SCALE FOR PROJECT DJANGO - 2 - SQ (/PROJECTS/DJANGO-2-SQL)

You should evaluate 1 student in this team

★

Git repository

```
git@vogsphere.42paris.fr:vogsphere/intra-uuid-9bd25ca5-e787-40
```

Introduction

For the smooth running of this evaluation, please respect the following rules:

- Remain polite, kind, respectful and constructive whatever happens during this conversation. It's a matter of confidence between you and the 42 community.

- Highlight the potential problems you ‘ve had with the work you're presented to the person or the group you're grading, and take the time to talk about and discuss those issues.

- Accept the fact that the exam subject or required functions might lead to different interpretations. Listen to your discussion partner's perspective with an open mind (are they right or wrong ?) and grade them as fairly as possible.
42's teaching methods can make sense only if peer-evaluation is taken seriously.

Guidelines

- You must only evaluate what you will find in the student's or group's GiT repository.

- Take the time to check that the GiT repository matches the student or group and the project.

- Double check that no malicious alias was used to mislead you and make you grade something different from the official repository content.

- If a script supposed to help evaluate the exam is supplied by either side, the other side will have to strictly check it to avoid nasty surprises.

- If the evaluating student has not yet taken this project, they will have to read the exam subject in its entirety before starting the evaluation.

- Use the flags available on this grading system to signal an empty or non-funcional project, a norm flaw, cheating, etc. In that case, evaluation stops and final grade is 0 (or -42 if it's a cheating problem). However, if it's not a cheating problem, you are invited to keep talking about the work that has been done (or not done, as a matter of fact) in order to identify the issues that lead to this stalemate and avoid it next time.

Attachments

subject.pdf (https://cdn.intra.42.fr/pdf/pdf/59667/en.subject.pdf)

d05.tar.gz (https://cdn.intra.42.fr/document/document/10787/d05.tar.gz)

Foreword

description

Observing the rules

- The repo contains the evaluated student's or group's work.
- The evaluated student or group can explain their work anytime during the evaluation.
- General and specific instructions of the day will be observed during the whole evaluation.
- For this project, you have to clone their Git repository on their station.

Yes No

Training Python-Django - 2 - ORM

*- Hard code the result of an exercise means this exercise get a 0. - You must test the exercises wit exercise doesn't work with Python3, even if it works with Python2, it's considered invalid. - Start rec new database: psql -c "drop database formationdjango;" psql -c "create database formationdjango; on osx) dropdb formationdjango createdb formationdjango - If some there are some migration folde delete them: rm -rf */migrations (from the project's root) - Every instruction requiring an URL consid development server of the exercise is running thanks to the ./manage.py runserver command. - Yo tests following the indicated order.*

Exercise 00 - Building a table in SQL

- Enter this command in the terminal:

  psql -U djangouser -d formationdjango -c "\d ex00_movies"

  The output must be: Did not find any relation
  named "ex00_movies".

Start the Django development server.

- Enter the url 127.0.0.1:8000/ex00/init in the brower:

  You must get a page simply indicating "OK"

- Enter this command in the terminal again:

  psql -U djangouser -d formationdjango -c "\d ex00_movies"

  The output should be:

  Column | Type | Modifiers

  ---------------+------------------------+-----------
  episode_nb | integer | not null title
  | character varying(64) | not null opening_crawl | text
  | director | character varying(32) | not null
  producer | character varying(128) | not null
  release_date | date | not null Indexes:
  "ex00_movies_pkey" PRIMARY KEY, btree (episode_nb)
  "ex00_movies_title_key" UNIQUE CONSTRAINT, btree (title)

- Once again, enter the URL: 127.0.0.1:8000/ex00/init The page should
  still display "OK".

Yes No

Exercise 01 - Build a table in the ORM

- Enter this command in the terminal:

  psql -U djangouser -d formationdjango -c "\d ex01_movies"

  The output should be: Did not find any relation
  named "ex01_movies".

- Ask the student to create the migration for the application ex01 only.
- Ask them to apply the migration.
- Enter the command in the terminal once again:

  psql -U djangouser -d formationdjango -c "\d ex01_movies"

  The output must be:

```
 Column     |          Type          | Modifiers

---------------+------------------------+----------- title
| character varying(64) | not null episode_nb |
integer | not null opening_crawl | text
| director | character varying(32) | not null
producer | character varying(128) | not null
release_date | date | not null Indexes:
"ex01_movies_pkey" PRIMARY KEY, btree (episode_nb)
"ex01_movies_title_key" UNIQUE CONSTRAINT, btree (title)
"ex01_movies_title_e9be860e_like" btree (title
varchar_pattern
```

Yes No

Exercise 02 - Inserting SQL data

- test the urls: 127.0.0.1:8000/ex02/display and 127.0.0.1:8000/ex02/populate "No data available" or error messages indicating that the ex02_movies table doesn't exist must be displayed.
- test the url: 127.0.0.1:8000/ex02/init "OK" must be displayed.
- test the url: 127.0.0.1:8000/ex02/display "No data available" must be displayed.
- test the url: 127.0.0.1:8000/ex02/populate for each film, "OK" must be displayed.
- test the url: 127.0.0.1:8000/ex02/display All the data described in the subject must be displayed. The order doesn't matter. The opening_crawl column must appear and display "None" or nothing at all.

Yes No

Exercise 03 - Inserting data in the ORM

- Operate the migration of the application ex03 only.
- test the url: 127.0.0.1:8000/ex03/display "No data available" must be displayed.
- test the url: 127.0.0.1:8000/ex03/populate for each film, "OK" must be displayed.
- test the url: 127.0.0.1:8000/ex03/display All the data described in the subject must be displayed. The order doesn't matter. The opening_crawl column must appear and display "None" or nothing at all.

Yes No

Exercise 04 - Deleting data in SQL

- test the urls: 127.0.0.1:8000/ex04/display, 127.0.0.1:8000/ex04/populate and 127.0.0.1:8000/ex04/remove . "No data available" or error messages indicating that the table ex04_movies doesn't exist must be displayed.
- test the url: 127.0.0.1:8000/ex04/init "OK" must be displayed.
- test the urls: 127.0.0.1:8000/ex04/display and 127.0.0.1:8000/ex04/remove "No data available" must be displayed.
- test the url: 127.0.0.1:8000/ex04/populate for each film, "OK" must be displayed.
- test the url: 127.0.0.1:8000/ex04/display All the data described in the subject must be displayed. The order doesn't matter. The opening_crawl column must appear and display "None" or nothing at all.
- test the url: 127.0.0.1:8000/ex04/remove A page must display a form with a drop-down list of film titles matching the ones in the ex04_movies table.
  Select a film and validate. The form must redisplay and the film you've just selected must not appear in the list anymore.

- test the url: 127.0.0.1:8000/ex04/display All the data described in the subject must be displayed, minus the film you've just deleted.
- test the url: 127.0.0.1:8000/ex04/populate For each film, an error must be displayed, except for the film you've just deleted that must make an "OK" appear.

Yes No

Exercise 05 - Deleting data in the ORM

- Operate the migration of the application ex05 only.
- test the urls: 127.0.0.1:8000/ex05/display and 127.0.0.1:8000/ex05/remove "No data available" must be displayed.
- test the url: 127.0.0.1:8000/ex05/populate For each film, "OK" must be displayed.
- test the url: 127.0.0.1:8000/ex05/display All the data described in the subject must be displayed. The order doesn't matter. The opening_crawl column must appear and display "None" or nothing at all.
- test the url: 127.0.0.1:8000/ex05/remove A page must display a form with a drop-down list of film titles matching the ones in the Movie model of this application.
  Select a film and validate. The form must redisplay and the film you've just selected must not appear in the list anymore.
- test the url: 127.0.0.1:8000/ex05/display All the data described in the subject must be displayed, minus the film you've just deleted.
- test the url: 127.0.0.1:8000/ex05/populate For each film, an error must be displayed, except for the film you've just deleted that must make an "OK" appear.

Yes No

Exercise 06 - Updating data in SQL

- test the urls: 127.0.0.1:8000/ex06/display, 127.0.0.1:8000/ex06/populate and 127.0.0.1:8000/ex06/update. "No data available" or error messages indicating that the table ex06_movies doesn't exist must be displayed.
- test the url: 127.0.0.1:8000/ex06/init "OK" must be displayed.
- test the urls: 127.0.0.1:8000/ex06/display 127.0.0.1:8000/ex06/update "No data available" must be displayed.
- test the url: 127.0.0.1:8000/ex06/populate For each film, "OK" must be displayed.
- test the url: 127.0.0.1:8000/ex06/display All the data described in the subject must be displayed, including the fields created and updated that must be set at a similar date. The order doesn't matter. The opening_crawl column must appear and display "None" or nothing at all.
- test the url: 127.0.0.1:8000/ex06/update A page must display a form with a drop-down list of film titles matching the ones in the ex06_movies table.
  Select a film, enter a text and validate. Then go to this address: 127.0.0.1:8000/ex06/display. The field "crawling_text" of the selected film must now include the text typed in the form. The field "updated" must be updated with the current time whereas the "created" fields mustn't have changed.

Yes No

Exercise 07 - Updating data in the ORM

- Operate the migration of the application ex05 only.

- test the urls: 127.0.0.1:8000/ex07/display and 127.0.0.1:8000/ex07/update "No data available" must be displayed.
- test the url: 127.0.0.1:8000/ex07/populate For each film, "OK" must be displayed.
- test the url: 127.0.0.1:8000/ex07/display All the data described in the subject must be displayed, included the created and updated fields which must be set at a similar date. The order doesn't matter. The opening_crawl column must appear and display "None" or nothing at all.
- test the url: 127.0.0.1:8000/ex07/update A page must display a form with a drop-down list of film titles matching the ones in the ex07 of this application as well as a text field.
  Select a film, type some text and validate. Then go to this address: 127.0.0.1:8000/ex07/display. The field "crawling_text" of the selected film must now include the text typed in the form. The field "updated" must be updated with the current time whereas the "created" fields mustn't have changed.

Yes No

Exercise 08 - Foreign key in SQL

- test the urls: 127.0.0.1:8000/ex08/display and 127.0.0.1:8000/ex08/populate "No data available" or error messages indicating the tables ex08_movies and ex08_people don't exist must be displayed.
- test the url: 127.0.0.1:8000/ex08/init "OK" must be displayed for each table.
- Let's check the ex08_people table. In the terminal, type the command: psql -U djangouser -d formationdjango -c "\d ex08_people" Check the output is indeed the description of a table with the fields and specifications described in the subject.
- Do the same with the ex08_planets table with the command: psql -U djangouser -d formationdjango -c "\d ex08_planets"
- test the urls: 127.0.0.1:8000/ex08/display. "No data available" must be displayed.
- test the url: 127.0.0.1:8000/ex08/populate For each file, "OK" must be displayed.
- test the url: 127.0.0.1:8000/ex08/display the result must be: Saesee Tiin - Iktotch - arid, rocky, windy Tion Medon
  - Utapau - temperate, arid, windy

Yes No

Exercise 09 - Foreign key in the ORM

- Operate the migration of the application ex09.
- Let's check the People model. In the terminal, enter the command: psql -U djangouser -d formationdjango -c "\d ex09_people" Check that the output indeed is the description of a table with the fields and specifications described in the subject.
- Do the same with the Planets model with the command: psql -U djangouser -d formationdjango -c "\d ex09_planets"
- test the url: 127.0.0.1:8000/ex09/display. The following should be displayed: "No data available, please use the following command line : "</br> followed by a command line. Execute it in the terminal from the project's root. If everything went well: "Installed 147 object(s) from 1 fixture(s)" must be displayed.
- test the url: 127.0.0.1:8000/ex09/display The display should be: Saesee Tiin - Iktotch - arid, rocky, windy Tion Medon

- Utapau - temperate, arid, windy

Yes No

Exercise 10 - Many to many in the ORM

- Operate the migration of the application ex09.
- Install the ex10_initial_fixture.json fixture.
- test the url: 127.0.0.1:8000/ex10/: A form must be dislayed. Check that that the fields match the ones described in the subject.
- Enter the following data and validate the form (the date format might have changed. Adapt it if necessary).

  Movies minimum release date : 1980-01-01 Movies maximum release date: 1989-12-31 Planet diameter greater than: 1 Character gender : n/a

The result must be (Not necessarily in this order):

```
Return of the Jedi                C-3PO   n/a     Tatooine
```

10465 Return of the Jedi R2-D2 n/a Naboo
12120 The Empire Strikes Back C-3PO n/a Tatooine
10465 The Empire Strikes Back R2-D2 n/a Naboo
12120

- Fill the form once again. This time, enter two identical dates (and any other data for the othe page must display: "Nothing corresponding to your research".

Yes No

Ratings

Don’t forget to check the flag corresponding to the defense

Ok Outstanding project

Empty work Incomplete work Cheat Crash Concer

Forbidden function

Conclusion

Leave a comment on this evaluation (2048 chars max)

Finish evaluation