WEEKLY REPORT #1

Project
WEB scraper for online stores
Team Members
Nikita Ruchkin (n.ruchkin@innopolis.university)
GitHUB
https://github.com/reterman/test_project.git

PART 1: BRIEF DESCRIPTION OF THE PROJECT IDEA

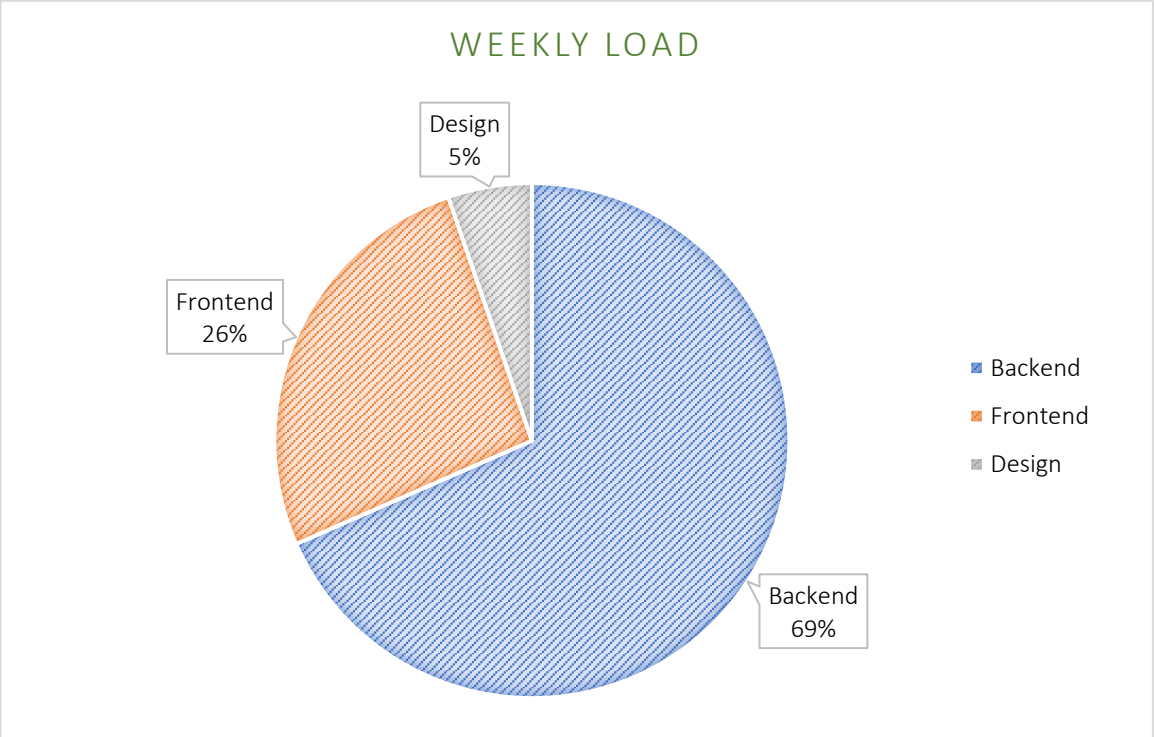
A price comparison tool that uses web scraping to gather pricing information from various online retailers for a specific product, and provides the user with a side-by-side comparison of the prices, allowing them to save money and make informed purchasing decisions. The tool can also provide notifications and alerts when the price of an item drops below a certain threshold.

PART 2: WEEKLY SPRINT

2.1. Workload distribution among team members

Since I am the only participant in my team, I performed every part of the project development on my own.

During the week, I spent a total of 12,5 hours on the first sprint.



2.2. Weekly tasks

#	Brief description	Hours spent
#1	Studying and connecting libraries and packages for scraping websites (request, selenium, bs4)	2

#2	Implementation of functions for scraping online shopping platform (Marketplace) (Wildberries.ru)	4,5
#3	Studying and installing libraries and packages to implement a web application (flask, django)	2
#4	Implementation of web pages with minimal functionality to test the functionality	1
#5	Study CSS framework for quick and quality method for implementing fronted part of the project (Bulma)	1
#6	Applying CSS framework in practice	1
#7	Cleaning the code and adding comments	0,5
#8	Compiling files and uploading to gitHub	0,5

2.3. Detailed description of the tasks

#1 Studying and connecting libraries and packages for scraping websites (request, selenium, bs4)

Requests:
Request is used for web scraping to download the HTML content of the web page.

Selenium:
Selenium is used for web scraping when the web page requires user interaction or has dynamic content that is loaded using JavaScript.

BeautifulSoup4 (bs4):
Once you have the HTML content of the web page, you can use BeautifulSoup to extract the relevant information.

#2 Implementation of functions for scraping online shopping platform (Marketplace) (Wildberries.ru)

To connect these libraries and packages, I would typically start by using Requests to download the HTML content of the web page. However, most modern web sites use dynamic content, I may need to use Selenium to automate the web browser. Once I have the HTML content, I can use BeautifulSoup to extract the relevant information.

#3 Studying and installing libraries and packages to implement a web application (flask, django)

Flask:
Flask is a lightweight web framework for Python that allows you to quickly build web applications.

Django:
Django is a more comprehensive web framework for Python that provides a powerful and scalable architecture for creating web applications.

After studying both technologies I settled on flask. However, in the future, if there is a need to use a database, I will have to consider a tool like Django.

#4 Implementation of web pages with minimal functionality to test the functionality

Creating HTML templates:
To create web pages using Flask, I need to create HTML templates that define the structure and content of the pages.

Defining routes:
To define routes in Flask, I can use the "@app.route" decorator and specify the URL path for the route. For example, "@app.route('/home')" defines a route for the home page of the web application.

Defining views:
To define views in Flask, I need to create Python functions that return the HTML content for each route. I can use the render_template function to render the HTML templates and pass any necessary data to the templates.

Running the Flask application:
To run the Flask application, I need to create an instance of the Flask class and run the app using the "app.run()" method.

Testing the application:
Once I have defined the routes and views for my web pages, I can test the application by running it using local port and accessing the pages in a web browser.

#5 Study CSS framework for quick and quality method for implementing fronted part of the project (Bulma)

Bulma:
Bulma is a modern CSS framework that is based on Flexbox and provides a range of pre-built UI components and styles for creating responsive web pages.

#6 Applying CSS framework in practice

Using ready-made templates and css-classes, quickly managed to implement a nice-looking initial interface, but in the future I will improve it, thinking of other design solutions.

#7 Cleaning the code and adding comments

The usual cleaning of the code and providing clear comments for further convenience orientation in the code

#8 Compiling files and uploading to gitHub

The layout of all files and sending via a request to the necessary repository, automatically adding the file .gitignore

VS Code has built-in support for Git, which makes it easy to manage and track changes to your code. You can use the Git pane to commit changes, create branches, and push changes to GitHub.

VS Code also has built-in support for GitHub, which allows you to interact with your GitHub repositories directly from the editor. You can clone repositories, create new repositories, and publish your code to GitHub without leaving VS Code.

PART 3: AN OBSTACLE THAT OCCURRED DURING THE SPRINT

#1 The lack of necessary skills for the implementation of the project, resulting in a waste of resources for their study.

#2 Vulnerability detection pulling information out of html code, as a result of dynamic changes in the naming of css-classes.

#3 Lack of compilation of the final stack of technologies used.

PART 4: THE MAIN TASKS FOR THE FOLLOWING SPRINTS

#1 To increase the number of used marketplaces.

#2 Improve the visual component of the site content.

#3 Optimize the work of selenium, to increase the speed of issuing product cards.

#4 The ability to search for products using sorting by price, rating, comments, etc.