**Documentation**

**File structure**

*avatars* – The folder where upload users' avatars

*docs* – The folder with documentation

*resume* – The folder where upload users' resumes

*account.php* –

Displaying the user profile and processing changes made to the profile

*apply.php*  - Transmits information about the job description

*createCategory.php* - Сategory creation handler

*createCategoryForm.php* - The form for creating a category

*createJob.php* - Сreating a job

*deleteCategory.php* - Removes a category from the site

*deleteJob.php* - Removes a job from the site

*footer.php* - Date bar at the bottom of the page

*getData.php* - Getting data about categories

*getJobDetails.php* - Getting data about job description

*getJobs.php* - Getting data about jobs in categories

*header.php* - Site header with home, registration and login buttons

*index.php* - Displaying the main page

*index.js* - Displaying a list of categories on the main page, click handler, pagination

*jobDescription.js* - Displaying job data obtained using apply.php

*jobs.js* - Displaying a list of jobs in the selected category, pagination

*jobs.php* - Getting a list of jobs in the selected category

*jobsDescription.php* - Displaying the job description

*loadResume.js* - Resume upload button handler

*loadResume.php* - Uploading a resume to the server

*log.php* - The handler of the login form

*login.php* - Displaying the login form

*logout.php* - Log out of account and go to the home page

*reg.php* - Registration form handler

*registration.php* - Registration form

*regValidate.js* - Checking the validity of the data on the client side

*style.css* - Styles for the entire site

*uploadJob.php* - Uploading a new job to the server

**Logic**

The project contains several **html** files used to define the structure of the page.

**СSS** (style.css) is used for styles.

**Php** is used for server-side validation as well as for working with the database (receiving and writing data).

The **javascript** is used to validate the entered data on the client side, display a list of categories, vacancies, and for pagination.

**Validation**

Client-side:

* the completeness of the required fields was checked using javascript
* checking the correctness of the entered data, image format and matching passwords using Javascript
* error output for the client using alert

Server-side:

* using htmlspecialchars to process client input
* checking completeness of the required fields
* checking the correctness of the entered data, image format and matching passwords
* checking for the existence of the entered email in the database
* image size reduction
* using prepared expressions for security against sql injections
* sending success or error messages in json format, which are then processed by the javascript

**Database structure**

The database contains four tables:

* **users**: *id, name, surname, email, password, role, avatar*

Сontains user data. The password is set using a hash, the avatar contains the path to the image.

* **categories**: *id, name*

Сontains information about categories.

* **jobs**: *id, name, description, salary, category\_id, company, email*

Сontains information about vacancies.

* **resumes**: *resume\_path*

Contains the path to the resume uploaded by the user.

**Ajax**

**Form validation (regValidate.js):**

An asynchronous AJAX request is used to validate the form data. The *submitAjax* function sends with POST data to the server (reg.php ), waits for a response in JSON format and returns a message with the validation result. If successful, redirects to the account.php.

**Uploading a list of categories/vacancies with pagination (index.js/jobs.js):**

An asynchronous AJAX request is also used to load data with pagination. The loadCategoriesWithPagination/loadJobsWithPagination function sends a request to the server (getData.php/getJobs.php ), updates the interface according to the received data and saves the current page number in the session storage.

Ajax works in a similar way to get a job description (jobDescription.js) and upload a resume to the server (loadResume.js).