Python Web Framework Retake Exam - 20 August 2023

Individual Project Assignment

This is the Individual Project Assignment for the Python Web Framework Course @ SoftUni.

1. General Requirements

Your Web application should use the following technologies, frameworks, and development techniques:

- The application must be implemented using **Django Framework**
  - The application must have at least **10 web pages**:
    - Can be created using **function-based views** or/and **class based-views**;
    - At least **5 of them must be class-based views**.
  - The application must have at least **5 independent models** (models created by extending, inheritance, and one-to-one relation are considered one model).
  - The application must have at least **5 forms**.
  - The application must have at least **5 templates**.

- Use **PostgreSQL** as a **Database Service**
  - Optionally, you can use **multiple storages** (including PostgreSQL), e.g., files, other Web services, databases (e.g., **MySQL/MariaDB/Oracle** / etc.)

- Use **Django Template Engine** or make the **Front-End** using **JavaScript**.
- **Templates** (your views must return HTML files) **- the same template could be re-used/ used multiple times** (according to adjustments, if such needed).
- Implement **Web Page Design** based on **Bootstrap / Google Material Design,** or **design your own**.

- The application must have **login/register/logout** functionality.
- The application must have **a public part** (A part of the website, which is accessible by everyone – un/authenticated users and admins).
- The application must have **a private** part (accessible only by authenticated users and admins).
- The application must have **a customized admin site** (accessible only by admins):
  - Add at least 5 custom options (in total) to the admin interface (e.g., filters, list display, ordering, etc.).

- **Unauthenticated users** (public part) **have only 'get' permissions, e.g., landing page, details, about page, and login/ register 'post' permissions.**
- **Authenticated users** (private part) **have full CRUD for all their created content.**
- **Admins** - at **least 2 groups** of admins:
  - One **must** have permission to do **full CRUD functionalities (superusers)**;
  - The other/s have permission to do **limited CRUD functionalities (staff)**.
  - User **roles** could be **manageable** from the admin site.
  - Make sure the **role management** is **secured** and **error-safe**.

- Implement **Exception Handling** and **Data Validation** to avoid **crashes** when **invalid data** is entered (both **client-side** and **server-side**)
    - When validating data, show appropriate messages to the user
- **For students who took part in the regular exam – they must implement a new "main" functionality!** It is considered **two additional views** with **CRUD operations** that are **extending** your current project. This should be done for a logged-in user.

2. Online Project Defense

Each student will have to deliver an **online defense** of their work in front of a trainer jury. Students will have **only 20 minutes**, which must be allocated as follows:

- **Demonstrate** how the application works (very shortly).
- Show the **source code** and explain how it works.
- Answer the jury's **questions**

Please be **strict with the timing**! On the 10th minute, your presentation ends. The remaining time will be for the Question/Answers session.

Open the project assets **beforehand** to save time.

Be **well prepared** to present the maximum of your work within the time given. It is highly recommended that you practice the presentation at home with a stopwatch to ensure that you will fit in the time provided.

3. Assessment Criteria

General Requirements – 70%

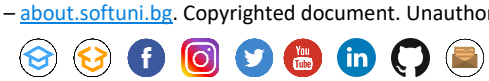
- **Functionality** - **0…20**
- Implementing **views correctly** (views should only do their work) **– 0...5**
- Implementing **models correctly - 0…5**
- Implementing **forms correctly - 0…5**
- Implementing **templates correctly** (using the template language) **– 0…5**
- Implement **Responsive Web Page Design - 0…5**
- Implementing **login/register functionality correctly - 0…5**
- **Exception handling** and/or **Data validation** (validation in the models and/or the forms) **– 0…5**
- **Security** (prevent SQL injection, XSS, CSRF, parameter tampering, etc.) **– 0…5**
- **Code quality** (well-structured code, following the MTV pattern, following SOLID principles, etc.) – **0…10**

Answering Questions – 30 %

Answer theoretical questions from all courses in SoftUni's Python program and potential functionality outside the scope of the project.

Bonuses – up to 15 %

- Write tests **(Unit & Integration)** for your **views/models/forms** - at least 10 tests
- Writing **asynchronous view/s** somewhere in the project
- **Extend your Django project with REST Capabilities**
- Extend **Django user**
- Host the application in a **cloud environment**



- **Additional functionality**, not explicitly described in this section, will be counted as a bonus if it has practical usage

4. Submission Deadline

- You **must** submit a **link** to your project **before 23:59 on 17-Aug-2023** using a survey that will show up on **15-Aug-2023**. You **can continue working** on your project until the end of 18-Aug-2023.
- A presentation schedule will be available on **18-Aug-2023** and will include only the projects that were **submitted beforehand**. Non-submitted projects will **NOT** be evaluated.

5. Additional Requirements

- Follow the **best practices** for **Object-Oriented design** and **high-quality code** for the **Web application**:
    - Use **data encapsulation**.
    - Use **exception handling** properly.
    - Use **inheritance**, **abstraction,** and **polymorphism** properly.
    - Follow the **principles** of **strong cohesion** and **loose coupling**.
    - Correctly **format** and **structure** your **code**, name your **identifiers**, and make the code **readable**.

- Well-looking **user interface** (**UI**).
- Good user experience (**UX**).
- Use a **source control system** by choice, e.g., **GitHub**, **BitBucket**.
    - Submit a link to your public source code repository.