**Часть А.**

1. **c) flask startproject**
2. **b) request**
3. **a) @app.route("/")**
4. **d) app.run()**
5. **a) session['user']**
6. **c) make\_response()**
7. **c) @app.route**
8. **a) Использовать несколько @app.route() над функцией**
9. **a) redirect()**

**Часть B.**

**1. Опишите процесс обработки HTTP-запроса в Flask. (4 балла)**

Когда пользователь отправляет HTTP-запрос к приложению Flask:

* Сервер принимает запрос и передает его приложению Flask.
* Flask определяет маршрут (route), соответствующий запрошенному URL и методу (GET, POST и т.д.).
* Если маршрут найден, вызывается привязанная к нему функция (view function).
* Эта функция обрабатывает запрос (например, получает параметры из request, выполняет запросы к базе данных и т.п.) и возвращает ответ.
* Ответ может быть текстом, HTML-страницей, JSON-данными или перенаправлением на другой URL.

**2. Что такое url\_for() и зачем он нужен в Flask? (4 балла)**

Функция url\_for() генерирует URL-адрес для указанной функции по ее имени.  
Это удобно, так как позволяет не прописывать URL-адрес вручную, а автоматически учитывать изменения маршрутов.

**Пример:**

from flask import Flask, url\_for

app = Flask(\_\_name\_\_)

@app.route('/home')

def home():

    return 'Главная страница'

# Использование url\_for

@app.route('/redirect')

def go\_home():

    return redirect(url\_for('home'))

**3. Что такое объект request в Flask и какие данные он содержит? (4 балла)**

Объект request предоставляет доступ к данным текущего HTTP-запроса. Он содержит:

* **request.method** — метод запроса (например, GET или POST).
* **request.args** — параметры URL-запроса.
* **request.form** — данные форм, отправленных методом POST.
* **request.files** — загруженные файлы.
* **request.headers** — заголовки запроса.
* **request.cookies** — куки-файлы.

**Пример:**

from flask import Flask, request

app = Flask(\_\_name\_\_)

@app.route('/data', methods=['GET', 'POST'])

def data():

    if request.method == 'POST':

        return f"Получены данные: {request.form['name']}"

    return "Ожидается POST-запрос"

**4. Что такое flash() в Flask и как оно используется? (4 балла)**

Функция flash() используется для отправки временных уведомлений пользователям. Эти уведомления можно вывести в шаблоне с помощью get\_flashed\_messages().

**Пример:**

from flask import Flask, flash, render\_template

app = Flask(\_\_name\_\_)

app.secret\_key = 'supersecretkey'

@app.route('/login', methods=['POST'])

def login():

    flash("Успешный вход в систему")

    return render\_template("index.html")

**5. Как в Flask реализуется редирект и когда он используется? (5 баллов)**

Редирект в Flask осуществляется с помощью функции redirect(). Он применяется для перенаправления пользователя на другой маршрут, например, после успешной отправки данных или при попытке доступа к устаревшему URL.

**Пример:**

from flask import Flask, redirect

app = Flask(\_\_name\_\_)

@app.route('/old')

def old():

    return redirect('/new')

@app.route('/new')

def new():

    return "Это новая страница"

**6. Что такое request.method и как с ним работать? (5 баллов)**

Атрибут request.method возвращает HTTP-метод текущего запроса (GET, POST, PUT, DELETE и т.д.). Он часто используется для выполнения различных действий в зависимости от типа запроса.

**Пример:**

from flask import Flask, request

app = Flask(\_\_name\_\_)

@app.route('/', methods=['GET', 'POST'])

def home():

    if request.method == 'POST':

        return "Получен POST-запрос"

    return "Получен GET-запрос"

**7. Как в Flask реализуется защита от CSRF-атак? (5 баллов)**

Для защиты от CSRF-атак в Flask используется расширение **Flask-WTF**, которое добавляет CSRF-токен в формы. Этот токен проверяется при отправке данных, предотвращая несанкционированные запросы.

**Пример настройки:**

from flask\_wtf import FlaskForm

from wtforms import StringField, SubmitField

from wtforms.validators import DataRequired

class MyForm(FlaskForm):

    name = StringField('Имя', validators=[DataRequired()])

    submit = SubmitField('Отправить')

**Пример шаблона с CSRF-токеном:**

<form method="POST">

  {{ form.hidden\_tag() }}

  {{ form.name.label }} {{ form.name() }}

  {{ form.submit() }}

</form>

**8. Как в Flask настроить статические файлы (CSS, JS, изображения)? (5 баллов)**

Flask автоматически обслуживает статические файлы из папки /static/.  
Файлы можно подключать в шаблонах с помощью url\_for().

**Пример структуры проекта:**

/static

    /css

        style.css

/templates

    index.html

app.py

**Пример подключения CSS в шаблоне:**

<head>

  <link rel="stylesheet" href="{{ url\_for('static', filename='css/style.css') }}">

</head>

**Часть С.**

**Пример реализации практических заданий:**

*from* flask *import* Flask, render\_template, request, flash

*import* sqlite3

app = Flask(\_\_name\_\_)

app.secret\_key = 'supersecretkey'

*# Подключение к базе данных SQLite*

def **get\_db\_connection**():

    conn = sqlite3.connect('database.db')

    conn.row\_factory = sqlite3.Row

*return* conn

*# Маршрут для отображения детальной информации о товаре*

@app.route('/product/<int:id>')

def **product\_detail**(*id*):

    conn = get\_db\_connection()

    product = conn.execute('SELECT \* FROM products WHERE id = ?', (*id*,)).fetchone()

    conn.close()

*return* render\_template('product\_detail.html', *product*=product)

*# Маршрут для отображения списка всех товаров*

@app.route('/products')

def **products**():

    conn = get\_db\_connection()

    products = conn.execute('SELECT \* FROM products').fetchall()

    conn.close()

*return* render\_template('products.html', *products*=products)

*# Пример уведомления с использованием flash*

@app.route('/add\_product', *methods*=['POST'])

def **add\_product**():

    flash('Товар успешно добавлен!')

*return* redirect('/products')

*if* \_\_name\_\_ == "\_\_main\_\_":

    app.run(*debug*=True)