

Содержание

Контрольная работа «Разработка динамических страниц на ЯП Python».....	3
I. Настройка локального сервера	3
1.1 Создание проекта	4
1.2 Создание Python файла.....	7
II. Написание и отладка CGI-скрипта.....	10
III Написание и отладка CGI-скриптов: получение данных.....	13
3.1 Создание html-файла (формы).....	14
3.2 Создание Python файла.....	18

Контрольная работа «Разработка динамических страниц на ЯП Python».

Цель работы: создание динамических страниц на языке высокого уровня Python при помощи CGI-скриптов.

CGI-скрипты - это исполняемые файлы, которые выполняются веб-сервером, когда в URL запрашивается соответствующий скрипт.

Методика выполнения работы включает следующие этапы:

1. Настройка локального сервера.
2. Написание и отладка CGI-скриптов.
3. Написание и отладка CGI-скриптов: получение данных

I. Настройка локального сервера

Выполним задание в программной среде PyCharm Community (рисунок 1.1).



Рисунок 1.1 – IDE PyCharm Community

1.1 Создание проекта

Выберем в меню File -> New Project (рисунок 1.2 – 1.3).

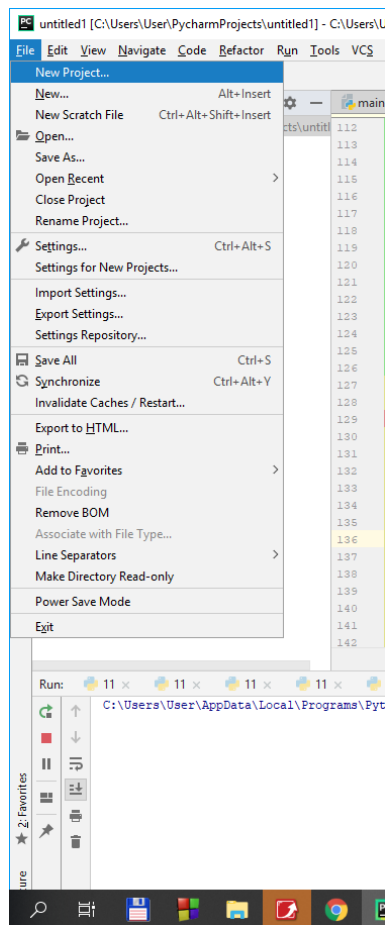


Рисунок 1.2 – New Project

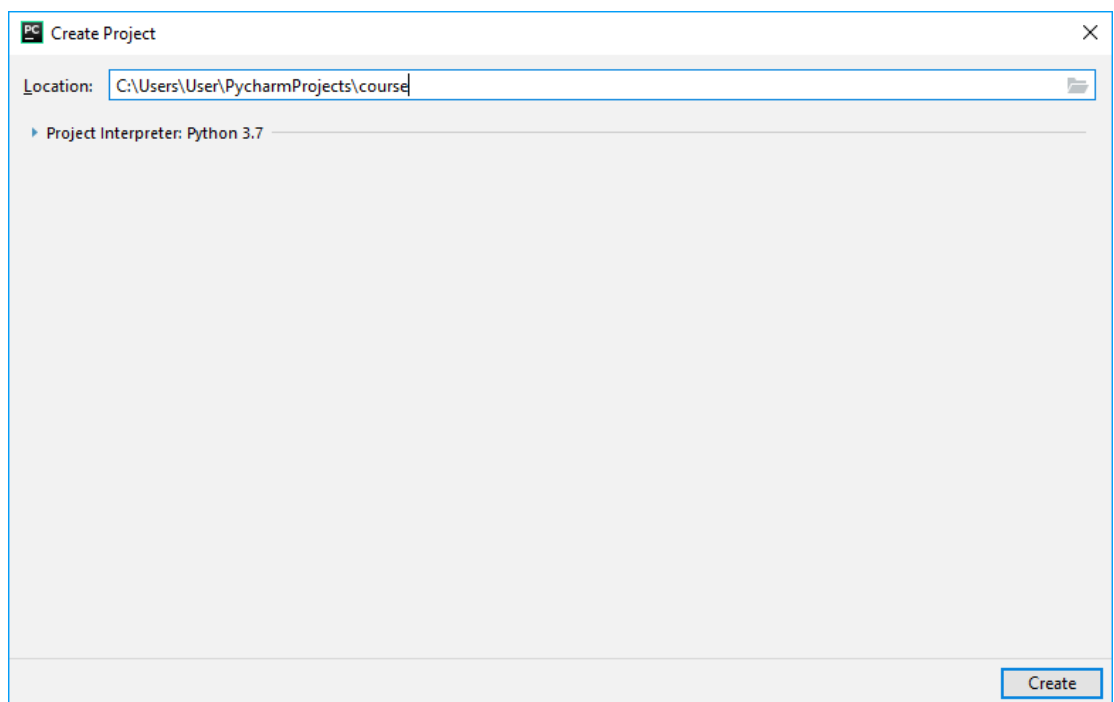


Рисунок 1.3 – Ввод имени проекта *Course*

Далее нажмем Create. Далее выводится диалоговое окно (рисунок 1.4).

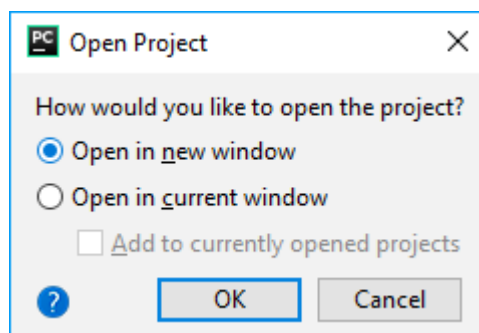


Рисунок 1.4 – Диалоговое окно отображения проекта (новое окно / существующее окно)

Выберем вариант расположения окна – новое.

Далее получим (рисунок 1.5).

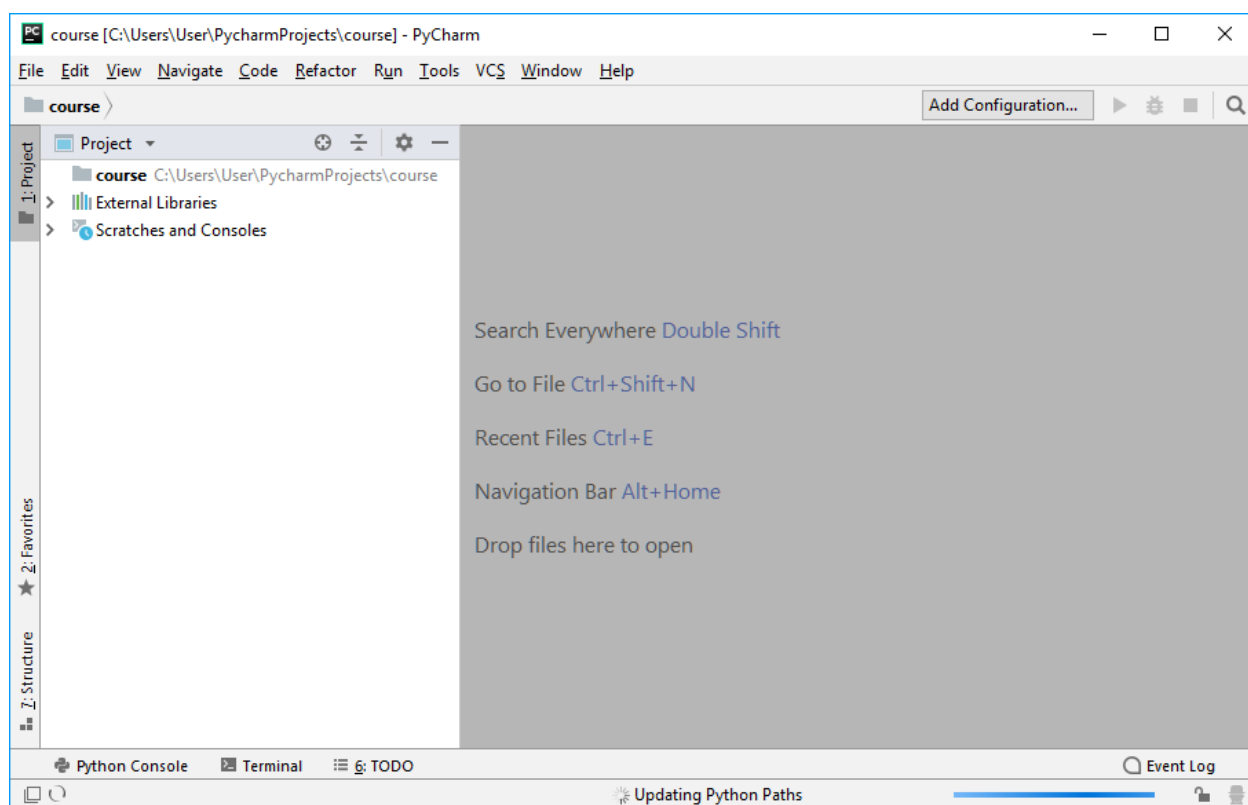


Рисунок 1.5 – IDE PyCharm Community

Проект создан.

Когда проект открывается, Вы видите главное окно, разделенное на несколько логических областей.

Project Tool Window. Панель инструментов проекта. В этом окне отображаются файлы вашего проекта.

PyCharm Editor. Редактор PyCharm. Находится с правой стороны, где вы пишете свой код. В нем есть вкладки для удобной навигации между открытыми файлами.

Navigation Bar. Панель навигации. Находится над редактором, позволяет быстро запускать и отлаживать ваше приложение, а также выполнять процедуры контроля версий VCS.

Left gutter. Левый столбец, вертикальная полоса рядом с редактором, показывает брекпоинты и обеспечивает удобный способ перехода по иерархии кода. Он также отображает номера строк.

Right gutter. Правый столбец, справа от редактора. PyCharm постоянно контролирует качество вашего кода и постоянно показывает результаты проверки в правом столбце: ошибки, предупреждения и т.д. Индикатор в правом верхнем углу показывает общий статус проверки кода для всего файла.

PyCharm Tool Windows. Панель инструментов PyCharm. Это специальные окна, прикрепленные к низу и сторонам рабочей области, которые обеспечивают доступ к типичным задачам, таким как управление проектами, поиск и навигация по исходному коду, интеграция с системами контроля версий и т.д.

Status Bar. Строка состояния. Указывает состояние вашего проекта и показывает различные предупреждения и информационные сообщения.

Кроме того, в нижнем левом углу окна PyCharm в строке состояния вы увидите кнопку. Эта кнопка переключает показ панелей инструментов. Если вы наведете указатель мыши на эту кнопку, появится список доступных в данный момент панелей (рисунок 1.6):

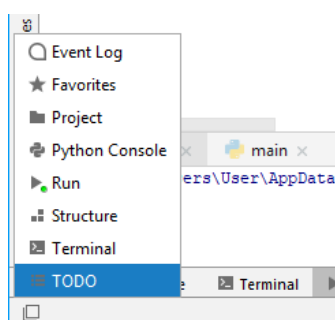


Рисунок 1.6 – Переключение между панелями IDE PyCharm Community

1.2 Создание Python файла.

В Python присутствует встроенный CGI сервер. Настроим его с помощью Python файла.

Для этого выберем в меню File->New->PythonFile (рисунок 1.7) и введем имя файла **main** (рисунок 1.8). В результате получим (рисунок 1.9).

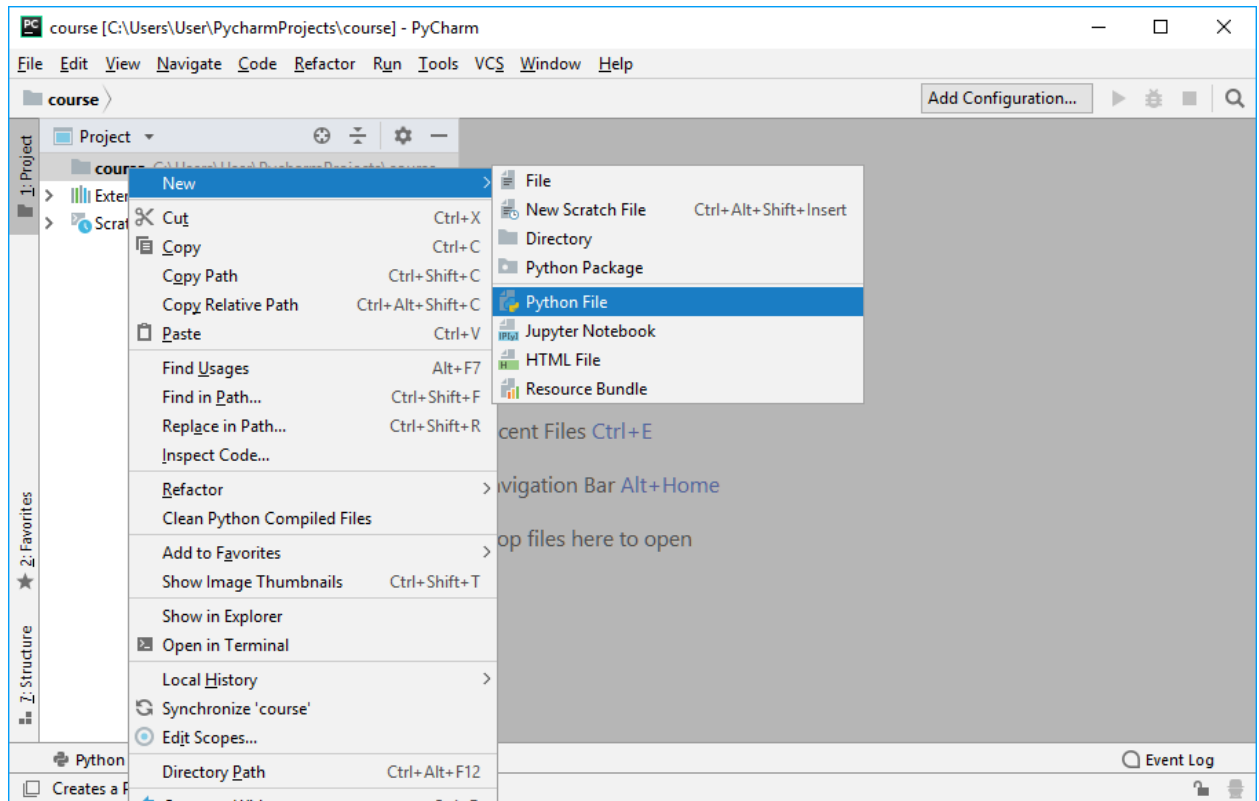


Рисунок 1.7 – File->New->PythonFile

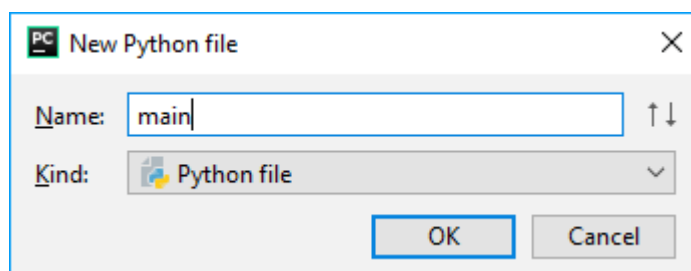


Рисунок 1.8 - Ввод имени файла *main*

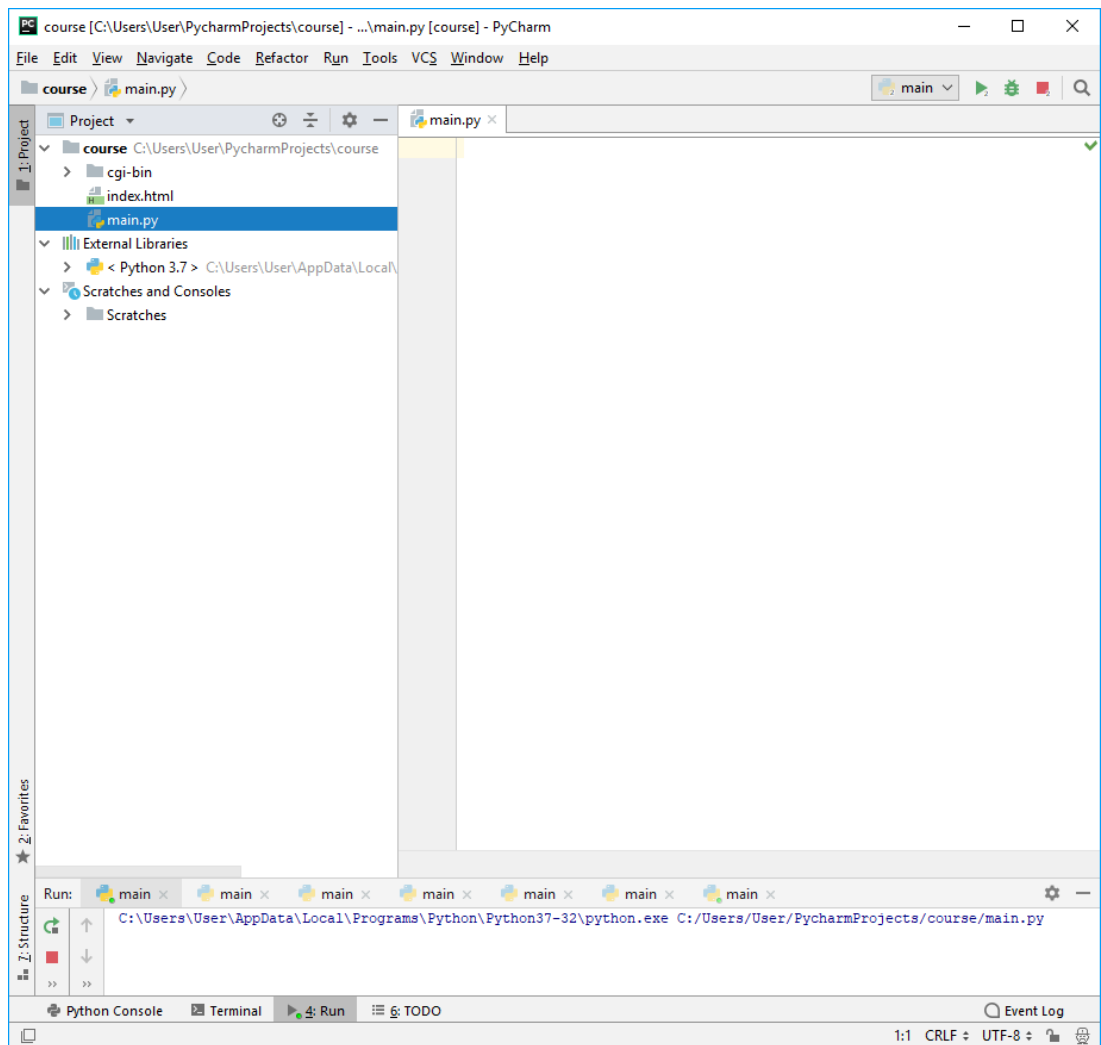


Рисунок 1.9 – Результат создания файла *main*

Для настройки локального сервера введем следующие строки кода (рисунок 1.10):

```
from http.server import HTTPServer, CGIHTTPRequestHandler
server_address = ("", 8000)
httpd = HTTPServer(server_address, CGIHTTPRequestHandler)
httpd.serve_forever()
```

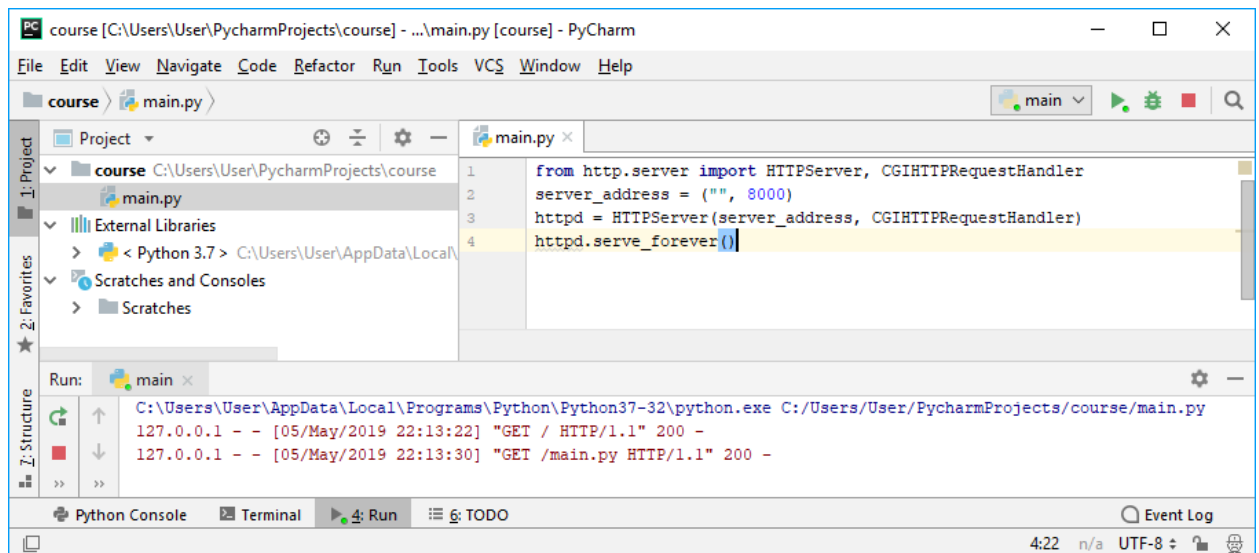


Рисунок 1.10 – Файл *main.py*

Далее откроем браузер и в адресной строке введем:

localhost:8000

Получим (рисунок 1.11):

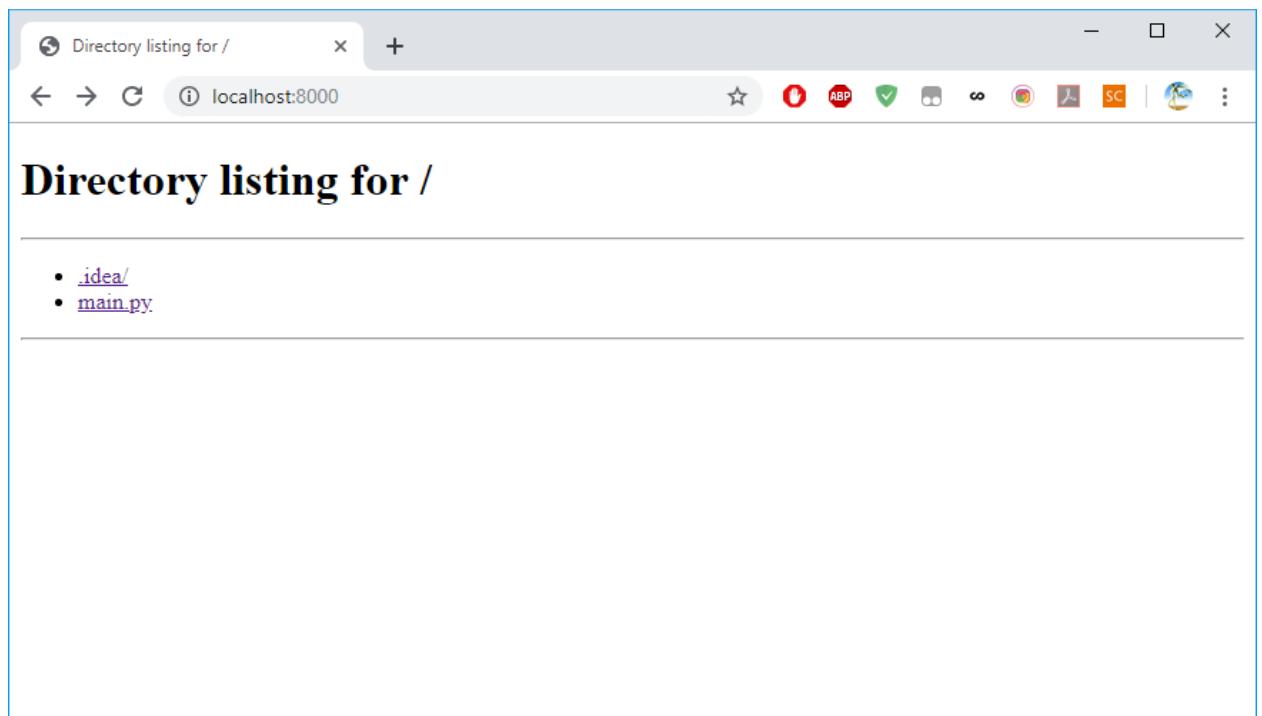


Рисунок 1.11 – Результат настройки сервера

II. Написание и отладка CGI-скрипта.

Напишем приветственные слова «Hello world!».

Для этого создадим директорию *cgi-bin* путем выбора в меню File->New->Directory (рисунок 2.1) и ввода имени директории (рисунок 2.2).

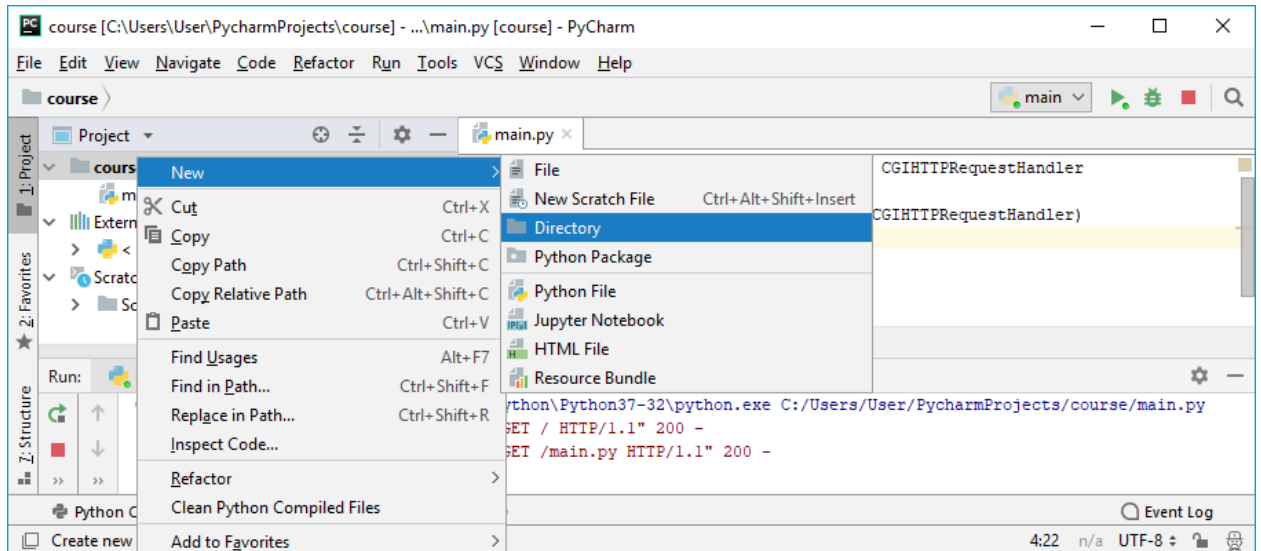


Рисунок 2.1 – File->New->Directory

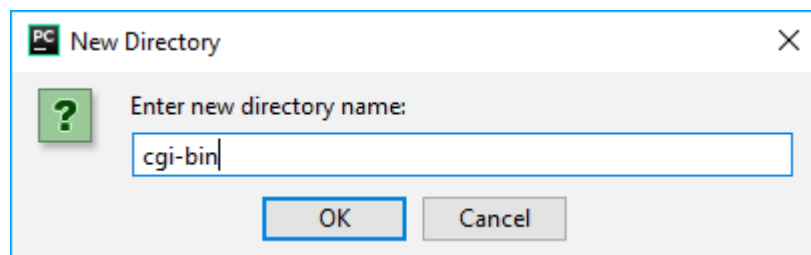


Рисунок 2.2 - Ввод имени директории *cgi-bin*

В директории *cgi-bin* создадим Python файл *test.py* (рисунок 2.3 – 2.4).

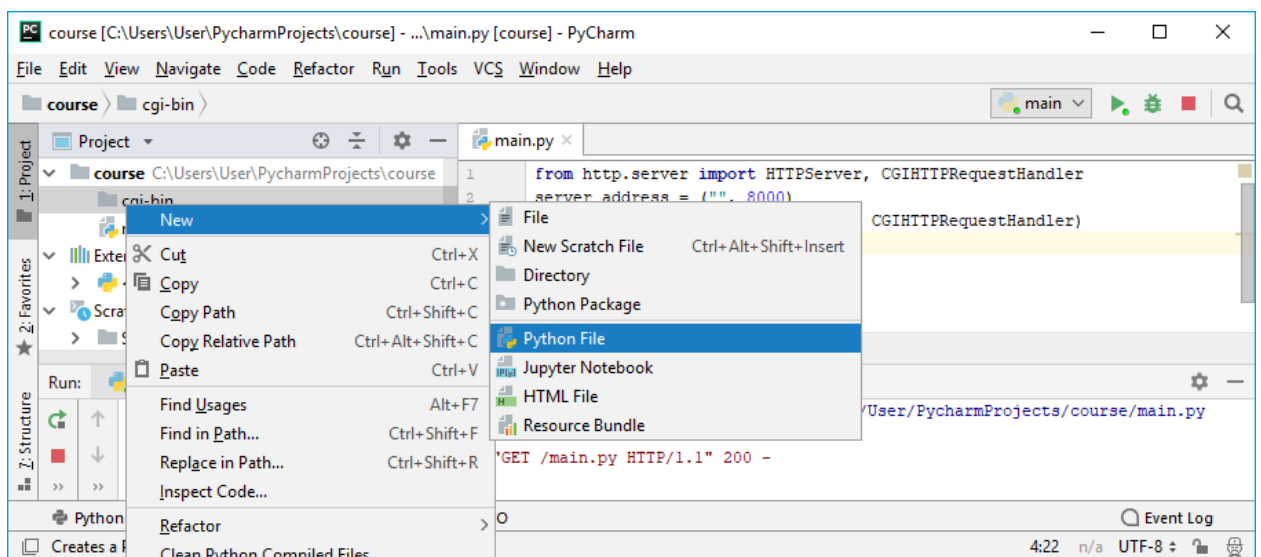


Рисунок 2.3 – File->New->PythonFile

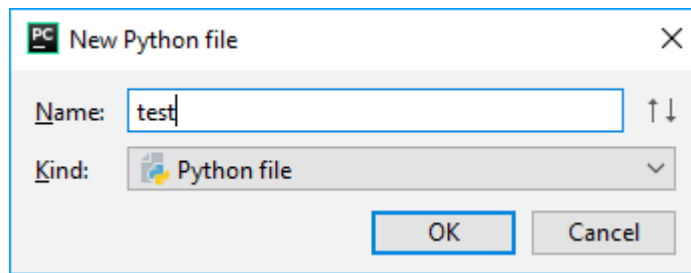


Рисунок 2.4 - Ввод имени файла *test*

Пропишем в файле test.py следующие строки кода (рисунок 2.5).

```
#!/usr/bin/env python3
print("Content-type: text/html")
print()
print("<h1>Hello world!</h1>")
```

***Примечание:** Первая строка говорит о том, что это Python скрипт (CGI-скрипты можно не только на Python писать).*

Вторая строка печатает заголовок. Он обозначает, что это будет html файл.

Третья строка (просто символ новой строки) отделяет заголовки от тела ответа.

Четвёртая печатает Hello world.

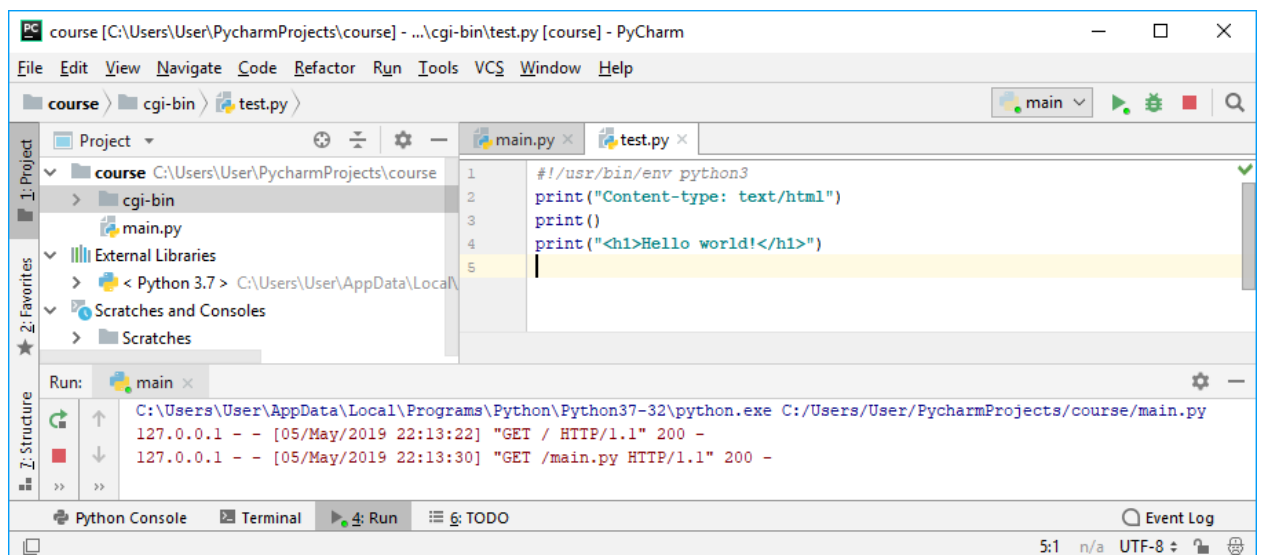


Рисунок 2.5 - Содержимое файла *test.py*

Далее откроем браузер и в адресной строке введем:

localhost:8000/cgi-bin/test.py

Получим (рисунок 2.6):

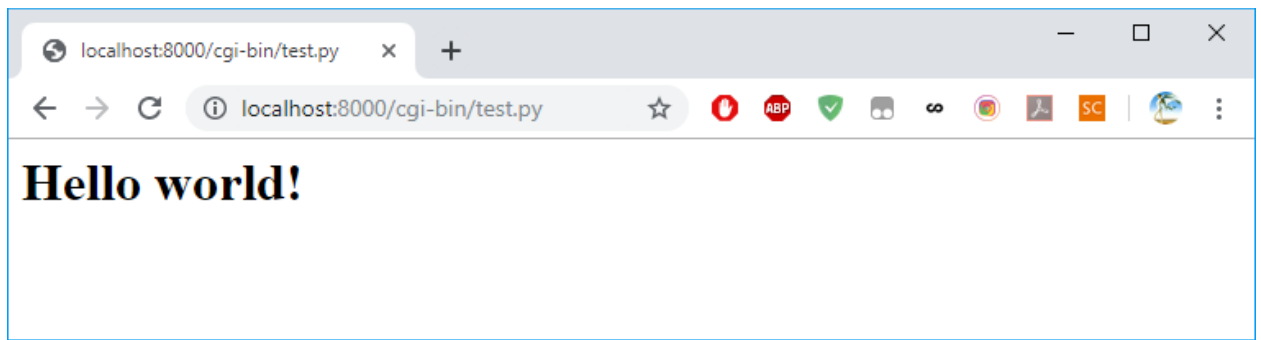


Рисунок 2.6 – Результат работы

III Написание и отладка CGI-скриптов: получение данных

В модуле CGI присутствует класс `FieldStorage`, который содержит в себе переданную в форме информацию (более подробная информация <https://docs.python.org/2/library/cgi.html>).

Класс `FieldStorage` имеет два метода получения значений данных формы: для одного и нескольких значений.

`FieldStorage.getfirst(name, default=None)` - всегда возвращает только одно (первое) значение, связанное с именем поля формы. Обратите внимание, что порядок, в котором будут получены значения, могут отличаться от браузера к браузеру. Если нет такого поля формы или значение не существует, то метод возвращает `default`.

`FieldStorage.getlist(name)` - возвращает список значений, связанных с именем поля формы.

3.1 Создание html-файла (формы)

Создадим в директории *cgi-bin* файл *index.html* (рисунок 3.1).

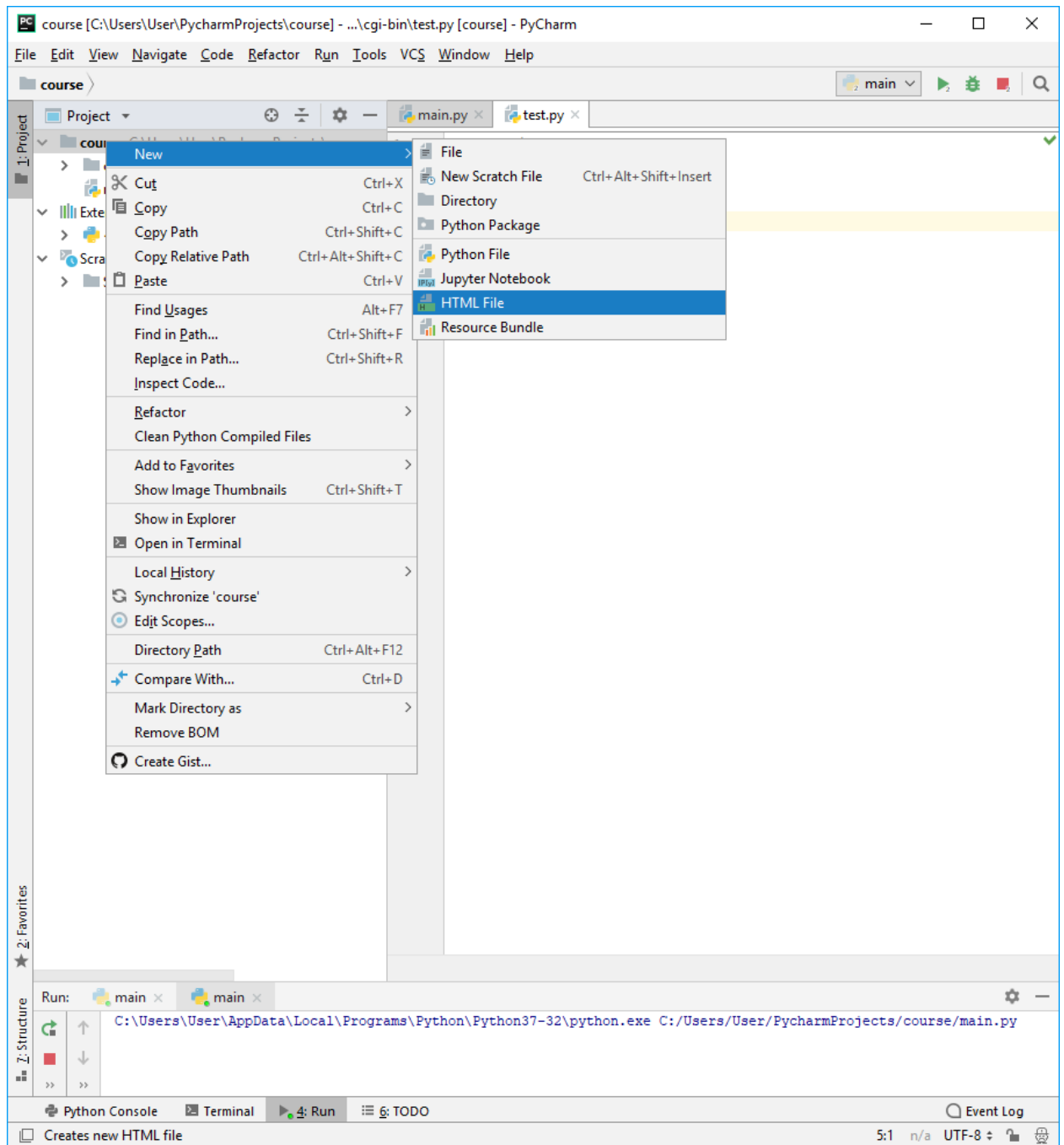


Рисунок 3.1 –New->HTML File

Введем имя файла *index* в следующем диалоговом окне (рисунок 3.2).

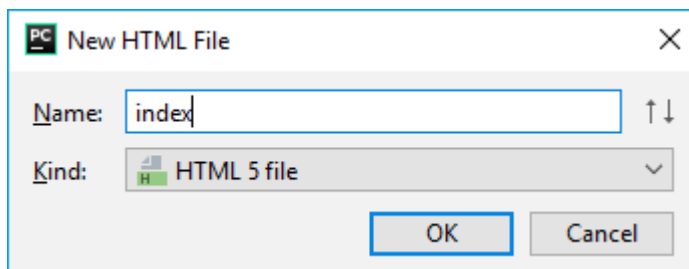


Рисунок 3.2 – Ввод имени html-файла *index*

Пропишем в файле *index.html* следующие строки кода (рисунок 3.3).

```
<<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
  <form action="/cgi-bin/course_form.py">
    <p><b>Персональная информация студента</b></p> <br>
    <p>Введите фамилию:</p>
    <input type="text" name="Surname"> <br><br>
    <p>Введите имя:</p>
    <input type="text" name="Name"> <br><br>
    <p>Введите отчество:</p>
    <input type="text" name="Patronymic"> <br><br>
    <p>Введите группу</p>
    <input type="text" name="Group"> <br><br>
    <p>Предложения / замечания по курсу дисциплины ЯП</p>

    <textarea name="Comment" cols="40" rows="3"></textarea></p>
    <p><input type="submit" value="Отправить">
    <input type="reset" value="Очистить"></p>
  </form>
</body>
</html>
```

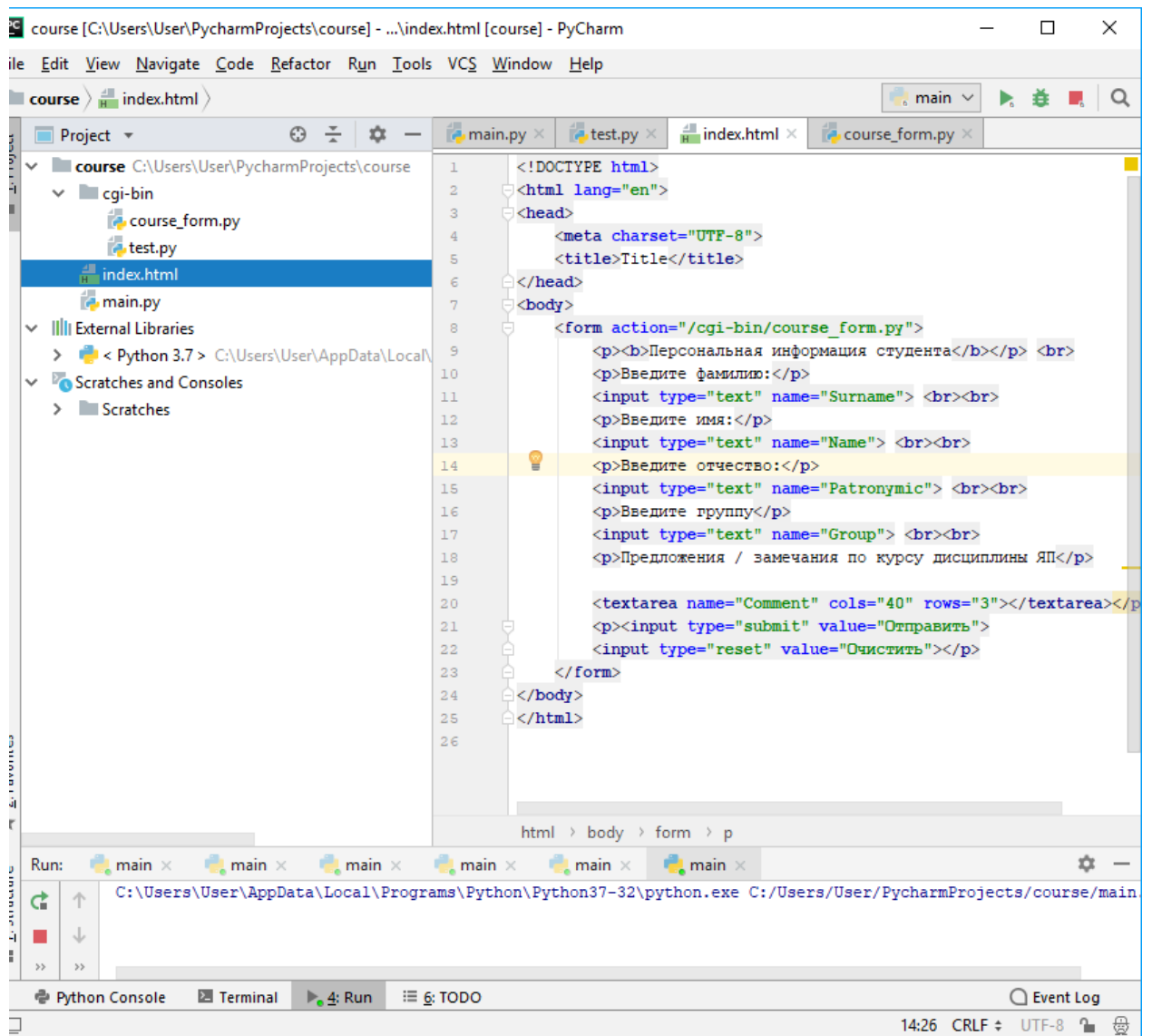
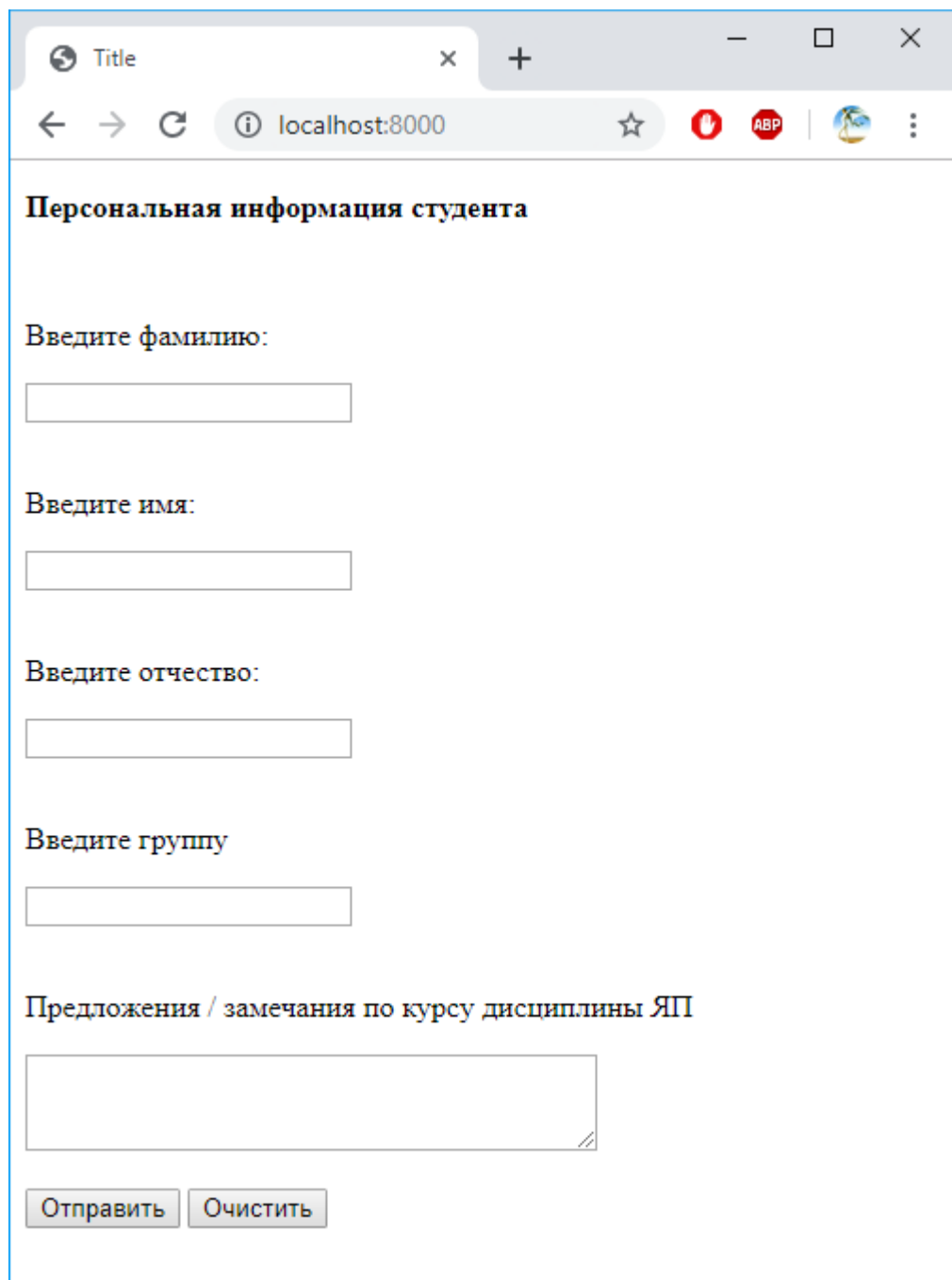


Рисунок 3.2 – Содержимое файла *index.html*

Запускаем локальный сервер, и переходим на **localhost:8000**. Получим (рисунок 3.3):



The image shows a web browser window with a single tab titled "Title". The address bar displays "localhost:8000". The page content is a form titled "Персональная информация студента". It contains four text input fields for "Введите фамилию:", "Введите имя:", "Введите отчество:", and "Введите группу". Below these is a larger text area for "Предложения / замечания по курсу дисциплины ЯП". At the bottom are two buttons: "Отправить" and "Очистить".

Персональная информация студента

Введите фамилию:

Введите имя:

Введите отчество:

Введите группу

Предложения / замечания по курсу дисциплины ЯП

Рисунок 3.3 – Результат работы

Далее создадим Python файл для обработки введенной информации.

3.2 Создание Python файла

Создадим в директории *cgi-bin* файл *course_form.py* (рисунок 3.4).

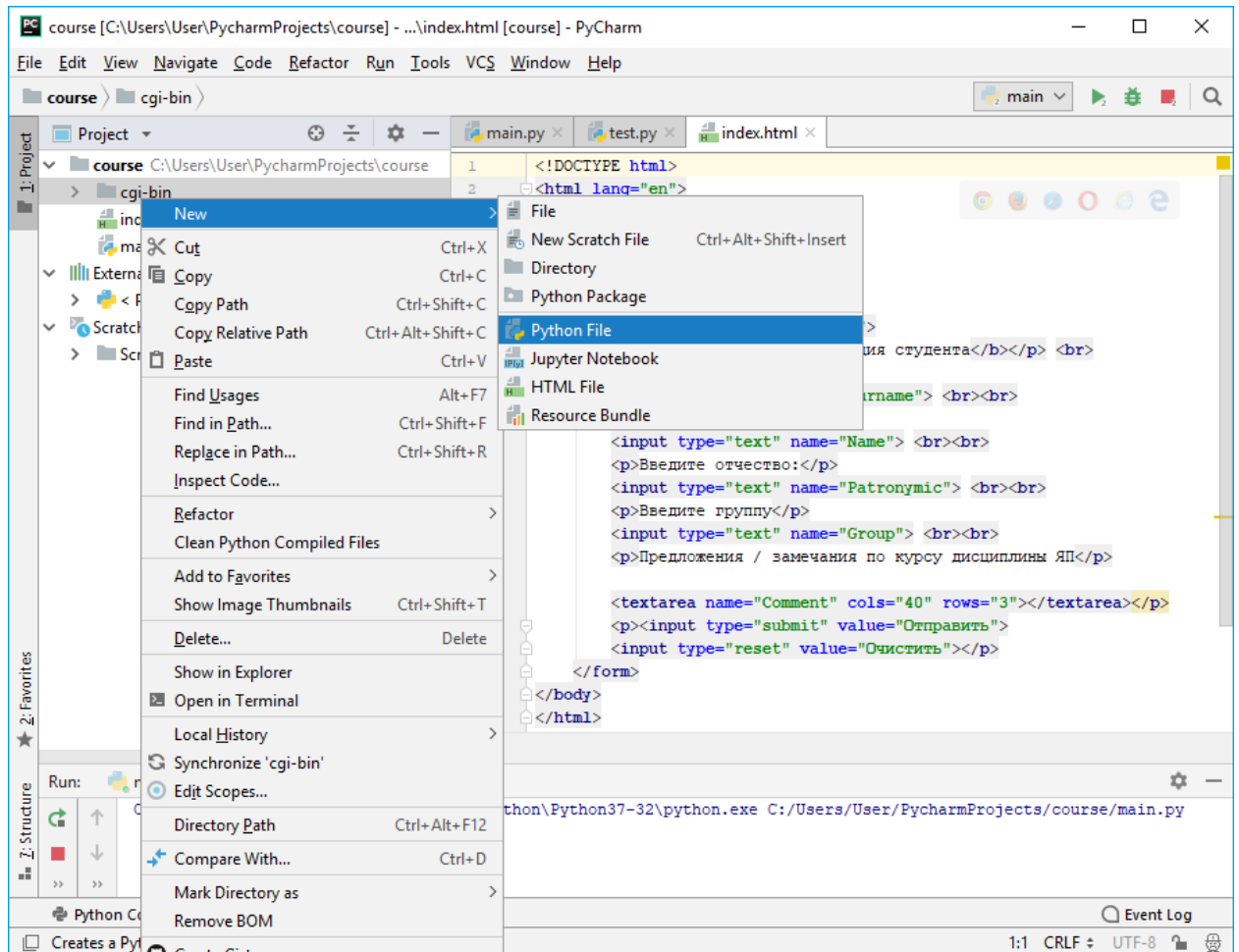


Рисунок 3.4 –New->Python File

Введем имя файла *course_form* в следующем диалоговом окне (рисунок 3.5).

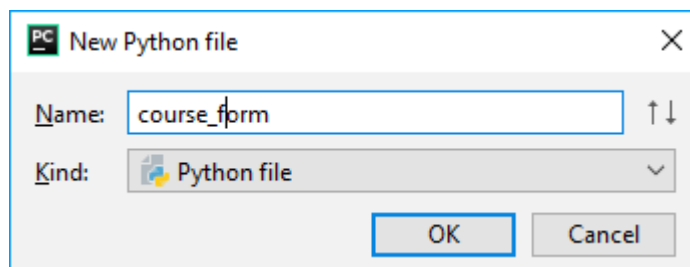


Рисунок 3.5 – Ввод имени Python-файла *course_form*

Пропишем в файле *course_form.py* следующие строки кода (рисунок 3.6).

```

#!/usr/bin/env python3
import cgi
import html

course_form = cgi.FieldStorage()

Surname = course_form.getfirst("Surname", "не задано")
Name = course_form.getfirst("Name", "не задано")
Patronymic = course_form.getfirst("Patronymic", "не задано")
Group = course_form.getfirst("Group", "не задано")
Comment = course_form.getfirst("Comment", "не задано")

Surname = html.escape(Surname)
Name = html.escape(Name)
Patronymic = html.escape(Patronymic)
Group = html.escape(Group)

print("Content-type: text/html\n")
print("""<!DOCTYPE HTML>
<html>
<head>
    <meta charset="utf-8">
    <title>Обработка данных форм</title>
</head>
<body>""")

print("<h1>Обработка персональных данных!</h1>")
print("<h1>Успешное выполнение контрольной работы!</h1>")
print("<p>Фамилия: {}</p>".format(Surname))
print("<p>Имя: {}</p>".format(Name))
print("<p>Отчество: {}</p>".format(Patronymic))
print("<p>Группа: {}</p>".format(Group))
print("<p>Ваши предложения/замечания: {}</p>".format(Comment))
print("""</body>
</html>""")

```

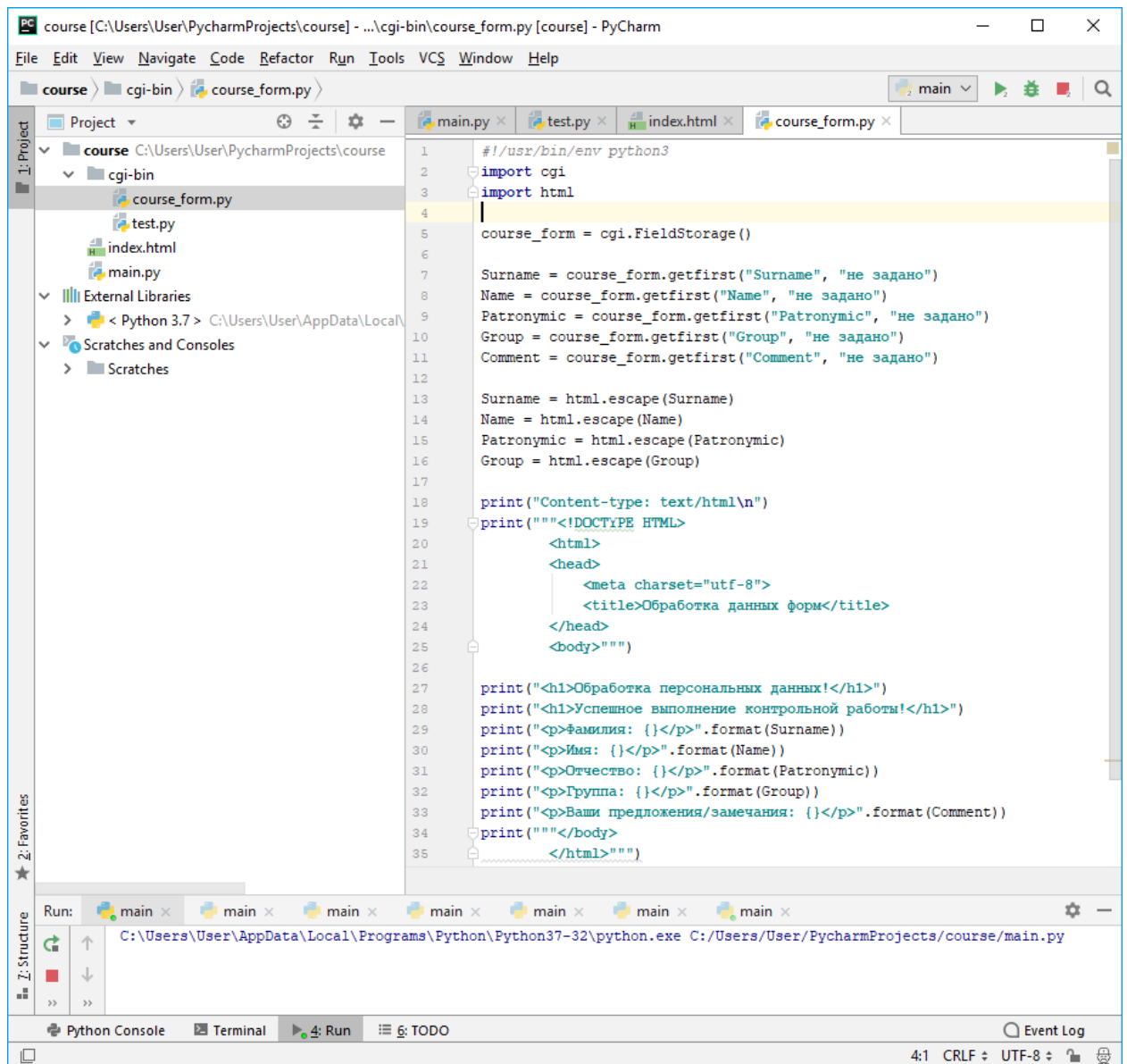
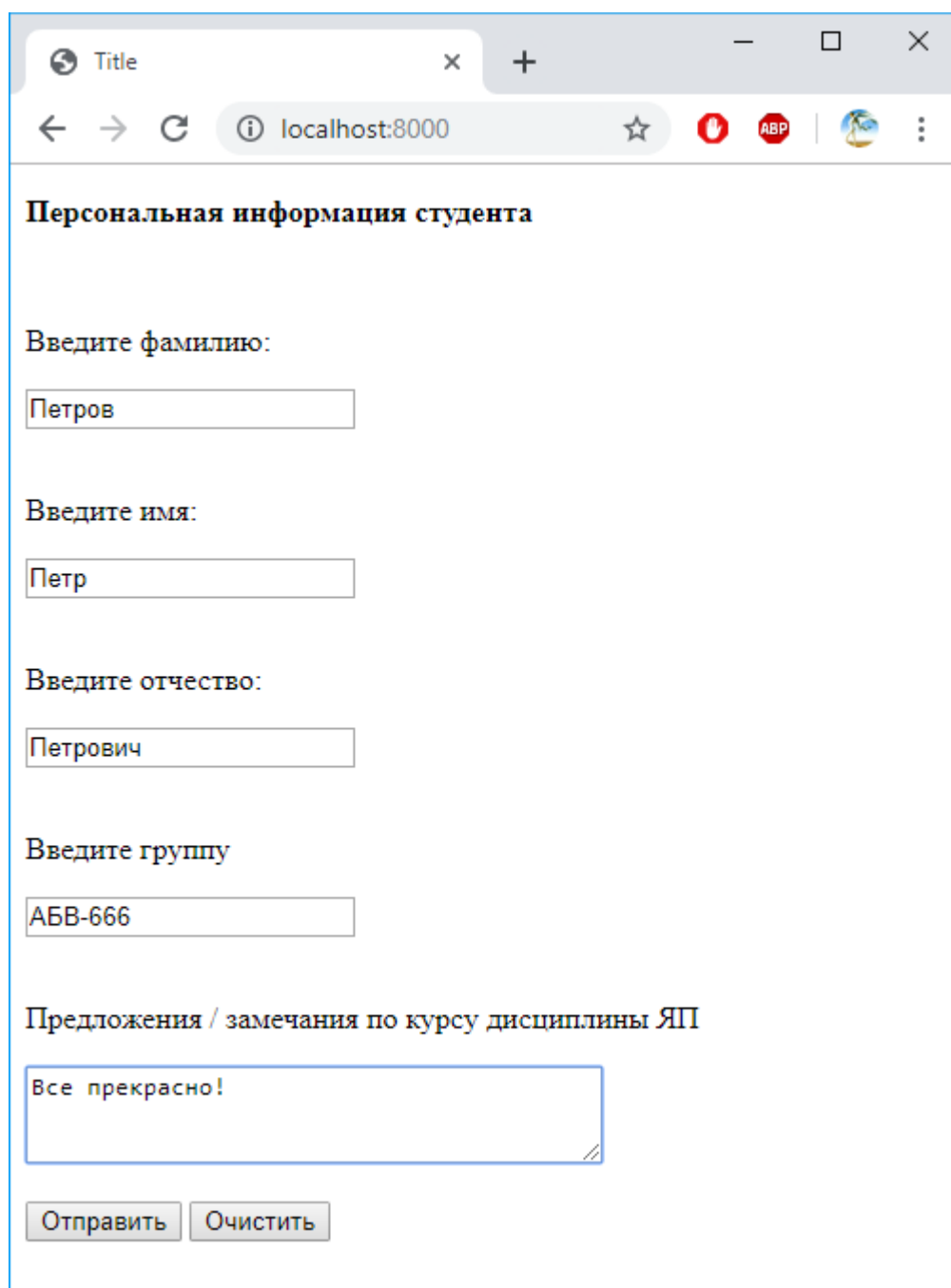


Рисунок 3.6 – Содержимое файла *course_form.pyl*

Запускаем локальный сервер, и переходим на **localhost:8000**.

Введем пользовательскую информацию (рисунок 3.7) и нажмем на кнопку «Отправить». Получим (рисунок 3.8):



The image shows a web browser window with a single tab titled "Title". The address bar displays "localhost:8000". The page content is as follows:

Персональная информация студента

Введите фамилию:

Введите имя:

Введите отчество:

Введите группу

Предложения / замечания по курсу дисциплины ЯП

Рисунок 3.7 – Ввод пользовательской информации

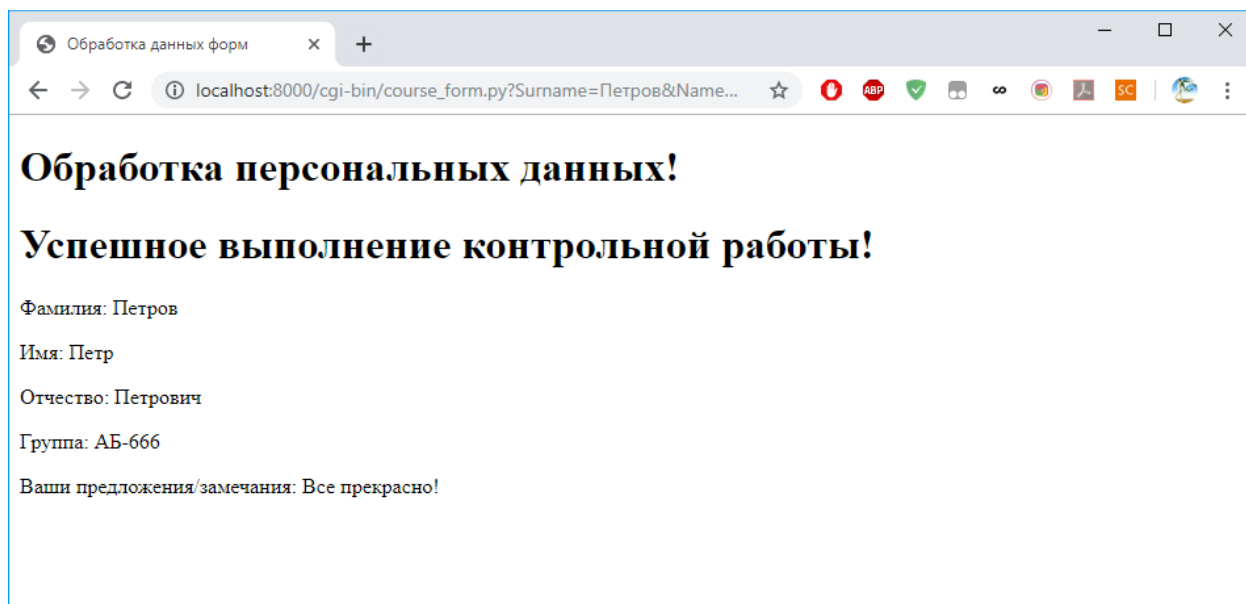


Рисунок 3.8 – Результат работы