self.client.get(self.list_url)
        assert response.status_code == 200

    def test_queryset(self):
        request = self.factory.get(self.list_url)
        view = ExerciseListView()
        view.request = request
        qs = view.get_queryset()
        self.assertQuerysetEqual(qs, Pathology.objects.all())
```

## Functional test

This is a test to display the list of exercises.

```
class LudicLanguageExerciseTest(StaticLiveServerTestCase):
    def setUp(self):
        self.driver = webdriver.Firefox(options=firefox_options)
        self.user = get_user_model().objects.create_user(
            username='Marieaumont', first_name='Marie', password='12test12', email='test@email.com')
        self.user_id = self.user.pk
        self.user.save()
        self.therapist = Profile(user=self.user, birth_date='1975-05-12', state=2,
                                bio='Marie est spécialisée .....', profile_pic='marieaumont.png'
                                ).save()
        self.pathology_id = Pathology.objects.create(name='Dyslexie').pk

        self.exercise = Exercise.objects.create(name='Améliorer ....', description='L'aider ...», picture1='Winter.png',
                                                picture2=>'puzzle.png», pathology_id=self.pathology_id, therapist_id=self.therapist,
                                                description_game=>'Rassemble ...», title_game='Puzzle').pk
    return super().setUp()

def test_exercise_list(self):
    driver = self.driver
    driver.implicitly_wait(10)
    driver.get("%s%s" % (self.live_server_url, '/login'))
    driver.find_element_by_id('id_username').send_keys('Marieaumont')
    driver.find_element_by_name('password').send_keys('12test12')
    driver.find_element_by_css_selector('.submit').click()
    driver.find_element_by_name('ludic_journey').click()
```

# shell\_plus

## Console shell\_plus

To test the requests, I installed **Django Extensions**.  
To debug the code, I installed and used **IPDB**.

```
language_project — IPython: ts_Project ts/language_project — pipenv Shell > Python — /v3x/9
from ludic_language.workshops.models import Workshop
# Shell Plus Django Imports
from django.core.cache import cache
from django.conf import settings
from django.contrib.auth import get_user_model
from django.db import transaction
from django.db.models import Avg, Case, Count, F, Max, Min, Prefetch, Q, Sum, When
from django.utils import timezone
from django.urls import reverse
from django.db.models import Exists, OuterRef, Subquery
Python 3.9.9 (v3.9.9:ccbbe6a345, Nov 15 2021, 13:29:20)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.1.1 -- An enhanced Interactive Python. Type '?' for help.

[In [1]: User.objects.all()
Out[1]: <QuerySet [<User: admin>, <User: Madelinefranklin>, <User: XavierDam>, <User: FredericJe>, <User: RoseMc>, <User: PatrickPe>, <User: AgatheSa>, <User: CharlotteBr>, <User: MarcusGa>, <User: ClarenceGr>, <User: MaggieEs>, <User: MarthaEs>, <User: Marieaumont>, <User: LilahBa>, <User: LucasDes>]>

[In [2]: Profile.objects.filter(Q(address__city__icontains=search) | Q(address__zip_code__icontains=search))
NameError: name 'search' is not defined
Input In [2], in <cell line: 1>O
----> 1 Profile.objects.filter(Q(address__city__icontains=search) | Q(address__zip_code__icontains=search))
NameError: name 'search' is not defined
[In [3]: Profile.objects.filter(Q(address__city__icontains=Paris) | Q(address__zip_code__icontains=75))
NameError: name 'Paris' is not defined
Input In [3], in <cell line: 1>O
----> 1 Profile.objects.filter(Q(address__city__icontains=Paris) | Q(address__zip_code__icontains=75))
NameError: name 'Paris' is not defined
MyModel.objects.filter(Q(flag=True) | Q(model_num__gt=15))
[In [4]: Profile.objects.filter(Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75))
Out[4]: <QuerySet [<Profile: XavierDam>, <Profile: Marieaumont>, <Profile: RoseMc>]>
[In [5]: Profile.objects.filter(Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75)).filter(profile__state=2)
AttributeError: 'Q' object has no attribute 'filter'
Input In [5], in <cell line: 1>O
----> 1 Profile.objects.filter(Q(flag=True) & Q(model_num__gt=15))
AttributeError: 'Q' object has no attribute 'filter'
Input In [6]: Profile.objects.filter(profile__user=2) Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75)
Input In [6]
----> 1 Profile.objects.filter(profile__user=2) Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75)
SyntaxError: invalid syntax
[In [7]: Profile.objects.filter(state=2, Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75))
Input In [7]
----> 1 Profile.objects.filter(state=2, Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75))
SyntaxError: unmatched ')'
from django.db.models import Q
MyModel.objects.filter(Q(flag=True) | Q(model_num__gt=15), name__startswith="H")
[In [8]: Profile.objects.filter(state=2, Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75))
Out[8]:
<QuerySet [<Profile: Marieaumont>, <Profile: PatrickPe>, <Profile: MarthaEs>, <Profile: RoseMc>, <Profile: Madelinefranklin>, <Profile: FredericJe>, <Profile: XavierDam>], <Q: (<OR: ('address__city__icontains', 'Paris'), ('address__zip_code__icontains', '75'))>>
MultipleObjectsReturned: QuerySet found more than one item. Use <QuerySet>.get() if you are trying to get one specific object that takes keyword arguments such as <filter>, <exclude>, <get>. Make
sure that when you use with <get> that you will only return one object or the <MultipleObjectsReturned> exception will be raised.
Profile.objects.filter(state=2, Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75))
Input In [9]
----> 1 Profile.objects.filter(state=2, Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75))
SyntaxError: positional argument follows keyword argument
[In [10]: Profile.objects.filter(Q(address__city__icontains='Paris') | Q(address__zip_code__icontains=75), state=2)
Out[10]: <QuerySet [<Profile: XavierDam>, <Profile: Marieaumont>, <Profile: RoseMc>]>
[In [11]: ]
```

# Coverage test

Coverage --> 93%

coverage run manage.py test -v 2 && cover report  
&& html

Coverage report: 93%				
Module	statements	missing	excluded	coverage
ludic_language/_init_.py	0	0	0	100%
ludic_language/base/_init_.py	0	0	0	100%
ludic_language/base/admin.py	1	0	0	100%
ludic_language/base/apps.py	4	0	0	100%
ludic_language/base/views.py	3	0	0	100%
ludic_language/exercises/_init_.py	0	0	0	100%
ludic_language/exercises/admin.py	5	0	0	100%
ludic_language/exercises/apps.py	4	0	0	100%
ludic_language/exercises/forms.py	26	0	77	100%
ludic_language/exercises/models.py	24	0	0	100%
ludic_language/exercises/views.py	78	11	0	86%
ludic_language/profiles/_init_.py	0	0	0	100%
ludic_language/profiles/admin.py	8	0	0	100%
ludic_language/profiles/apps.py	4	0	0	100%
ludic_language/profiles/forms.py	41	0	0	100%
ludic_language/profiles/models.py	39	1	0	97%
ludic_language/profiles/views.py	83	10	0	88%
ludic_language/reviews/forms.py	21	0	0	100%
ludic_language/reviews/models.py	11	0	0	100%
ludic_language/workshops/_init_.py	0	0	0	100%
ludic_language/workshops/admin.py	1	0	0	100%
ludic_language/workshops/apps.py	4	0	0	100%
ludic_language/workshops/forms.py	25	2	0	92%
ludic_language/workshops/models.py	12	1	0	97%
ludic_language/workshops/views.py	56	1	0	98%
manage.py	12	2	0	83%
<b>Total</b>	<b>462</b>	<b>34</b>	<b>0</b>	<b>93%</b>

```
language_project — pipenv shell • bash — 283x91
Applying exercises.0008_alter_exercise_picture1Alter_exercise_picture2... OK
Applying profiles.0008_alter_profile_therapist_pathology... OK
Applying exercises.0009_exercise_description_game_exercise_title_game_and_more... OK
Applying exercises.0010_alter_ludicjourney_assessment_and_more... OK
Applying exercises.0011_pathology_therapist_pathology... OK
Applying profiles.0003_remove_profile_therapist_pathology... OK
driver = self.driver
Applying sessions.0001_initial... OK
driver.get('http://'+(self.live_server_url, '/login/'))
driver.implicitly_wait(10)
Applying workshops.0001_initial... OK
driver.find_element_by_id('id_username').send_keys('Marieaumont')
Applying workshops.0002_remove_workshop_report... OK
driver.find_element_by_name('password').send_keys('12test12')
Applying workshops.0003_workshop_report... OK
driver.find_element_by_css_selector('.submit').click()
System check identified no issues (0 silenced).
test_queryset_Ludic_language.exercises.tests.ExerciseListTest ... ok
test_view_ok_Ludic_language.exercises.tests.ExerciseListTest ... ok
test_form_ok_Ludic_language.exercises.tests.LudicJourneyAddTest ... ok
test_view_ok_Ludic_language.exercises.tests.LudicJourneyAddTest ... ok
test_view_ok_Ludic_language.exercises.tests.LudicJourneyDetailTest ... ok
test_queryset_Ludic_language.exercises.tests.ExerciseListTest ... ok
test_view_ok_Ludic_language.exercises.tests.ExerciseListTest ... ok
test_detail_Ludic_language.exercises.tests.ExerciseListTest ... ok
test_userprofile_missing_email_invalid(Ludic_language.profiles.tests.test_forms.UserProfileFormTest) ... ok
test_userprofile_missing_first_name(Ludic_language.profiles.tests.test_forms.UserProfileFormTest) ... ok
test_userprofile_password_invalid(Ludic_language.profiles.tests.test_forms.UserProfileFormTest) ... ok
test_userprofile_valid(Ludic_language.profiles.tests.test_forms.UserProfileFormTest) ... ok
test_age_invalid(Ludic_language.profiles.tests.test_models.ProfileTest) ... ok
test_delete_valid(Ludic_language.profiles.tests.test_views.PatientDeleteTest) ... ok
test_status_code_200(Ludic_language.profiles.tests.test_views.PatientDetailTest) ... ok
test_view_ok_Ludic_language.profiles.tests.ExerciseListTest ... ok
test_form_ok_Ludic_language.profiles.tests.ExerciseListTest ... ok
test_view_ok_Ludic_language.profiles.tests.ExerciseListTest ... ok
test_form_valid(Ludic_language.workshops.tests.test_forms.ReformFormTest) ... /Users/xavgab/Documents/Python/13_Projet_13/language_project/.venv/lib/python3.9/site-packages/django/db/models/fields/__init__.py (2022-02-12 15:00:00) while time zone support is active.
    driver.find_element_by_name('bit_ch_date').send_keys('12/12/2015')
    driver.find_element_by_name('bio').send_keys('Clarence est un ....')
ok
pathology.html
test_workshop_invalid_date(Ludic_language.workshops.tests.test_forms.WorkshopFormTest) ... ok
test_workshop_invalid_link(Ludic_language.workshops.tests.test_forms.WorkshopFormTest) ... ok
test_workshop_valid(Ludic_language.workshops.tests.test_forms.WorkshopFormTest) ... ok
test_report_none(Ludic_language.workshops.tests.test_models.WorkshopTest) ... ok
test_report_ok(Ludic_language.workshops.tests.test_models.WorkshopTest) ... ok
test_report_list_patient(Ludic_language.workshops.tests.test_views.ReportDetailViewTest) ... ok
test_report_list_patient(Ludic_language.workshops.tests.test_views.ReportListViewTest) ... /Users/xavgab/Documents/Python/13_Projet_13/language_project/.venv/lib/python3.9/site-packages/django/db/models/fields/__init__.py (2022-03-11 00:00:00) while time zone support is active.
    driver.find_element_by_css_selector('.submit').click()
    driver.find_element_by_name('patient').click()
warnings.warn(
ok
pathology.html
test_update_report_speech(Ludic_language.workshops.tests.test_views.ReportUpdateTest) ... ok
test_form_ok(Ludic_language.workshops.tests.test_views.WorkshopAddViewTest) ... ok
test_view_ok(Ludic_language.workshops.tests.test_views.WorkshopAddViewTest) ... ok
    driver
test_exercise_list(Ludic_language.exercises.tests.test_functional.LudicLanguageExerciseTest) ... ok
test_form(Ludic_language.profiles.tests.test_functional.LudicLanguageProfileTest) ... ok
test_login(Ludic_language.profiles.tests.test_functional.LudicLanguageProfileTest) ... ok
    find_element_by_name("search_therapist")
test_search_therapist(Ludic_language.profiles.tests.test_functional.LudicLanguageProfileTest) ... ok
    search_bar.send_keys("Paris")
    search_bar.send_keys(Keys.RETURN)
Ran 35 tests in 32.692s
ok
Destroying test database for alias 'default' ('test_ludiclanguage_db')...
Name      # of assertions      Stmts      Miss      Cover
ludic_language/_init_.py          0          0     100%
ludic_language/base/_init_.py      0          0     100%
ludic_language/base/admin.py       1          0     100%
ludic_language/base/apps.py        4          0     100%
ludic_language/base/views.py       3          0     100%
ludic_language/exercises/_init_.py 0          0     100%
ludic_language/exercises/admin.py 5          0     100%
ludic_language/exercises/apps.py   4          0     100%
ludic_language/exercises/forms.py 26          6    77%
ludic_language/exercises/models.py 24          0     100%
ludic_language/exercises/views.py  78         11    86%
ludic_language/profiles/_init_.py   0          0     100%
ludic_language/profiles/admin.py   8          0     100%
ludic_language/profiles/apps.py    4          0     100%
ludic_language/profiles/forms.py   41          0     100%
ludic_language/profiles/models.py  39         1    97%
ludic_language/profiles/views.py   83         10    88%
ludic_language/settings.py        21          0     100%
ludic_language/urls.py            11          0     100%
ludic_language/workshops/_init_.py 0          0     100%
ludic_language/workshops/admin.py  1          0     100%
ludic_language/workshops/apps.py   4          0     100%
ludic_language/workshops/forms.py  25          2    92%
ludic_language/workshops/models.py 12         1    92%
ludic_language/workshops/views.py  56         1    98%
manage.py                          12         2    83%
TOTAL                            462        34    93%
Wrote HTML report to htmlcov/index.html
([language_project] ) bash-3.2$
```

## Data migration

### Data migration

Creating data using the Admin platform Django  
python manage.py makemigrations --empty exercises  
python manage.py migrate  
Heroku run bash  
python manage.py migrate

### Admin Django

Creating data using the Admin platform Admin Django  
Heroku

```
def create_pathologies(apps, schema_editor):
    Pathology = apps.get_model('exercises', 'Pathology')
    Pathology.objects.get_or_create(name='Dyslexie', description='Troubles de la lecture et de l'écriture')

def reverse_pathologies(apps, schema_editor):
    pass

class Migration(migrations.Migration):

    dependencies = [
        ('exercises', '0011_pathology_therapist_pathology'),
    ]

    operations = [
        migrations.RunPython(create_pathologies, reverse_pathologies),
    ]
migrations --> 0012_auto_44455466.py
```

The screenshot shows the Django Admin interface at the URL <https://ludic-language.herokuapp.com/admin/>. The page title is "Administration de Django". The main content area is titled "Site d'administration". It features three main sections: "AUTHENTIFICATION ET AUTORISATION" (Groups, Users), "EXERCICES" (Exercises, Pathologys), and "PROFILES" (Address). Each section has "Ajouter" and "Modification" buttons. On the right side, there is a sidebar titled "Actions récentes" with a list of recent actions, each with a green plus sign icon and a brief description.

AUTHENTIFICATION ET AUTORISATION	
Groupes	<a href="#">Ajouter</a> <a href="#">Modification</a>
Utilisateurs	<a href="#">Ajouter</a> <a href="#">Modification</a>

EXERCICES	
Exercises	<a href="#">Ajouter</a> <a href="#">Modification</a>
Pathologys	<a href="#">Ajouter</a> <a href="#">Modification</a>

PROFILES	
Address	<a href="#">Ajouter</a> <a href="#">Modification</a>

Actions récentes	
Mes actions	
<a href="#">+ Puzzle</a>	Exercise
<a href="#">+ Recette de Cookies pour 6 personnes</a>	Exercise
<a href="#">+ Tri sélectif</a>	Exercise
<a href="#">+ Racontes moi une histoire</a>	Exercise
<a href="#">+ Chantes moi la suite ...</a>	Exercise

## Deployment

Settings --> prod.py

```
SECRET_KEY = os.getenv('SECRET_KEY')
sentry_sdk.init(
    https://223debeaaaf0d46aa97f5f0f1e6cd5c45@o1176728.ingest.sentry.io/6274790,

    # Set traces_sample_rate to 1.0 to capture 100%
    # of transactions for performance monitoring.
    # We recommend adjusting this value in production.
    traces_sample_rate=1.0
)

BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
INSTALLED_APPS = INSTALLED_APPS + ['storages', ]
DEBUG = False
ALLOWED_HOSTS = ['.herokuapp.com']
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': os.getenv('DB_NAME'),
        'USER': os.getenv('DB_USER'),
        'PASSWORD': os.getenv('DB_PASSWORD'),
        'HOST': os.getenv('DB_HOST'),
        'PORT': '5432'
    }
}
# Amazon S3
AWS_ACCESS_KEY_ID = os.getenv('AWS_ACCESS_KEY_ID')
AWS_SECRET_ACCESS_KEY = os.getenv('AWS_SECRET_ACCESS_KEY')
AWS_STORAGE_BUCKET_NAME = os.getenv('AWS_STORAGE_BUCKET_NAME')

AWS_S3_REGION_NAME = os.getenv('AWS_S3_REGION_NAME')
AWS_S3_SIGNATURE_VERSION = os.getenv('AWS_S3_SIGNATURE_VERSION')
AWS_S3_FILE_OVERWRITE = False
AWS_DEFAULT_ACL = None

DEFAULT_FILE_STORAGE = 'storages.backends.s3boto3.S3Boto3Storage'

# Activate Django-Heroku.
django_heroku.settings(locals())


```

settings --> prod.py

# Heroku

Salesforce Platform

HEROKU

Personal > ludic-language

Overview Resources Deploy Metrics Activity Access Settings

Get a complete visualization of your app in a team-based continuous delivery environment with [Heroku Pipelines](#). [Hide](#) [Create a Heroku Pipeline](#)

Installed add-ons \$0.00/month [Configure Add-ons](#)

Heroku Postgres Hobby Dev postgresql-infinite-24007	Sentry Free sentry-lively-79860
---	---------------------------------

Dyno formation \$0.00/month [Configure Dynos](#)

This app is using free dynos

web gunicorn ludic\_language.wsgi --log-file - [ON](#)

Collaborator activity [Manage Access](#)

gabrielleazadian@gmail.com	<a href="#">23 deploys</a>
----------------------------	----------------------------

Latest activity [All Activity](#)

gabrielleazadian@gmail.com: Deployed e0a2a6b2	Yesterday at 7:12 PM · v50
gabrielleazadian@gmail.com: Build succeeded	Yesterday at 7:11 PM · <a href="#">View build log</a>
gabrielleazadian@gmail.com: Set AWS_S3_REGION_NAME config var	Yesterday at 7:09 PM · v49
gabrielleazadian@gmail.com: Set AWS_S3_SIGNATURE_VERSION config var	Yesterday at 7:06 PM · v48
gabrielleazadian@gmail.com: Deployed 74da26b3	Yesterday at 6:50 PM · v47
gabrielleazadian@gmail.com: Build succeeded	Yesterday at 6:50 PM · <a href="#">View build log</a>
gabrielleazadian@gmail.com: Deployed dae6b81e	Yesterday at 6:44 PM · v46
gabrielleazadian@gmail.com: Build succeeded	Yesterday at 6:43 PM · <a href="#">View build log</a>

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Slug size 116.5 MiB of 500 MiB

Heroku git URL <https://git.heroku.com/ludic-language.git>

Config Vars

Config vars change the way your app behaves. In addition to creating your own, some add-ons come with their own.

Config Vars

AWS_ACCESS_KEY_ID	AKIA5QSEISC6YSIYQEXR	<a href="#">Edit</a> <a href="#">X</a>
AWS_S3_REGION_NAME	eu-west-3	<a href="#">Edit</a> <a href="#">X</a>
AWS_S3_SIGNATURE_VERSION	s3v4	<a href="#">Edit</a> <a href="#">X</a>
AWS_SECRET_ACCESS_KEY	ugHRupPSzOtGU/W4BH4pZuBBjocyh1B/12C1	<a href="#">Edit</a> <a href="#">X</a>
AWS_STORAGE_BUCKET_NAME	ludiclanguagebucket	<a href="#">Edit</a> <a href="#">X</a>
AWS_URL	<a href="https://ludiclanguagebucket.s3.amazonaws.com/">https://ludiclanguagebucket.s3.amazonaws.com/</a>	<a href="#">Edit</a> <a href="#">X</a>
DATABASE_URL	postgres://gsrdgbkdepaehn:d4d95bdæ0c	<a href="#">Edit</a> <a href="#">X</a>
DJANGO_SETTINGS_MODULE	ludic_language.settings.prod	<a href="#">Edit</a> <a href="#">X</a>
ENVIRONMENT	production	<a href="#">Edit</a> <a href="#">X</a>
SECRET_KEY	9q2eh0+r^q06u@mv#2i2et-5v^+h2@^kmxyrk	<a href="#">Edit</a> <a href="#">X</a>
SENTRY_DSN	<a href="https://223debeaaaf0d46aa97f5f0fe6cd5">https://223debeaaaf0d46aa97f5f0fe6cd5</a>	<a href="#">Edit</a> <a href="#">X</a>

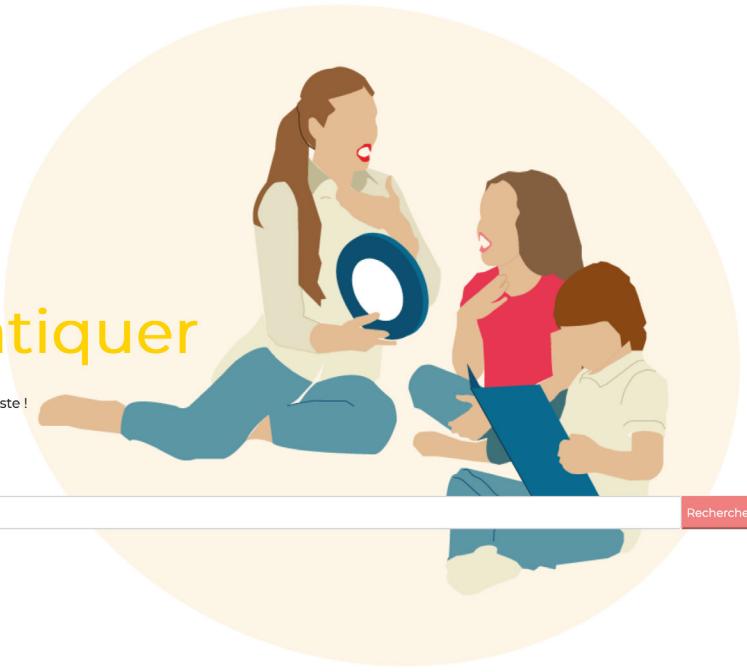
Ludic Language

Heroku --> plateform deployed

L'art de bien parler...  
C'est l'art de bien pratiquer

Apprendre en s'amusant pour lutter contre les troubles du langage avec l'aide d'un orthophoniste !

Où ?  Rechercher



LudicLanguage

https://ludic-language.herokuapp.com

ABOUT US MENTIONS LEGALES CONTACT SIGN IN VOUS ÊTES PROFESSIONNEL DE SANTÉ