|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ЦИФРОВАЯ КАФЕДРА 01

КУРС «WEB-РАЗРАБОТЧИК»

**РАСЧЁТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КОМАНДНОМУ ПРОЕКТУ***

***НА ТЕМУ:***

***«РАЗРАБОТКА МНОГОПОЛЬЗОВАТЕЛЬСКОГО  
ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ РЕШЕНИЯ ЗАДАЧ СОПРОТИВЛЕНИЯ МАТЕРИАЛОВ»***

Участники проекта:

Студент ФН2-41Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  И.П. Шаманов

(Группа) (Подпись, дата) (И.О.Фамилия)

Студент СМ13-41Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** И.А. Иванов

(Группа) (Подпись, дата) (И.О.Фамилия)

СОДЕРЖАНИЕ

[1. Техническое задание на реализацию проекта по курсу веб-разработки 3](#_Toc140594983)

[1.1. Цель 3](#_Toc140594984)

[1.2. Задачи 3](#_Toc140594985)

[1.3. Требования к оформлению пояснительной записки 3](#_Toc140594986)

[1.4. Организационные требования 4](#_Toc140594987)

[1.5. Технические требования 4](#_Toc140594988)

[1.6. Требования к наименованию переменных 6](#_Toc140594989)

[1.7. Требования к публикации и развертыванию веб-приложения 6](#_Toc140594990)

[2 Сервис «Балка Онлайн» 7](#_Toc140594991)

[3 Создание веб-приложения для проекта «Балка Онлайн» 8](#_Toc140594992)

[3.1. Техническое задание 8](#_Toc140594993)

[3.2. Общие требования 8](#_Toc140594994)

[3.3. Главная страница веб-приложения 8](#_Toc140594995)

[3.4. Калькулятор 9](#_Toc140594996)

[3.5. Административное управление через Django (со стороны администрации сайта) 9](#_Toc140594997)

[3.6. База данных 9](#_Toc140594998)

[4. Структура и управление веб-приложением 10](#_Toc140594999)

[5. Работа веб-приложения 13](#_Toc140595000)

[5.1. Главная страница 13](#_Toc140595001)

[5.2. Калькулятор 14](#_Toc140595002)

[5.3. Создание учётной записи 16](#_Toc140595003)

[5.4. Форум 17](#_Toc140595004)

[5.5. Страница администратора сайта 18](#_Toc140595005)

[6. Программная реализация компонентов 19](#_Toc140595006)

[6.1. Модуль расчётов 19](#_Toc140595007)

[6.2. Первоначальная реализация веб-интерфейса раздела «Калькулятор» 25](#_Toc140595011)

1. Техническое задание на реализацию проекта по   
курсу веб-разработки

## Цель

• Разработка веб-приложения с использованием фреймворка Django.

## Задачи

* Проектирование и прототипирование веб-приложения с использованием BPMN – диаграмм и дизайн-макетов (Можно использовать Figma / FigJam / Miro);
* Разработка серверной логики и моделей данных с использованием Django;
* Разработка шаблонов отображения данных с помощью HTML / CSS / JS;
* Разработка сопроводительной документации по установке и запуску
* веб-приложения;
* Публикация и развертывание веб-приложения;
* Оформление пояснительной записки к проекту.

## Требования к оформлению пояснительной записки

• Титульный лист с наименованием проекта, ФИО исполнителя;

• Содержание;

• Нумерация страниц;

• Шрифт: Times New Roman, 14 пт;

• Выравнивание: по ширине страницы;

• Межстрочный интервал: 1,5;

• Вставки исходного кода оформляются в виде листингов, с подписями;

• Блок-схемы и изображения оформляются в виде рисунков, с подписями.

## Организационные требования

* Проект выполняется в командах по 4-5 человек;
* Командное взаимодействие происходит через Gitlab, развернутый на сервере МГТУ;
* Вклад каждого студента в проект оценивается по коммитам;

## Технические требования

• Проект реализуется с помощью фреймворка Django, версия 4.2;

• UI-компоненты выполнены с помощью фреймворка Bootstrap / Material / Taiga;

• Проект многопользовательский, поддерживается регистрация и авторизация пользователей, реализована выдача уникального контента для каждого пользователя

• Реализовано меню, которое позволяет перемещаться между разделами веб-приложения;

• В проекте используется минимум 3 связанных между собой сущности (например, факультет-группа-студент):

• Связь сущностей должна быть не только логической, но и физической, на уровне модели данных;

• Реализовано минимум 3 компонента:

• Минимум 1 из компонентов отображает данные в виде списка;

• В проекте используются формы;

• Реализован CRUD подход:

• Для GET методов реализована передача параметров через query;

• Для PUT и DELETE методов реализована передача параметров через URL;

• Для POST методов реализована передача параметров через form-data.

• В случае, если запрашиваемой страницы не существует, пользователь должен быть перенаправлен на страницу с ошибкой 404;

• Реализована проверка входных параметров на корректность на стороне серверной логики. В случае, если входные параметры некорректны, пользователь должен получать осмысленное описание ошибки;

• Используется валидация данных в формах;

• Реализован процесс авторизации / регистрации с использованием нескольких пользовательских групп:

• Для неавторизованного пользователя часть функционала, связанная с добавлением, обновлением, удалением информации, должна быть недоступна.

• В части HTML / CSS кода:

• Не используется атрибут style;

• Не используется !important;

• Идентификаторы используются строго по своему назначению;

• В классах, в случае необходимости, реализован подход наследования;

• Используется селектор лоботомированной совы;

• Форматирование исходного кода выполнено с использованием отступов;

• Исходный код проходит проверку линтерами PYLINT-DJANGO и DJLINT;

• При развертывании приложения режим отладки должен быть деактивирован;

• В консоли веб-браузера не должны отображаться ошибки, вызванные работой веб-приложения.

## Требования к наименованию переменных

* Наименования переменных должны коротко и точно отражать их суть;
* Наименования переменных не должны быть написаны с использованием транслитерации;
* Наименования переменных не должны содержать орфографических ошибок;
* Наименования переменных должны быть оформлены в едином стиле.

## Требования к публикации и развертыванию веб-приложения

* Приложение должно разрабатываться с использованием системы
* контроля версий:
  + - Наименование коммитов должны отражать суть внесенных
* изменений;
* Репозиторий с приложением должен быть опубликован и иметь
* публичный режим доступа к моменту проверки работы;
* Приложение, развернутое на сервере, не должно содержать каталог .git и
* другие системные файлы питона или вспомогательных утилит;
* Необходимо реализовать CI \ CD процесс, который будет автоматически

публиковать приложение на сервере при добавлении коммита в master-ветку.

# 2 Сервис «Балка Онлайн»

В процессе обучения мы не раз сталкивались с отсутствием удобных приложений-калькуляторов, позволяющих проводить необходимые инженерные расчеты. Одни были достаточно трудны в освоении, вторые имели неудобный интерфейс, мешающий корректной работе, третьи предлагали оформить платную подписку. Поэтому мы решили разработать свой инженерный модуль по расчету оптимальных параметров механических систем.

Такой проект был выбран не случайно, сопротивление материалов – важная инженерная дисциплина, требующая большого количества расчетов. В этом случае веб-интерфейс будет очень выгоден, так как пользователю не потребуется установка дополнительных ресурсов.

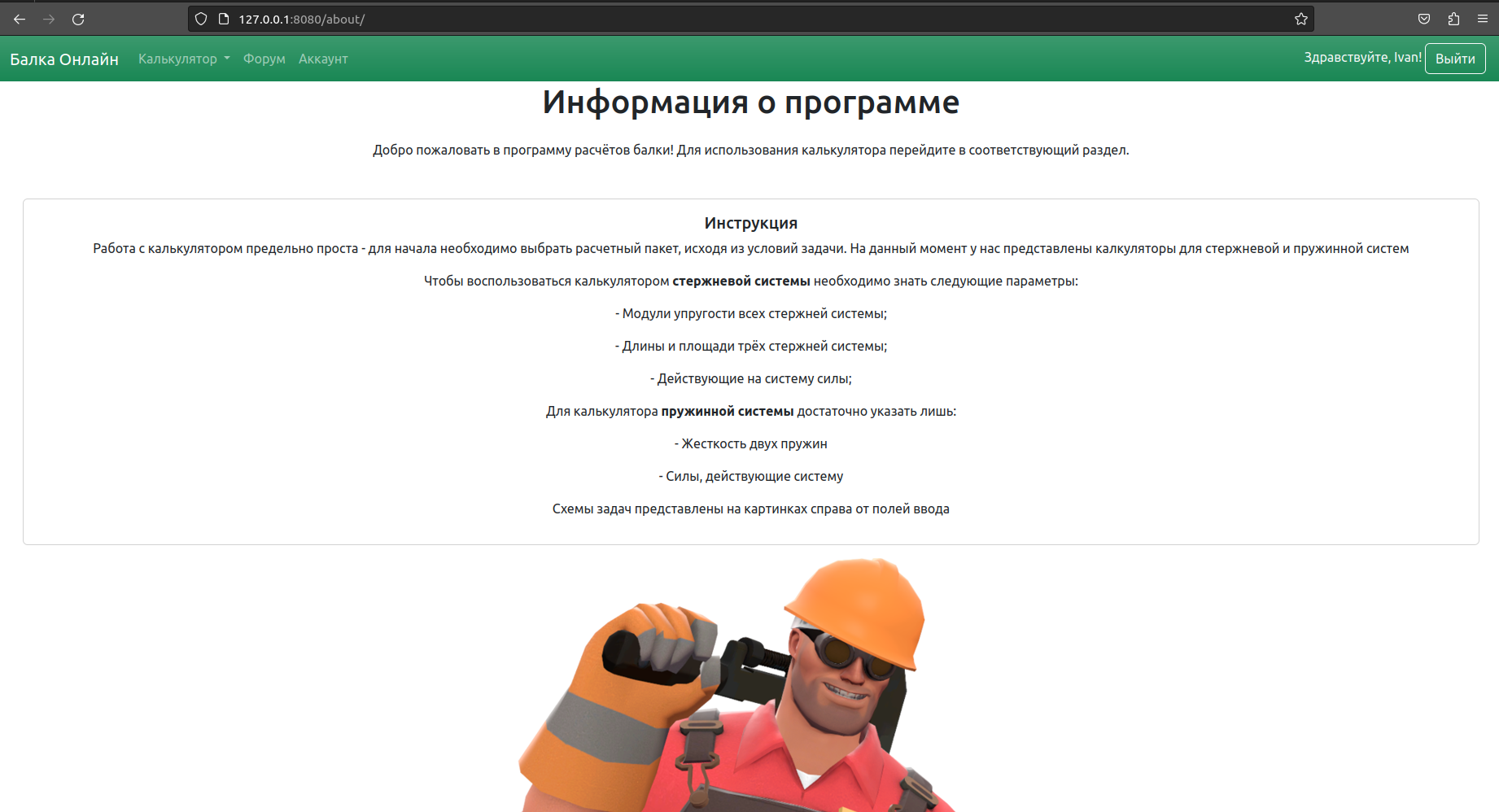


Рисунок 1 – Главная страница сервиса «Балка Онлайн»

# 3 Создание веб-приложения для проекта «Балка Онлайн»

## 3.1. Техническое задание

Требуется разработать многопользовательский веб-интерфейс, содержащий расчётные модули для задач сопротивления материалов. Для начала реализовать калькуляторы для стержневой и пружинной систем. Реализовать регистрацию пользователя, форум единомышленников для обсуждения вопросов и пожеланий.

## 3.2. Общие требования

* Использовать современные технологии и фреймворки;
* Бэкенд – Django, Python;
* Адаптивность под разные устройства (компьютер, смартфон);
* Страницы:
  + Главная страница;
  + Калькулятор:
    - Доступ имеют только зарегистрированные пользователи;
    - Выбор системы расчёта;
      * + Форум;
        + Аккаунт.

## 3.3. Главная страница веб-приложения

Хедер:

1. Логотип;
2. Навигация по главной странице;
3. Кнопка входа в аккаунт.

Контент:

1. Общая информация о приложении;
2. Инструкция по работе с калькуляторами.

Футер: стандартный футер с информацией.

## Калькулятор

На каждой странице будут представлены поля для ввода значений переменных, изображение механической системы с пояснениями, контейнер с выбором метода оптимизации и кнопка запуска.

## Административное управление через Django (со стороны администрации сайта)

На этой странице будет происходить управление учётными записями пользователей.

## База данных

База данных будет хранить информацию о пользователях: имя, фамилию, адрес электронной почты.

# Структура и управление веб-приложением

При обсуждении идеи веб-приложения была создана блок-схема его работы с помощью сервиса Miro (см. рис. 2).

Рисунок 2 – Блок схема работы веб-приложения

Внешний вид приложения менялся в процессе разработки. Например, рисунке 3 изображена первая версия отображения главной страницы. Был использован css-файл, созданный по туториалам из книги Антонио Меле «Django 4 в примерах». Далее мы отказались от пользовательских css-файлов и перешли к фреймворку Bootstrap 5.

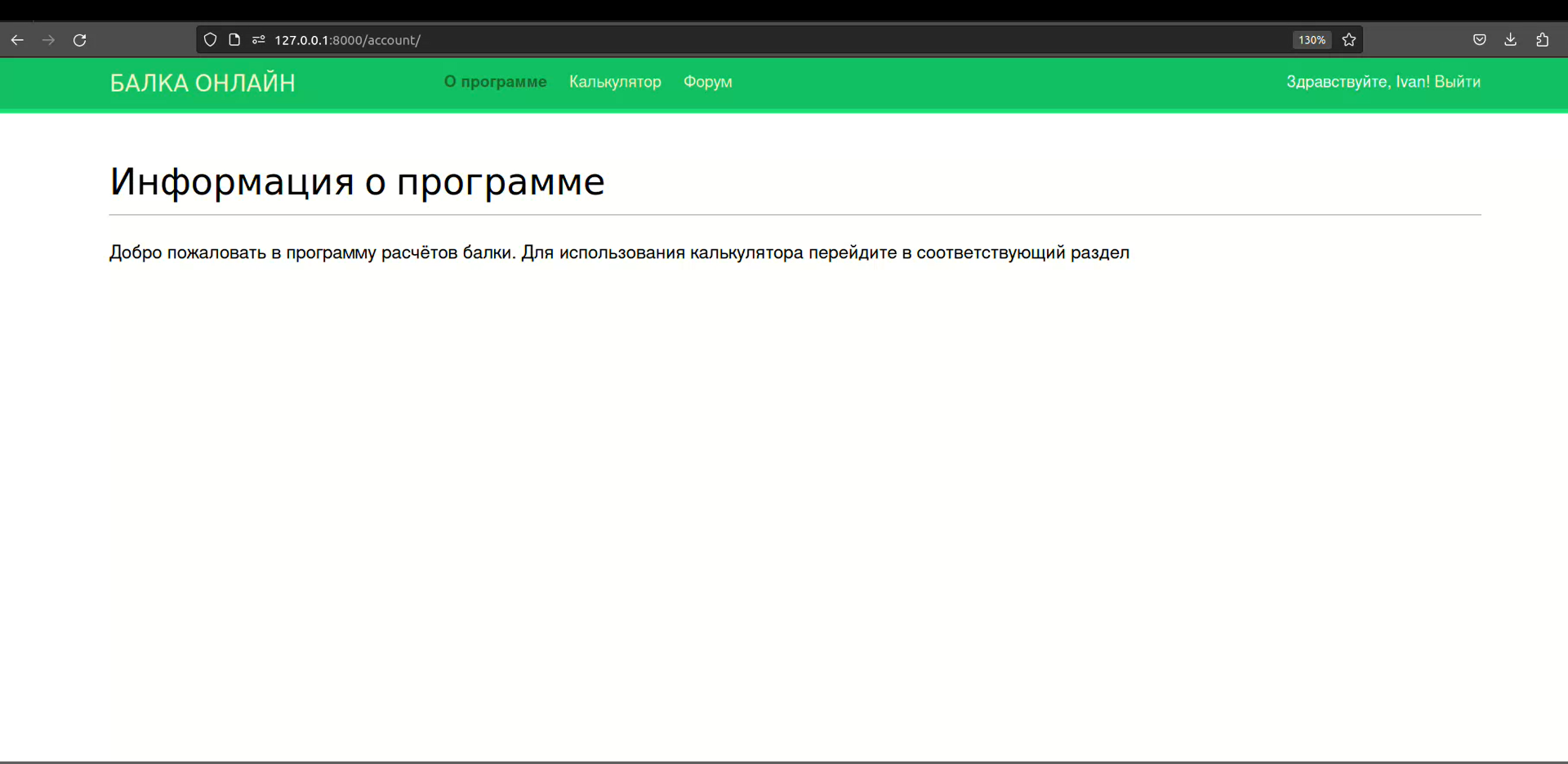


Рисунок 3 – Прототип главной страницы

На рисунке 4 изображен прототип отображения расчётного модуля.

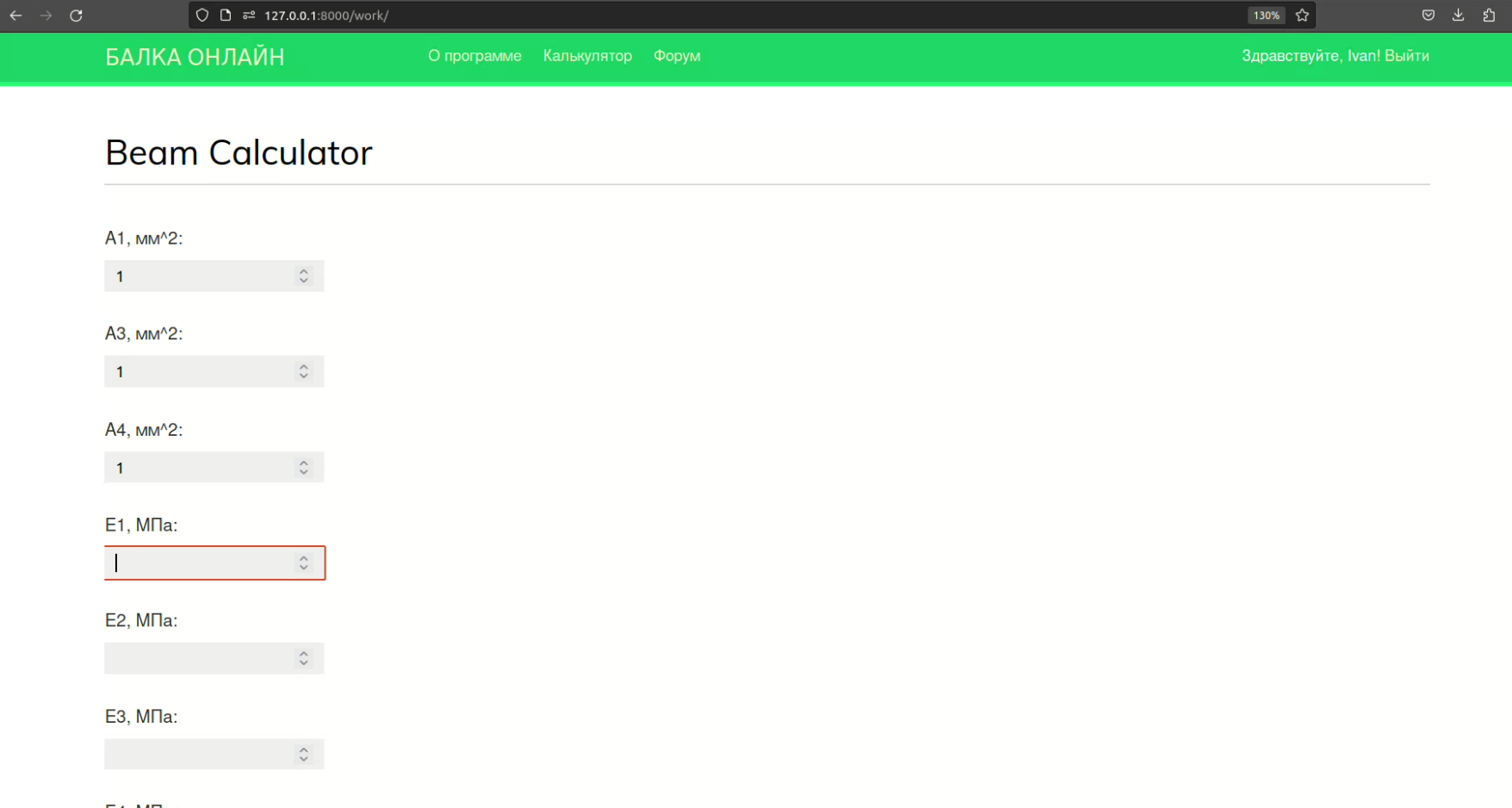


Рисунок 4 – Прототип расчётного модуля

На рисунке 5 изображен прототип отображения формы для входа в учётную запись.

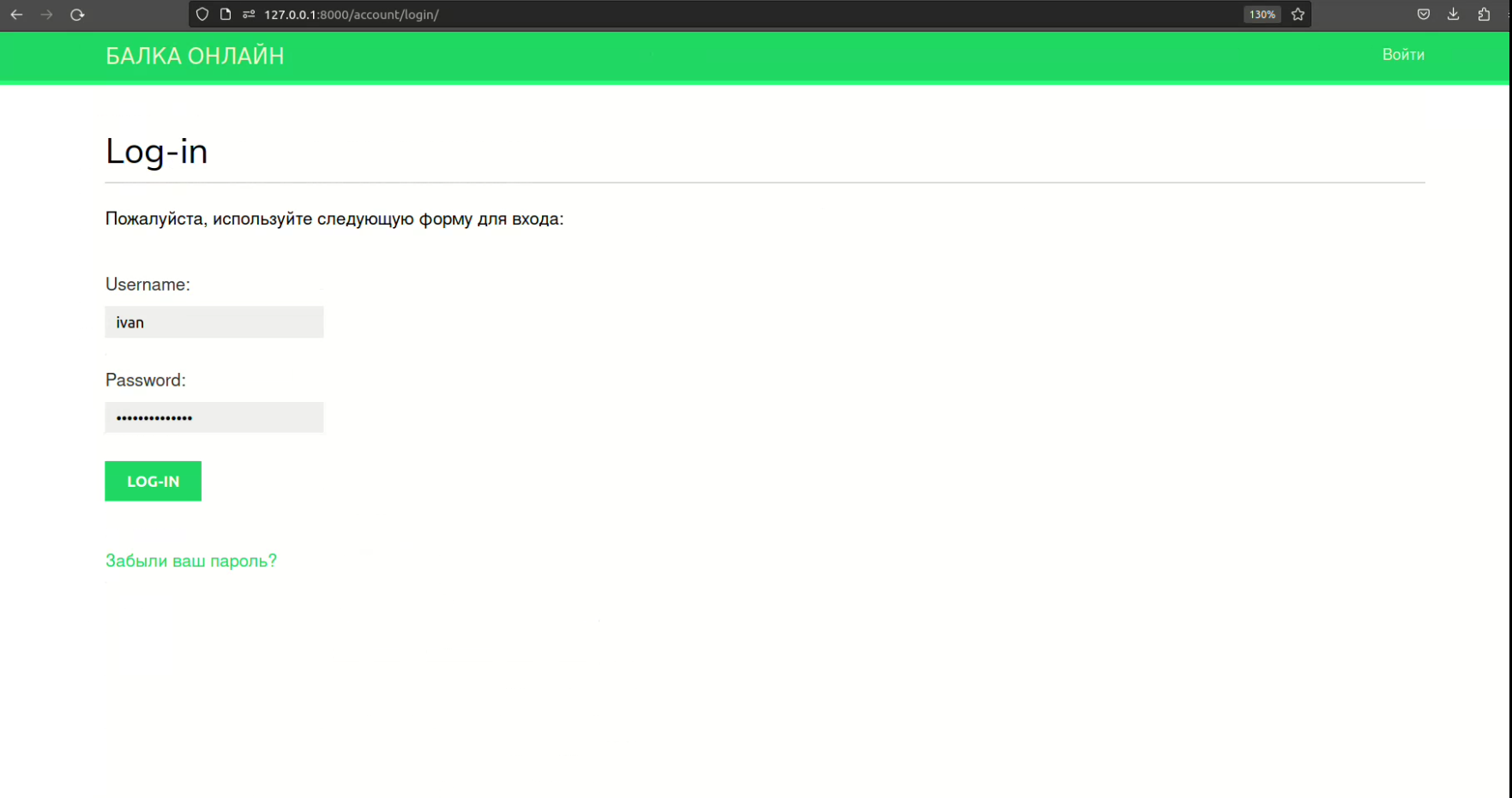


Рисунок 5 – Прототип формы для входа в аккаунт

# Работа веб-приложения

## Главная страница

Напомним, что сервис доступен только авторизированным пользователям, поэтому при переходе на главную страницу сайта, пользователь увидит информацию о программе и кнопку «Войти» в меню навигации, расположенном сверху.

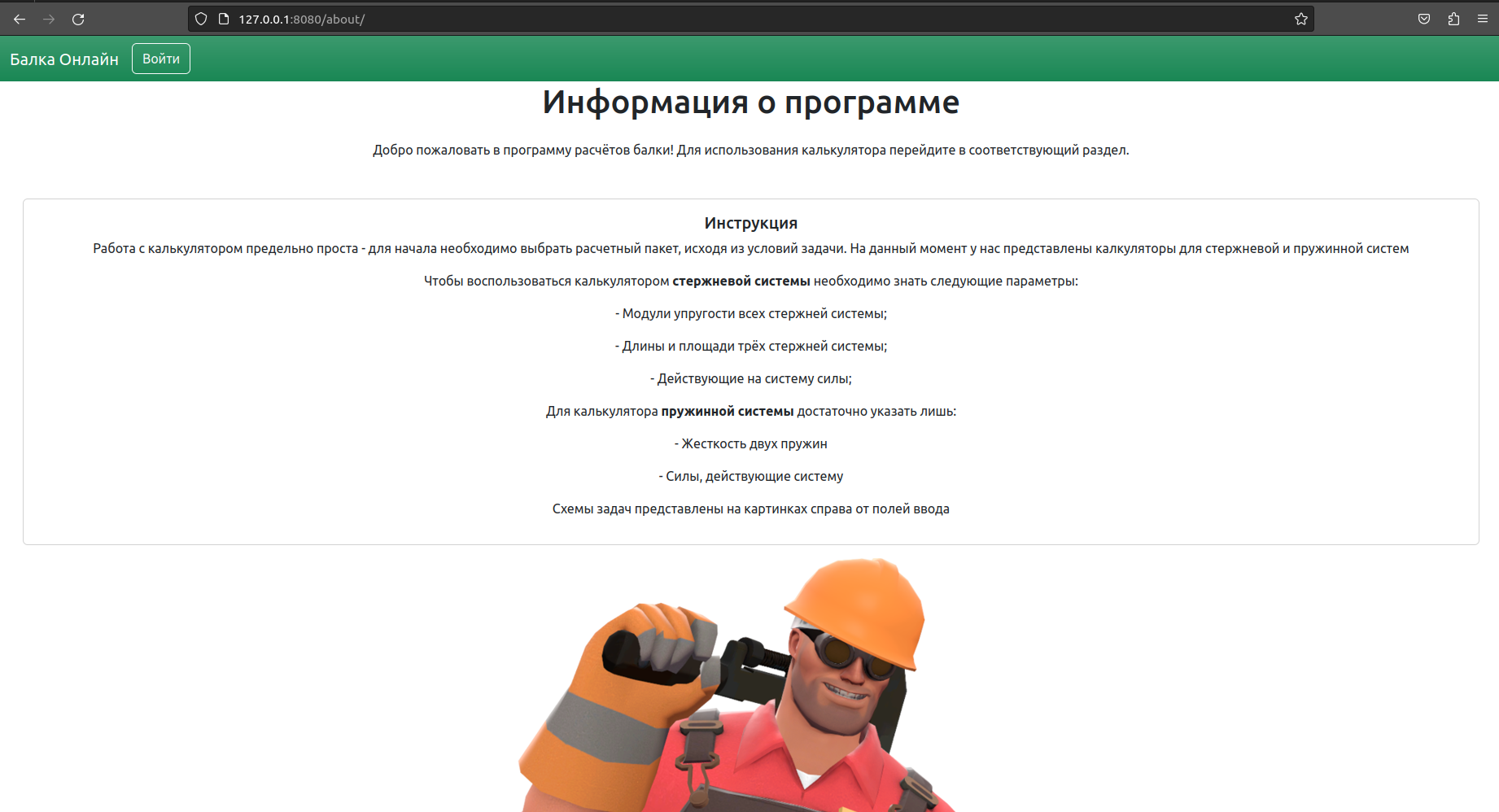


Рисунок 6 – Главная страница неавторизированного пользователя

Далее после успешной авторизации (и при необходимости регистрации) на главной странице сайта появится расширенное меню навигации, содержащее кнопки:

* Калькулятор;
* Форум;
* Аккаунт;
* Выйти.

Также в верхней шапке можно увидеть приветствие, содержащее имя пользователя.

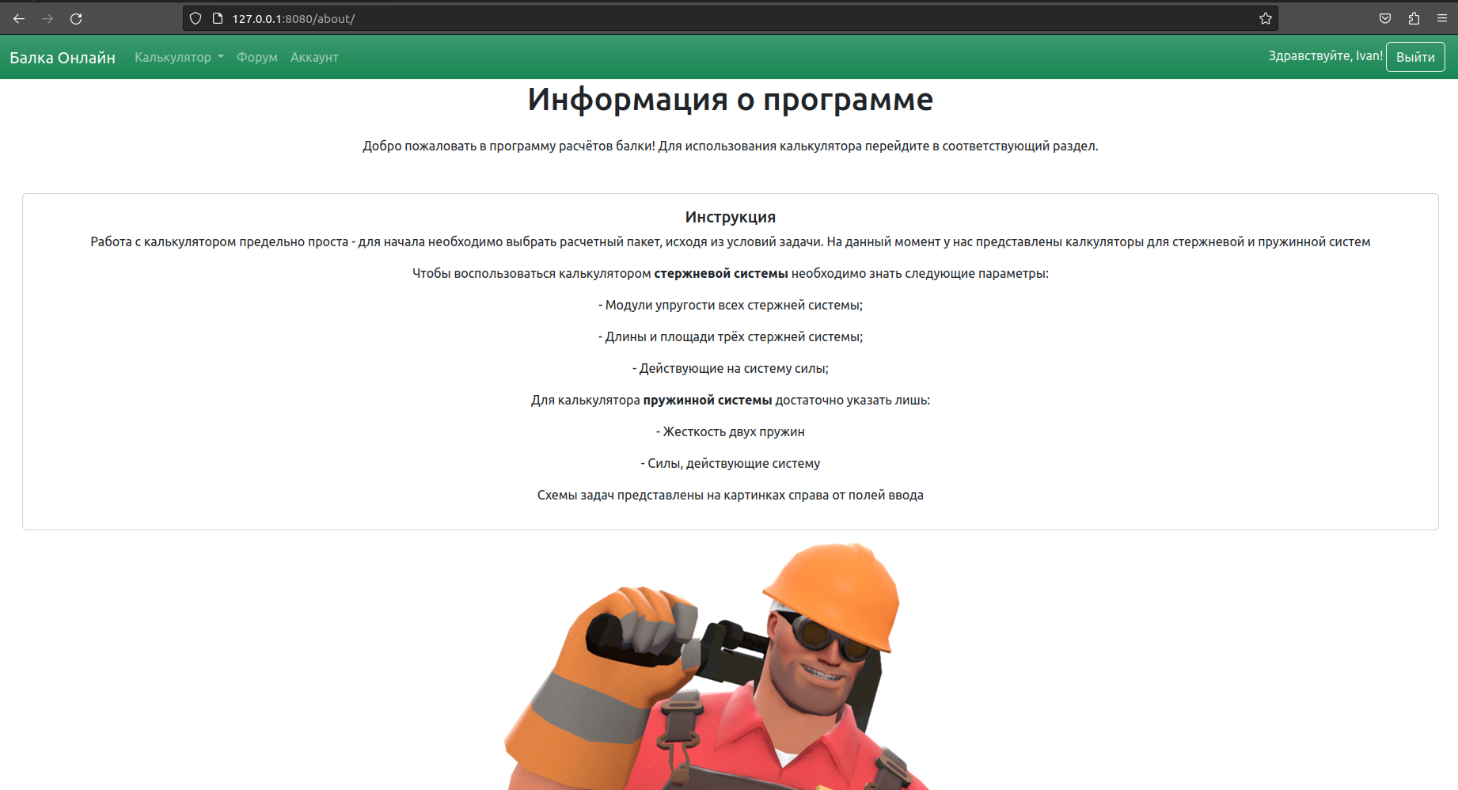


Рисунок 7 – Главная страница авторизированного пользователя

## Калькулятор

Чтобы войти на страницу калькулятора, изображенного на рисунке 8, нужно пройти регистрацию или ввести уже зарегистрированные email и пароль. На этой странице пользователь может ввести необходимые значения переменных в форму, а затем, нажав на кнопку «Вычислить», получить результаты.

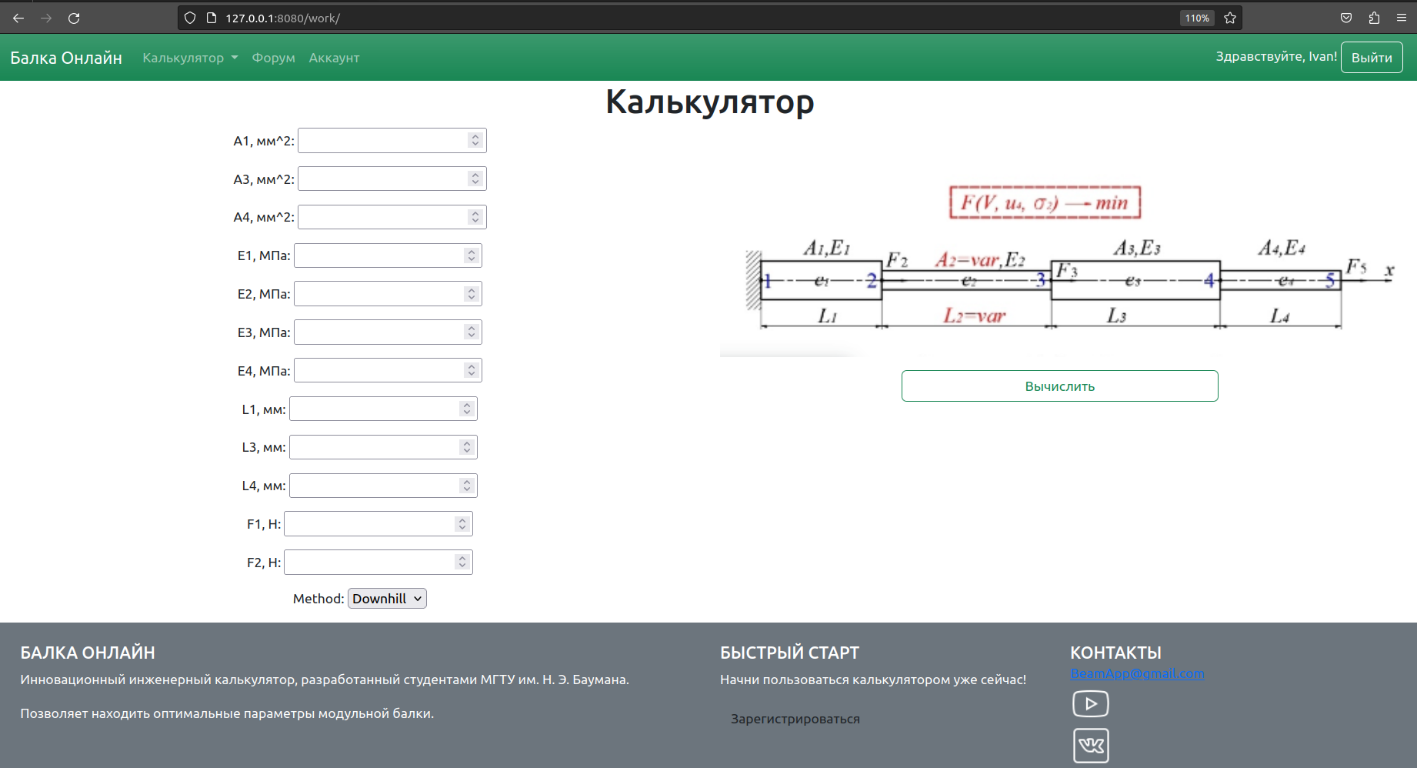


Рисунок 8 – Страница модуля расчётов стержневой системы

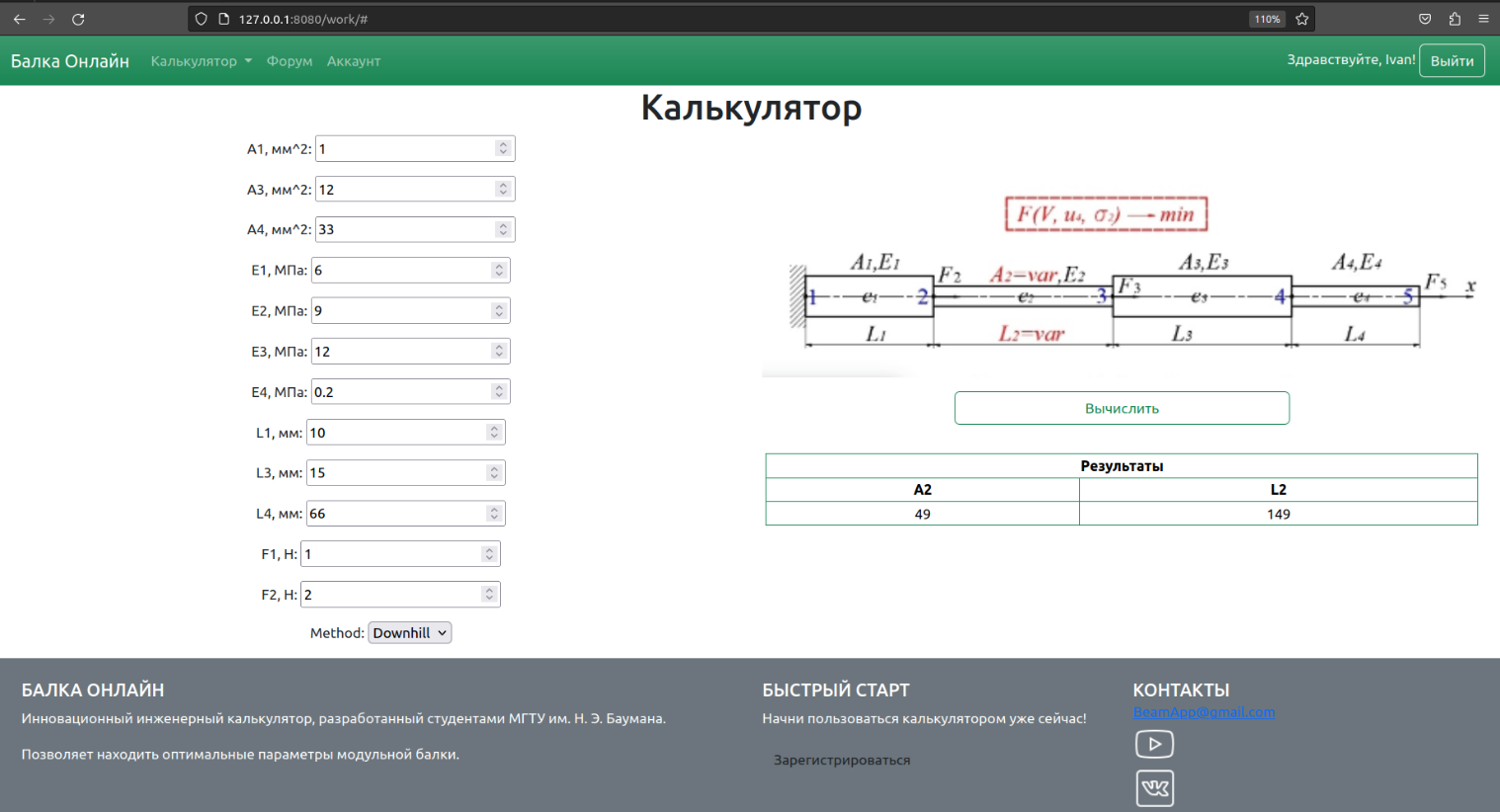


Рисунок 9 – Страница модуля расчётов стержневой системы с результатами

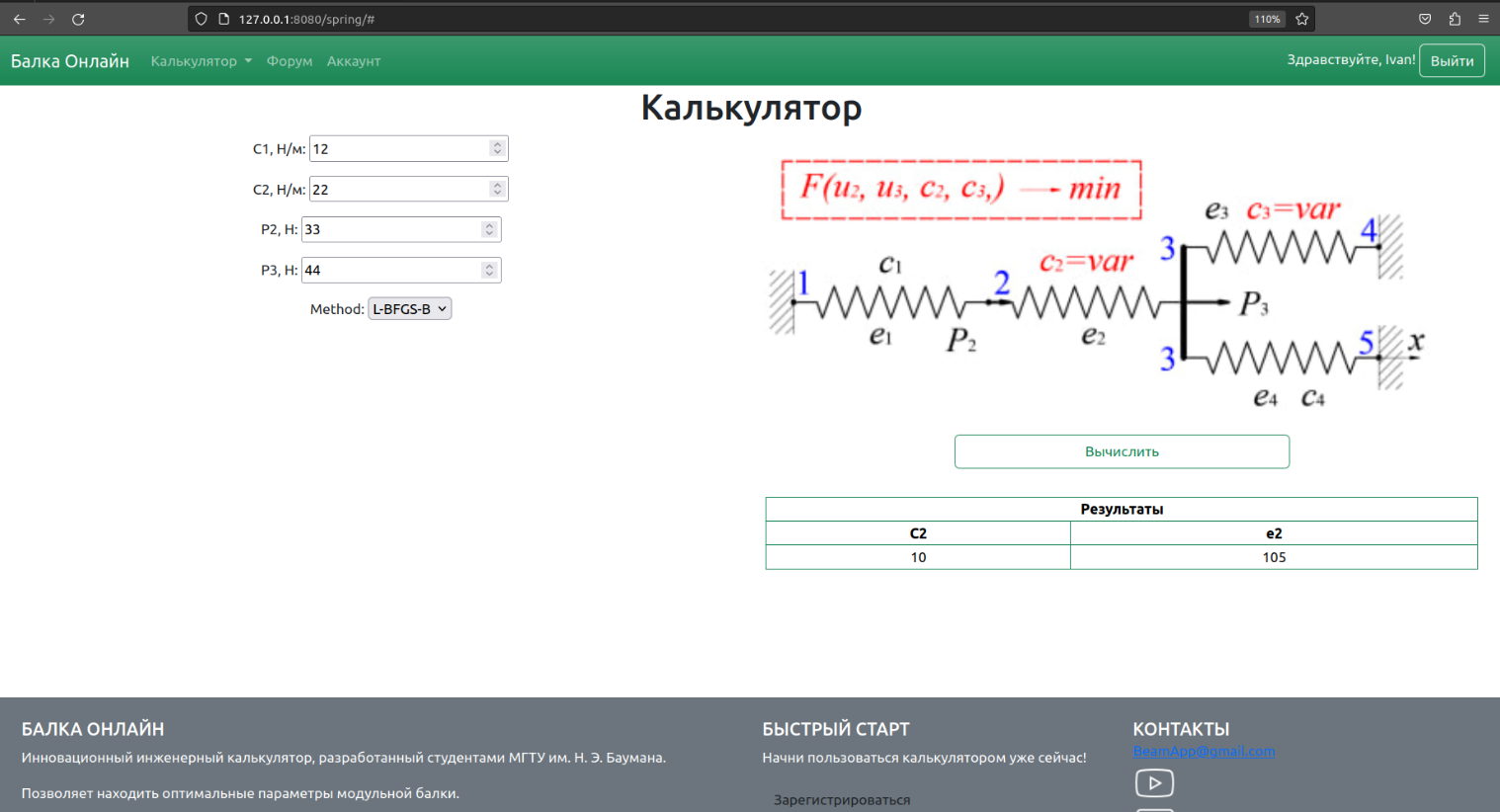


Рисунок 10 – Страница модуля расчётов пружинной системы с результатами

## Создание учётной записи

Для регистрации на сайте требуется ввести следующие данные пользователя:

* Имя пользователя (Username);
* Имя;
* Email;
* Пароль.

На рисунке 11 изображена страница регистрации. После регистрации пользователь получает доступ к функционалу сайта.

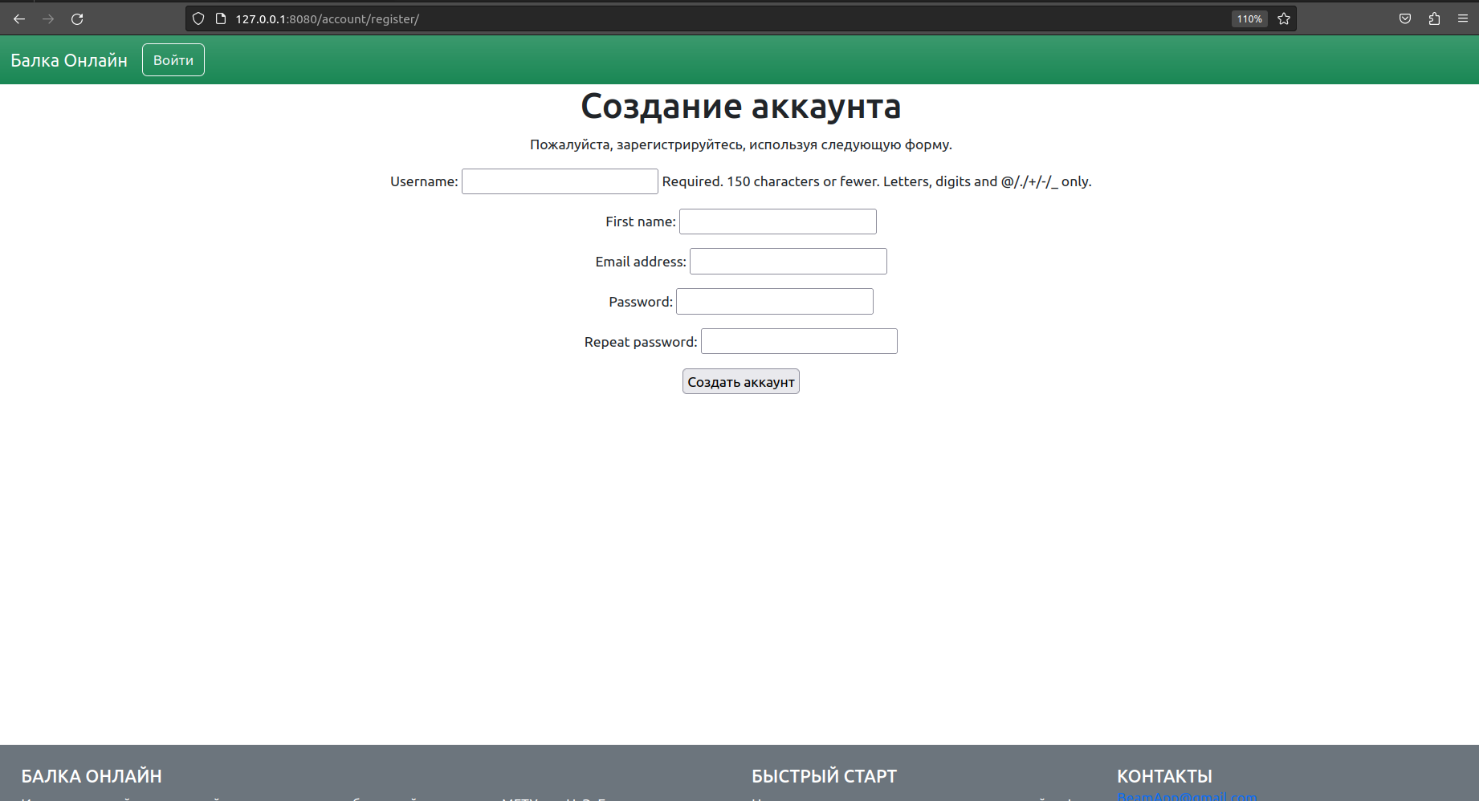


Рисунок 11 – Страница регистрации пользователя

## Форум

После входа на страницу «Форум», пользователь сможет создать новое обсуждение и поучаствовать в уже существующих.

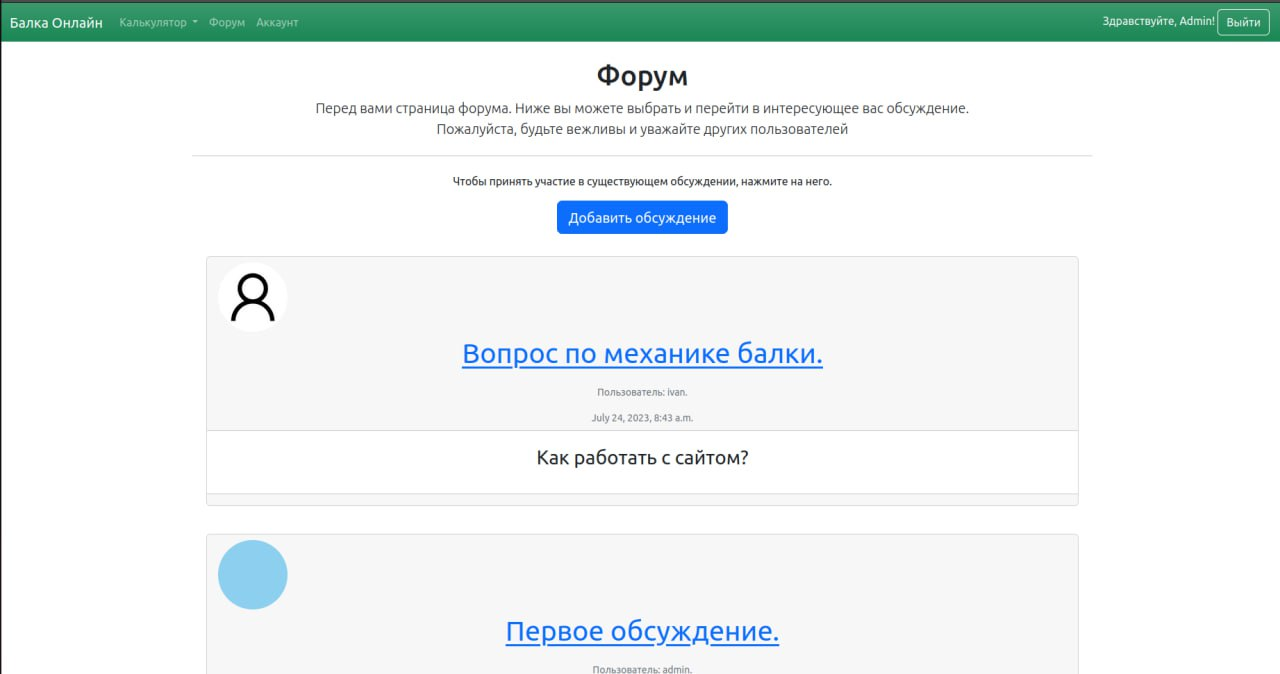


Рисунок 12.1 – Страница форума

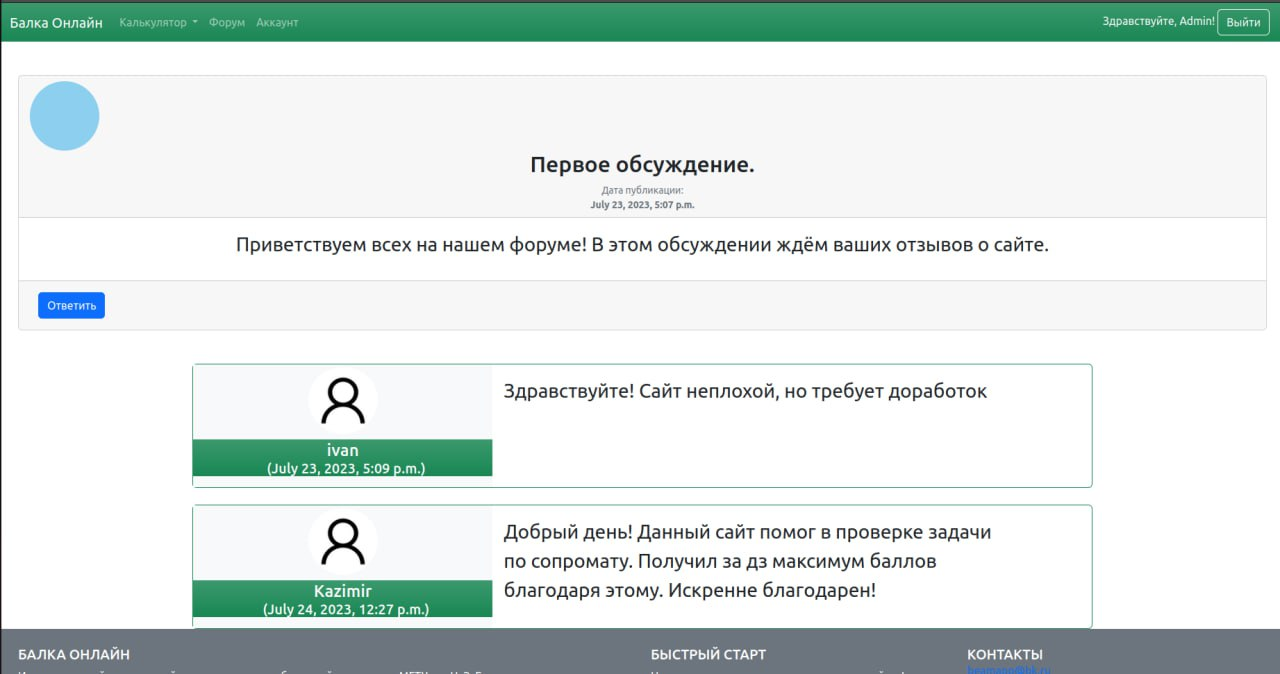


Рисунок 12.2 – Страница обсуждения

## Страница администратора сайта

Для управления данными сайта и учётными записями пользователей, существует предлагаемая Django страница администратора, изображенная на рисунке 13.

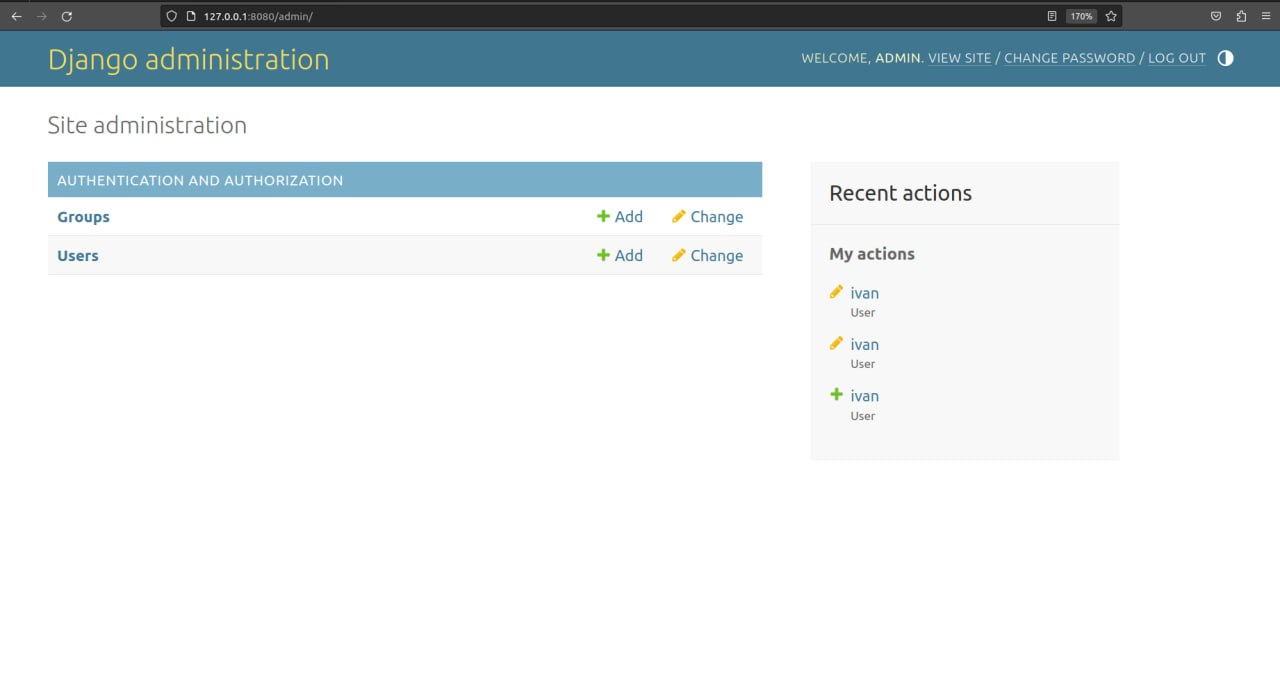


Рисунок 13 – Страница администратора

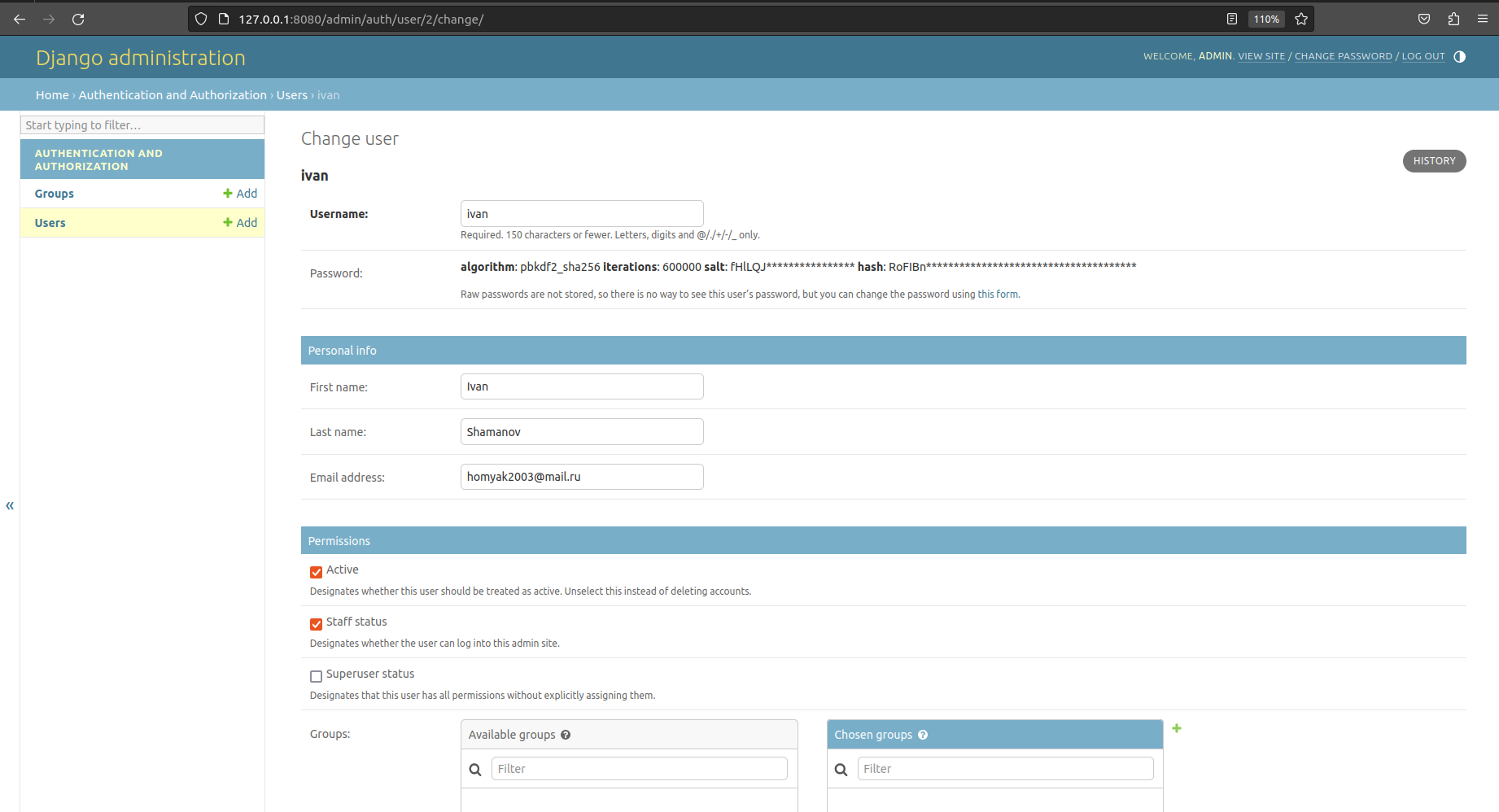


Рисунок 14 – Управление данными пользователя

# Программная реализация компонентов

## Модуль расчётов

Основа приложения расчетный модуль, он состоит из нескольких функций.  
1) Функция для решения СЛАУ: Метод Гаусса

При решении СЛАУ используются два класса методов: а) прямые – позволяющие найти решение за определённое число операций; б) итерационные – позволяющие получить решение в результате последовательных приближений.

**def** gaussElimin(a, b):

# Implementation of gaussElimin function

n=len(b)

**for** k **in** range(0, n-1):

**for** i **in** range(k+1,n):

**if** a[k,k] != 0.0:

lam=a[i,k]/a[k,k]

a[i,k+1:n]=a[i,k+1:n]-lam\*a[k,k+1:n]

b[i]=b[i]-lam\*b[k]

**for** k **in** range(n-1,-1,-1):

b[k]=(b[k]-np.dot(a[k,k+1:n],b[k+1:n]))/a[k,k]

**return** b

Листинг 1 - Функция для решения СЛАУ

2) Функция для поиска экстремумов. Симплекс метод (метод Нелдера-Мида)

Базовая задача оптимизации ставится как задача математического программирования:

|  |  |
| --- | --- |
|  |  |

где *F*(*X*) – целевая функция; *X* – вектор управляемых (проектных) параметров; *f*(*X*) и *y*(*X*) – функции-ограничения; *D*x – допустимая область в пространстве управляемых параметров. Запись интерпретируется как задача поиска экстремума целевой функции путем варьирования управляемых параметров в пределах допустимой области.

Методы оптимизации классифицируют по ряду признаков.

1) Число управляемых параметров: одномерные и многомерные.

2) Наличие/отсутствие ограничений: условной и безусловной.

3) Число экстремумов: одно- и многоэкстремальные. Если метод ориентирован на определение какого-либо локального экстремума, то такой метод относится к локальным методам. Если же результатом является глобальный экстремум, то метод называют методом глобального поиска.

4) Порядок производной целевой функции по управляемым параметрам. Если производные не используются, то метод нулевого порядка. Могут быть методы первого или второго порядков. Методы первого порядка – градиентные.

Наиболее простой метод – перебора (требуется большое число испытаний модели). Если задача имеет небольшую размерность, то можно использовать модификации метода сопряженных градиентов. Также распространено использование методов нулевого порядка и их модификаций: симплекс метод, метод Хука-Дживса, сопряженных направлений Пауэлла. При большом количестве варьируемых параметров высокую эффективность дают различные варианты «генетических алгоритмов».

Симплекс – это *n*-мерный тетраэдр. Если *n*=2, то симплекс представляет собой треугольник, если *n*=3, то тетраэдр. Данный метод заключается в перемещении симплекса в области проектирования, пока оптимальная точка не окажется внутри. Направление движения определяется значениями минимизируемой функции *F*(*x*) в вершинах. Вершину с наибольшим значением *F*(*x*) будем называть *H*i, а вершину с наименьшим значением – *L*o. Величина перемещения определяется расстоянием *d*, которое измеряется от вершины *H*i до центра противоположной грани (для треугольника – это середина противоположной стороны). Рассмотрим *n*=2.

Алгоритм будет выполняться до тех пор, пока характерный размер симплекса *d* не станет меньше или равен заданной погрешности ε. Первый шаг – перемещение вершины *H*i на расстояние 2*d* (рисунок 4.9, позиция 2). Если в новой вершине значение функции будет меньшим или равным, чем в *L*o, то применяем данную операцию. После чего пробуем сместить исходную вершину *L*o на расстояние 3*d* (позиция 3). Если в новой вершине значение функции будет большим, чем в *L*o, то сравниваем это значение с *H*i. Если оно больше *H*i, то исходную вершину *H*i смещаем на расстояние 0,5*d* (позиция 4). Если оно меньше *H*i, то уменьшаем площадь треугольника (позиция 5).

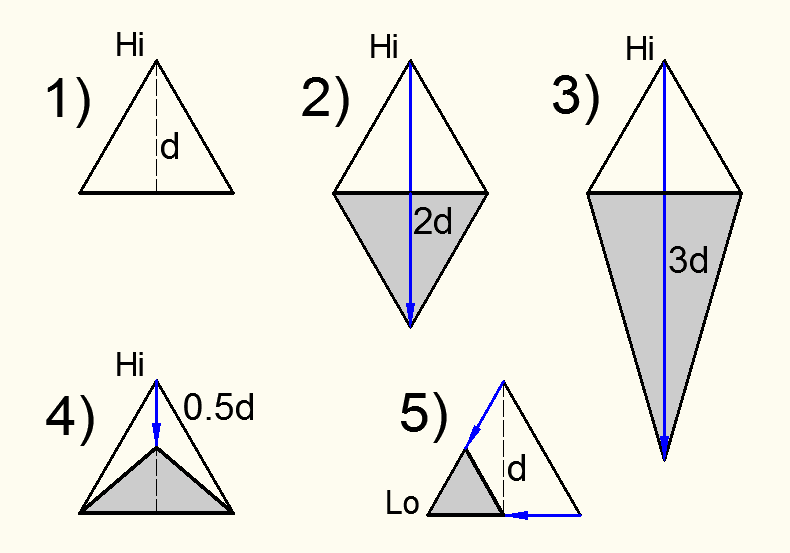


Рисунок 15 – Графическая иллюстрация симплекс метода

def downhill(F, xStart, side=1, tol=0.1):

# xStart - начальный вектор данных

# side - длина стороны симплекса (по умолчанию 0.1)

n=len(xStart) # Число варьируемых параметров

x=np.zeros((n+1,n))

f=np.zeros(n+1)

# создаем начальный симплекс:

x[0]=xStart

for i in range(1,n+1):

x[i]=xStart

x[i,i-1]=xStart[i-1]+side

# определение значений функции F в вершинах симплекса:

for i in range(n+1): f[i]=F(x[i])

#Главный цикл

for k in range(4000):

# Находим вершины с наибольшим и наименьшим значениями:

iLo=np.argmin(f)

iHi=np.argmax(f)

# Вычисляем вектор смещения:

d=(-(n+1)\*x[iHi]+np.sum(x,axis=0))/n

# Проверка условия достижения минимума:

if math.sqrt(np.dot(d,d)/n)<tol: return x[iLo]

# Попытка отражения

xNew=x[iHi]+2.0\*d

fNew=F(xNew)

if fNew<=f[iLo]: #то принимаем отражение

x[iHi]=xNew

f[iHi]=fNew

# Пытаемся еще продлить отражение

xNew=x[iHi]+d

fNew=F(xNew)

if fNew<=f[iLo]: # то принимиаем продление

x[iHi]=xNew

f[iHi]=fNew

else:

# Снова пробуем продление:

if fNew<=f[iHi]: # то принимиаем продление

x[iHi]=xNew

f[iHi]=fNew

else:

#Применяем сжатие

xNew=x[iHi]+0.5\*d

fNew=F(xNew)

if fNew<=f[iHi]: # то принимиаем продление

x[iHi]=xNew

f[iHi]=fNew

else:

# Используем уменьшение площади

for i in range(len(x)):

if i!=iLo:

x[i]=(x[i]-x[iLo])\*0.5

f[i]=F(x[i])

print("Too many iterations in downhill")

return x[iLo]

Листинг 2 - Функция для поиска экстремумов

3) Программная реализация метода конечных элементов

В МКЭ исходная область определения функции разбивается с помощью сетки, в общем случае неравномерной, на отдельные подобласти – конечные элементы. Искомая непрерывная функция аппроксимируется кусочно-непрерывной, определенной на множестве конечных элементов. Аппроксимация может задаваться произвольным образом, но чаще всего для этих целей используются полиномы, которые подбираются так, чтобы обеспечить непрерывность искомой функции в узлах на границах элементов.

Для расчета простейших конструктивных элементов используются пружины и линейные стержневые элементы. Плоские двухмерные элементы применяются для моделирования мембран, тонких пластин, тонкостенных оболочек и т.д. Объемные трехмерные элементы применяются при исследовании объемного НДС.

def F(x, form\_data):

# Implementation of F function

#A1\_var, A3\_var, A4\_var, E1\_var, E2\_var, E3\_var, E4\_var, L1\_var, L3\_var, L4\_var, F1\_var, F2\_var = create()

A1\_var = float(form\_data['var1'])

A3\_var = float(form\_data['var2'])

A4\_var = float(form\_data['var3'])

E1\_var = float(form\_data['var4'])

E2\_var = float(form\_data['var5'])

E3\_var = float(form\_data['var6'])

E4\_var = float(form\_data['var7'])

L1\_var = float(form\_data['var8'])

L3\_var = float(form\_data['var9'])

L4\_var = float(form\_data['var10'])

F1\_var = float(form\_data['var11'])

F2\_var = float(form\_data['var12'])

elementNodes=np.array([[1,2],[2,3],[3,4]])

numberNodes=elementNodes.max()

numberElements=elementNodes.shape[0]

prescribedDof=np.array([1,4])

A=np.array([A1\_var,x[0],A3\_var, A4\_var])

E=np.array([E1\_var,E2\_var,E3\_var, E4\_var])

L=np.array([L1\_var,x[1],L3\_var, L4\_var])

L\_obr=1/L

k\_st=E\*A/L

V=A[0]\*L[0]+A[1]\*L[1]+A[2]\*L[2]+A[3]\*L[3]

displacements=np.zeros((numberNodes,1))

force=np.zeros((numberNodes,1))

stiffness=np.zeros((numberNodes,numberNodes))

Sigma=np.zeros((numberElements,1))

force[1]=F1\_var

force[2]=F2\_var

for e in range (0,numberElements):

elementDof=elementNodes[e]

stiffness[elementDof[0]-1,elementDof[0]-1]=stiffness[elementDof[0]-1,elementDof[0]-1]+k\_st[e]

stiffness[elementDof[0]-1,elementDof[1]-1]=stiffness[elementDof[0]-1,elementDof[1]-1]-k\_st[e]

stiffness[elementDof[1]-1,elementDof[0]-1]=stiffness[elementDof[1]-1,elementDof[0]-1]-k\_st[e]

stiffness[elementDof[1]-1,elementDof[1]-1]=stiffness[elementDof[1]-1,elementDof[1]-1]+k\_st[e]

stiffness2=stiffness

Nodes\_num=list(range(1,numberNodes+1))

activeDof=[]

for i in Nodes\_num:

if i not in prescribedDof:

activeDof.append(i)

stiffness2=np.delete(stiffness2,(prescribedDof-1),axis=1)

stiffness2=np.delete(stiffness2,(prescribedDof-1),axis=0)

force2=force

force2=np.delete(force2,(prescribedDof-1),axis=0)

displacements2=gaussElimin(stiffness2,force2)

for i in range (0, len(activeDof)):

displacements[activeDof[i]-1]=displacements2[i]

Reactions=np.dot(stiffness,displacements)

for i in range (0,numberElements):

Sigma[i]=E[i]/L[i]\*(displacements[i+1]-displacements[i])

dd=(displacements[2]).tolist()

dd1=dd[0]

return (V+(displacements[3]\*10000)+(x[0]-50)\*\*4+(x[1]-150)\*\*2+(Sigma[1]-500)\*2)

Листинг 3 – реализация метода конечных элементов



## Первоначальная реализация веб-интерфейса раздела «Калькулятор»

На странице есть 12 полей ввода, куда пользователь вписывает значения параметров системы, картинка самой системы с пояснениями, контейнер с выбором метода решения и кнопка запуска. При нажатии на неё сайт обновляется и снизу появляется таблица с выводом решения.

Данные, введённые пользователем сохраняются в форме, затем в views.py через функцию create данные обрабатываются и выводятся на странце.

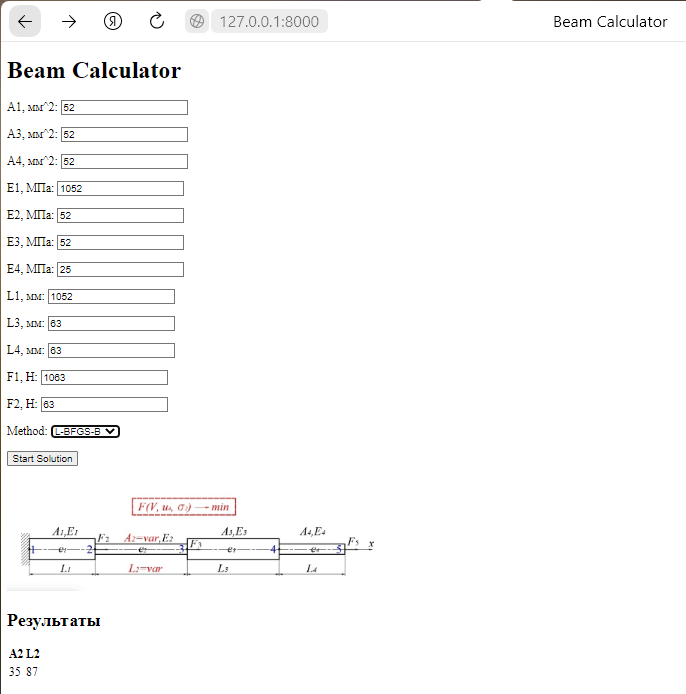


Рисунок 16 – Отображение приложения Calculator в браузере

**F**rom Django import forms

class InputForm(forms.Form):

var1 = forms.FloatField(label=’A1, мм^2’)

var2 = forms.FloatField(label=’A3, мм^2’)

var3 = forms.FloatField(label=’A4, мм^2’)

var4 = forms.FloatField(label=’E1, Мпа’)

var5 = forms.FloatField(label=’E2, Мпа’)

var6 = forms.FloatField(label=’E3, Мпа’)

var7 = forms.FloatField(label=’E4, Мпа’)

var8 = forms.FloatField(label=’L1, мм’)

var9 = forms.FloatField(label=’L3, мм’)

var10 = forms.FloatField(label=’L4, мм’)

var11 = forms.FloatField(label=’F1, H’)

var12 = forms.FloatField(label=’F2, H’)

var\_choices = [

(0, ‘Downhill’),

(1, ‘L-BFGS-B’)

]

var = forms.ChoiceField(label=’Method’, choices=var\_choices)

Листинг 4 – forms.py

def create(request):

form = InputForm(request.POST)

rez1 = None # Initialize rez1 with a default value

rez2 = None # Initialize rez2 with a default value

if form.is\_valid():

var1 = form.cleaned\_data[‘var1’]

var2 = form.cleaned\_data[‘var2’]

var3 = form.cleaned\_data[‘var3’]

var4 = form.cleaned\_data[‘var4’]

var5 = form.cleaned\_data[‘var5’]

var6 = form.cleaned\_data[‘var6’]

var7 = form.cleaned\_data[‘var7’]

var8 = form.cleaned\_data[‘var8’]

var9 = form.cleaned\_data[‘var9’]

var10 = form.cleaned\_data[‘var10’]

var11 = form.cleaned\_data[‘var11’]

var12 = form.cleaned\_data[‘var12’]

var = form.cleaned\_data[‘var’]

…..

return render(request, ‘work/numbers.html’, {‘form’: form, ‘rez1’: rez1, ‘rez2’: rez2})

else:

form = InputForm()

return render(request, ‘work/numbers.html’, {‘form’: form})

Листинг 5 – основная часть функции Create views.py

{% load static %}

<!DOCTYPE html>

<html>

<head>

<title>Beam Calculator</title>

<link rel=”stylesheet” type=”text/css” href=”{% static ‘numbers.css’ %}”>

</head>

<body>

<h1>Beam Calculator</h1>

<form method=”POST” action=”#”>

{% csrf\_token %}

{{ form.as\_p }}

<button type=”submit”>Start Solution</button><br>

<img src=”{% static ‘work/images/1.png’ %}” alt=”Scheme"><br>

{% if rez1 and rez2 %}

<h2>Результаты</h2>

<table>

<thead>

<th>A2</th>

<th>L2</th>

</thead>

<tr>

<td>{{ rez1 }}</td>

<td>{{ rez2 }}</td>

</tr>

</table>

{% endif %}

</body>

</html>

Листинг 6 – numbers.html, отвечающий за отображение страницы