

ИНДИВИДУАЛЬНЫЙ ПРОЕКТ 6 ЭТАП

Коняева Марина НФИбд-01-21

03.06.2022

Размещение двуязычного сайта на Github.

1. Сделать поддержку английского и русского языков.
2. Разместить элементы сайта на обоих языках.
3. Разместить контент на обоих языках.
4. Сделать пост по прошедшей неделе.
5. Добавить пост на тему по выбору (на двух языках).

Выполнение лабораторной работы

1. Сделать поддержку английского и русского языков.

Разместить элементы сайта на обоих языках. Разместить контент на обоих языках. (Изображение 1.1-3)


```
# Default language
en:
  languageCode: en-us
  # Uncomment for multi-lingual sites, and move English content into `en` sub-folder.
  contentDir: content/en

# Uncomment the lines below to configure your website in a second language.
ru:
  languageCode: ru-ru
  contentDir: content/ru
  title: Chinese website title...
  params:
    description: Site description in Chinese...
  menu:
    main:
      - name: Главная
        url: '#about'
        weight: 10
      - name: Псоты
        url: '#about'
        weight: 20
      - name: Проекты
        url: '#about'
        weight: 30
      - name: Мероприятия
```

Академический сайт научного работника

Главная Посты Проекты Мероприятия Публикации Связь

🔍 🌙 🌐



Биография

Марина Коняева является студенткой бакалавриата Российского университета дружбы народов на факультете физико-математических и естественных наук по направлению: Основы информатики и информационных технологий. Ее научные интересы включают моделирование и анализ производительности телекоммуникационных систем, методы анализа производительности телекоммуникаций и системы потерь со случайными требованиями.

Скачать мое resumé.

Коняева Марина Александровна

студентка


(6) Российский университет дружбы народов (РУДН)

Интересы

- Искусственный интеллект
- Анализ данных
- Информационный поиск

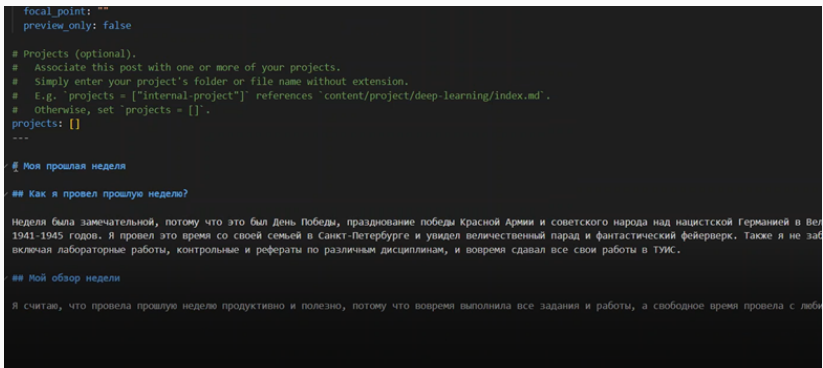
Образование

🎓 Основы информатики и информационных технологий, 2025
Университет РУДН



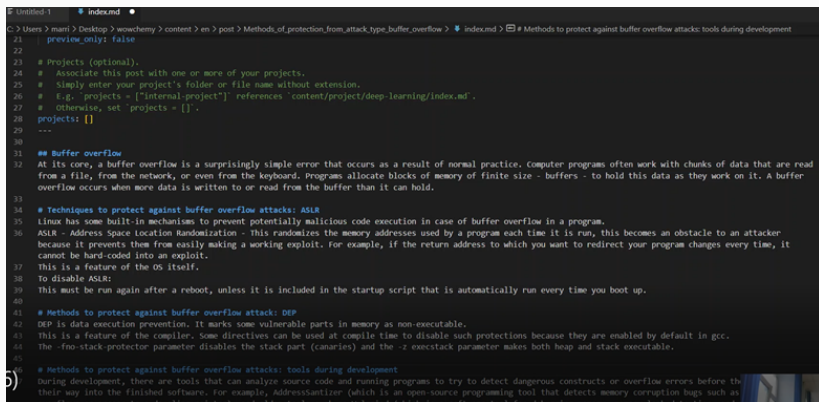
Изображение 1.2 Сделать записи для персональных проектов

2. Добавим к сайту пост по прошедшей неделе (Изображение 2.1)



Изображение 2.1 пост по прошедшей неделе

3. Добавить пост на тему по выбору (Изображение 3.1)



```

C:\Users\mari> Desktop > wowchemy > content > en > post > Methods_of_protection_from_attack_type_buffer_overflow > index.md > # Methods to protect against buffer overflow attacks: tools during development
21 | preview_only: false
22
23 # Projects (optional).
24 # Associate this post with one or more of your projects.
25 # Simply enter your project's folder or file name without extension.
26 # E.g. 'projects = ["internal-project"]' references 'content/project/deep-learning/index.md'.
27 # Otherwise, set 'projects = []'.
28 projects: []
29 ---
30
31 ## Buffer overflow
32 At its core, a buffer overflow is a surprisingly simple error that occurs as a result of normal practice. Computer programs often work with chunks of data that are read from a file, from the network, or even from the keyboard. Programs allocate blocks of memory of finite size - buffers - to hold this data as they work on it. A buffer overflow occurs when more data is written to or read from the buffer than it can hold.
33
34 ## Techniques to protect against buffer overflow attacks: ASLR
35 Linux has some built-in mechanisms to prevent potentially malicious code execution in case of buffer overflow in a program.
36 ASLR - Address Space location Randomization - This randomizes the memory addresses used by a program each time it is run, this becomes an obstacle to an attacker because it prevents them from easily making a working exploit. For example, if the return address to which you want to redirect your program changes every time, it cannot be hard-coded into an exploit.
37 This is a feature of the OS itself.
38 To disable ASLR:
39 This must be run again after a reboot, unless it is included in the startup script that is automatically run every time you boot up.
40
41 ## Methods to protect against buffer overflow attack: DEP
42 DEP is data execution prevention. It marks some vulnerable parts in memory as non-executable.
43 This is a feature of the compiler. Some directives can be used at compile time to disable such protections because they are enabled by default in gcc.
44 The -fno-stack-protector parameter disables the stack part (canaries) and the -z execstack parameter makes both heap and stack executable.
45
46 ## Methods to protect against buffer overflow attacks: tools during development
47 During development, there are tools that can analyze source code and running programs to try to detect dangerous constructs or overflow errors before they their way into the finished software. For example, AddressSanitizer (which is an open-source programming tool that detects memory corruption bugs such as overflow or access to a disabled pointer), and older tools such as Valgrind (which is a software tool for debugging memory, memory leak detection, and
```

Изображение 3.1 языки научного программирования

В ходе выполнения данного этапа индивидуального проекта добавили двуязычный сайт на Github, а также создали посты.