Цифровая олимпиада «Волга–IT`20»

Дисциплина «Интернет-программирование (PHP)»
Задание финального этапа

Задание 1. Упрощенная CMS

Web приложение с личным кабинетом.

Исходные данные:

Нужно разработать веб-приложение в котором можно совершить покупки (без платёжной системы), просматривать данные о покупках:

- № заказа;
- сумма в рублях;
- дата покупки;
- статус заказа (меняется в администратором);
- адрес доставки;
- состояние счёта в личном кабинете

Доступные технологии:

- Backend: Чистый PHP или любой фреймворк;
- Frontend: React, Vue.js, JQuery;
- CSS: Material UI, Twitter Bootstrap, Tailwind CSS и в целом любой CSS фреймворк или библиотека компонентов.

Базовая задача (обязательные требования):

Общие требования:

- исходные данные должно быть обязательно получены с сервера;

- приложение должно быть выложено на https://github.com;
- в корне репозитория должен присутствовать файл readme.md с описанием архитектуры и инструкциями по запуску приложения;
- для тестирования работы приложения должно быть создано несколько учетных записей, информация о них должна присутствовать в readme.md файле;

Что обязательно должно быть реализовано:

- регистрация \ авторизация;
- два уровня доступа, если не пользователь не авторизован, то ему видна только страница авторизации или регистрации. После авторизации ему должны быть доступны: витрина со списком товаров для покупки и личный кабинет;
- текущий баланс пользователя (при регистрации будет даваться определённая сумма, например 2000р);
- список заказов пользователя, представленный в табличном виде, с возможностью сортировки списка, фильтрации (по статусам, дате, сумме и тд, минимум по 3м полям), поиска по этим же полям;
- страница конкретного заказа со всей информацией по нему (№, комментарий, дата платежа, статус, сумма и тд);
- создание нового нового заказа, через витрину (должна учитываться сумма, которая осталась у покупателя);
- простая админка для подтверждения \ отмены заказа (можно использовать Laravel Nova, Voyager, Backpack и другие готовые решения);

Максимальное количество баллов: 100

<u>Дополнительные задачи:</u>

- При разработке соблюдать принципы SOLID;
  Максимальное количество баллов: 50
- Фронтэнд - SPA;
  Максимальное количество баллов: 50
- Сделать адаптивный дизайн.
  Максимальное количество баллов: 50

Максимальное количество баллов за задание: 250

Задание 2. Сервис доставки почтовых отправлений.

Исходные данные:

Реализовать веб-сервис для управления почтовыми отправлениями, с REST API для расчета стоимости и оформления доставки.

Доступные технологии:

- Backend: Чистый PHP или любой фреймворк;
- Frontend: React, Vue.js, JQuery;
- CSS: Material UI, Twitter Bootstrap, Tailwind CSS и в целом любой CSS фреймворк или библиотека компонентов.

Базовая задача (обязательные требования):

Общие требования:

- приложение должно быть выложено на https://github.com;
- в корне репозитория должен присутствовать файл readme.md с описанием архитектуры и инструкциями по запуску приложения;
- для тестирования работы приложения должно быть создано несколько учетных записей, информация о них должна присутствовать в readme.md файле;

Что обязательно должны быть реализовано:

1. Регистрация:
    a. Страница регистрации должна содержать следующие поля:
        i. Фамилия
        ii. Имя
        iii. Email
        iv. Пароль
    b. Регистрация должна соблюдать следующие правила валидации:
        i. Email не должен повторяться (т.е. нельзя зарегестрироваться два раза с одним и тем же паролем)
        ii. Пароль должен состоять из более 6 символов (минимум одну букву и цифру)
    c. При успешной регистрации пользователю уходит приветственное письмо (текст и дизайн на свое усмотрение)
    d. Можно зарегистрироваться только клиентом, чтобы стать администратором, надо включить соответствующий флаг в БД.
2. Авторизация:
    a. Страница авторизации состоит из двух полей:

        i. email
        ii. пароль
    b. Должна быть возможность восстановить забытый пароль (ссылка/кнопка):
        i. поле email и кнопка Восстановить
        ii. при нажатии на кнопку, если пользователь с таким Email существует, то ему уходит письмо, с ссылкой на страницу создания нового пароля (страница с полем Пароль и Повторить пароль). Пароли должны совпадать и валидироваться как при регистрации.
        iii. После смены пароля, необходимо попросить клиента авторизоваться с новым паролем.
    c. В случае успешной авторизации пользователь попадает в личный кабинет.
3. Личный кабинет:
    a. Главная страница личного кабинета - список отправлений (стандартная таблица)
        i. список должен поддерживать постраничную навигацию (по 15 отправлений на странице)
            1. В списке должен отображаться:
                a. трек-код
                b. дата создания
                c. статус
                d. город отправления
                e. город назначения
                f. стоимость доставки
        ii. поиск по трекинг коду, статусу, городу назначению и города отправления
        iii. кликнув на отправление можно перейти на детальную информацию отправления
    b. Страница детальной информации отправления
        i. Содержит таблицу с историей смены статусов
        ii. Содержит информацию об отправлении (куда, откуда, вес)
    c. Страница профиля
        i. Содержит информацию о пользователе с возможностью ее редактирования (Имя, Фамилия, email)
        ii. Есть возможность сменить пароль
            1. надо ввести старый пароль и новый пароль

2. пароль должен валидироваться также, как при регистрации

iii. Есть возможность установить флаг получения писем об изменении статуса (если флаг включен, то после смены статуса клиенту приходит письмо с этой информацией)

iv. Есть возможность сгенерировать API ключ.

1. Ключ должен формироваться функцией md5()
2. В функцию может поступать информацию на ваше усмотрение (например Email + рандомное число).

4. Личный кабинет администратора
   a. Главная страница личного кабинета - список клиентов (стандартная таблица)
      i. список должен поддерживать постраничную навигацию (по 15 пользователей на странице)
         1. В списке должен отображаться:
            a. Фамилия Имя
            b. email
            c. количество отправлений
            d. сумма стоимости всех отправлений
      ii. поиск по Имени/Фамилии, по email
      iii. можно отсортировать по сумме стоимости всех отправлений (от большего к меньшему и наоборот)
      iv. кликнув на пользователя должен открыться список отправлений клиента (страница 4.b с фильтром по пользователю)
   b. Страница всех отправлений (стандартная таблица)
      i. список должен поддерживать постраничную навигацию (по 15 отправлений на странице)
         1. В списке должен отображаться:
            a. имя/фамилия клиента
            b. трек-код
            c. дата создания
            d. статус
            e. город отправления
            f. город назначения
            g. стоимость доставки
      ii. Поиск по трекинг коду, статусу, городу отправления, городу назначения

iii. можно отсортировать по сумме стоимости отправления (от большего к меньшему и наоборот)
iv. У отправления можно сменить статус (Selectbox с вариантами статусов)
1. Варианты статусов:
a. ожидает отправки
b. отправлен
c. доставлен
d. выдан получателю
2. При смене статуса уходит письмо (если пользователь включил соответствующий флаг) (см 3.c.iii)

5. Главная страница
a. Поле поиска по трекинг коду
i. Если отправление найдено, то открыть страницу с таблицей с историей смены статусов (без детальной информации об отправлении).
b. Ссылка на регистрацию
c. Ссылка на авторизацию
6. REST API:
a. Доступ к REST API должен быть ограничен наличием клиентского API ключа. Без валидного API ключа все запросы должны откланяться.
b. Расчет стоимости доставки (стоимость доставки зависит от веса посылки)
i. Отправка POST запроса в JSON формате, следующего содержания:
{ "city_from": "", "country_from": "", "city_to": "", "country_to": "", "weight": ""}
ii. Ответ должен включать в себя: данные, которые он принял и стоимость доставки:
{ "info": {"city_from": "", "country_from": "", "city_to": "", "country_to": "", "weight": ""}, "price": ""}
c. Формирование отправления - после выполнения данного запроса, создается отправление (которое потом будет отображаться в списках отправлений)
i. Отправка POST запроса в JSON формате, следующего содержания:

1 { "address_from": "", "city_from": "", "country_from": "", "address_to": "", "city_to": "", "country_to": "", "weight": ""}

Ответ должен включать в себя, данные которые он принял, стоимость доставки, и трек-код

```
1 {  "info": {"address_from": "", "city_from": "",  "country_from": "",  "address_to": "",  "city_to": "",  "country_to": "",  "weight": ""}, "tracking_code": "", "price": ""}Address_from, city_from, country_from, address_to, city_to, country_to, weight, tracking_code, price
```

d. Получить статус отправления
    i. Отправка GET запроса с параметром tracking_code
    ii. Ответ должен включать в себя данные которые он принял и информацию об отправлении:

```
1 {  "tracking_code": "", "info": {"status": "", "address_from": "", "city_from": "",  "country_from": "",  "address_to": "", "city_to": "",  "country_to": "",  "weight": ""}}
```

7. Расчет стоимости доставки:
    a. Все города и страны, на которые происходит запрос считаются валидными. Нет необходимости проверять реальный ли город, или нет.
    b. Если доставка внутри страны, то стоимость доставки рассчитывается по следующей формуле:
        i. S = N(city) *10 * We
            1. S - стоимость доставки
            2. N(city) - количество букв в названии города
            3. We - вес отправления (в граммах)
    c. Если доставка будет в другую страну, то формула следующая:
        i. S = (N(country) + N(city)) *100 * We
            1. S - стоимость доставки
            2. N(country) - количество букв в названии страны
            3. N(city) - количество букв в названии города
            4. We - вес отправления (в граммах)

Максимальное количество баллов: 100

<u>Дополнительные задачи:</u>

- Строгое следование PSR;

Максимальное количество баллов: 50

- Сделать возможность определения стоимости доставки в зависимости от габаритов

    - Запрос на расчет доставки и создание отправления должен включать в себя габариты (высоту, ширину, длину): {"height": "", "width": "", "length": ""}
    - Формула: S = N(city) *10 * We * (H * W * L)
        - H - высота (в мм),
        - W - ширина (в мм),
        - L - длина (в мм)

Максимальное количество баллов: 50

- Реализовать отправку WebHook’ов:
    - При формировании API ключа необходимо запрашивать URL для WebHook’ов.
    - При изменении статуса необходимо отправить WebHook с данными отправления (трек-код, статус отправления, данные отправления)

Максимальное количество баллов: 50

Максимальное количество баллов за задание: 250