



МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ
УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И
ТЕХНОЛОГИЧЕСКОГО ОБРАЗОВАНИЯ**
Кафедра информационных технологий и электронного обучения

Основная профессиональная образовательная программа
Направление подготовки 09.03.01 Информатика и вычислительная техника
Направленность (профиль) «Технологии разработки программного обеспечения»
форма обучения – очная

Курсовая работа

Управление проектом по разработке серверной составляющей
информационной системы для студенческой биржи труда

Обучающегося 3 курса
Мельникова Федора Владиславовича

Научный руководитель:
Кандидат физико-математических наук,
доцент кафедры ИТиЭО
Жуков Николай Николаевич

Санкт-Петербург
2021

Оглавление

Введение.....	3
Глава 1. Описание проекта и выбор методологии управления проектом	4
1.1. Предметная область	4
1.2. Методология управления разработкой	4
Глава 2. Управление проектом и выполненные задачи	7
2.1. Управление задачами.....	7
2.2. Управление временем выполнения задач.....	8
2.3. Организация работы над базой данных и кодовой базой проекта.....	8
2.4. Выполненные задачи	9
2.5. Диаграмма прецедентов	13
Заключение	14
Приложение	16
Приложение 1	16
Приложение 2	18

Введение

Деятельность биржи труда предполагает работу с большими объёмами информации, в связи с чем целесообразно разработать информационную систему, позволяющую упростить работу с данными.

Разработка информационной системы, как правило, состоит из нескольких этапов и предполагает участие команды разработчиков. В связи с этим важной частью работы является управление проектом. Применение методологии управления проектами позволяет организовать деятельность по решению задач и достижению поставленных целей проекта.

Цель работы: разработка серверной составляющей информационной системы для Регионального центра содействия трудоустройству и адаптации к рынку труда выпускников.

Для выполнения поставленной цели необходимо выполнить следующие задачи:

- определить функции системы в соответствии с предметной областью;
- исследовать различные методологии управления проектами и выбрать методологию, подходящую для разработки системы;
- разработать компоненты серверной составляющей информационной системы в соответствии с техническим заданием;
- провести тестирование компонентов системы.

Объектом является процесс взаимодействия соискателей и работодателей с операторами Центра с целью трудоустройства (поиска работы или персонала).

Предметом является информационная система студенческой биржи труда.

Глава 1. Описание проекта и выбор методологии управления проектом

1.1. Предметная область

Региональный центр содействия трудоустройству и адаптации к рынку труда выпускников организует биржу вакансий. Вакансии поступают от организаций и предприятий. Студенты обращаются в Центр с целью поиска работы. Соискатели вакансий самостоятельно или через операторов составляют резюме, которые проверяются операторами и просматриваются работодателями. Специалисты Центра (операторы) подбирают подходящие вакансии.

1.2. Методология управления разработкой

Существует большое количество моделей управления проектами: каскадная, итеративная, спиральная, гибкая методология (Agile) и др.

Каскадная модель — модель процесса разработки программного обеспечения, в которой процесс разработки выглядит как поток, последовательно проходящий фазы анализа требований, проектирования, реализации, тестирования, интеграции и поддержки.

Итеративный подход — это выполнение работ параллельно с непрерывным анализом полученных результатов и корректировкой последующих этапов работы. Проект при этом подходе в каждой фазе развития проходит повторяющийся цикл: Планирование — Реализация — Проверка — Корректировка.

Спиральная модель представляет собой процесс разработки программного обеспечения, сочетающий в себе как итеративность, так и этапность. Отличительной особенностью этой модели является специальное внимание рискам, влияющим на организацию жизненного цикла. Каждый виток спирали соответствует созданию фрагмента или версии программного обеспечения, на нём уточняются цели и характеристики проекта, определяется его качество и планируются работы следующего витка спирали. Таким образом углубляются и последовательно конкретизируются детали проекта и

в результате выбирается обоснованный вариант, который доводится до реализации.

Гибкая методология разработки — обобщающий термин для целого ряда подходов и практик, основанных на ценностях Манифеста гибкой разработки программного обеспечения и 12 принципах, лежащих в его основе.

Большинство гибких методологий нацелены на минимизацию рисков путём сведения разработки к серии коротких циклов, называемых итерациями, которые обычно длятся две-три недели. Каждая итерация сама по себе выглядит как программный проект в миниатюре и включает все задачи, необходимые для выдачи мини-прироста по функциональности: планирование, анализ требований, проектирование, программирование, тестирование и документирование. Хотя отдельная итерация, как правило, недостаточна для выпуска новой версии продукта, подразумевается, что гибкий программный проект готов к выпуску в конце каждой итерации. По окончании каждой итерации команда выполняет переоценку приоритетов разработки.

Гибкая методология Scrum использует итеративный (регулярный повтор полного цикла работы над продуктом с непрерывным анализом результатов предыдущего этапа, корректировкой требований и процесса) и инкрементальный подходы (приращение результатов предыдущего этапа), чтобы управлять рисками и улучшать прогнозируемость.

Благодаря итерационной разработке, продукт можно выпускать быстро, с высокой периодичностью, а его качество можно постоянно улучшать, внося определенные доработки и корректировки. Таким образом, работоспособная и потенциально полезная версия продукта доступна в любой момент.

Kanban — метод управления разработкой, реализующий принцип «точно в срок» и способствующий равномерному распределению нагрузки между работниками. При данном подходе весь процесс разработки прозрачен для всех членов команды. Методика предполагает обсуждение производительности в режиме реального времени.

Для визуализации списка задач используется Kanban-доска — инструмент управления проектами, который помогает наглядно представить задачи, ограничить объем незавершенной работы и добиться максимальной эффективности (или скорости). С помощью карточек и столбцов на доске Kanban команды по техническим вопросам и сервисные команды могут понять, какой объем работы следует взять на себя, и выполнить этот объем, придерживаясь принципов непрерывного совершенствования.

Для работы над проектом выбрана методология Kanban, т.к. изначально были определены цели и задачи разработки, разработано техническое задание с определёнными требованиями.

Глава 2. Управление проектом и выполненные задачи

2.1. Управление задачами

Задачи были внесены в Kanban-доску в Trello — онлайн-сервис для управления проектами небольших групп.

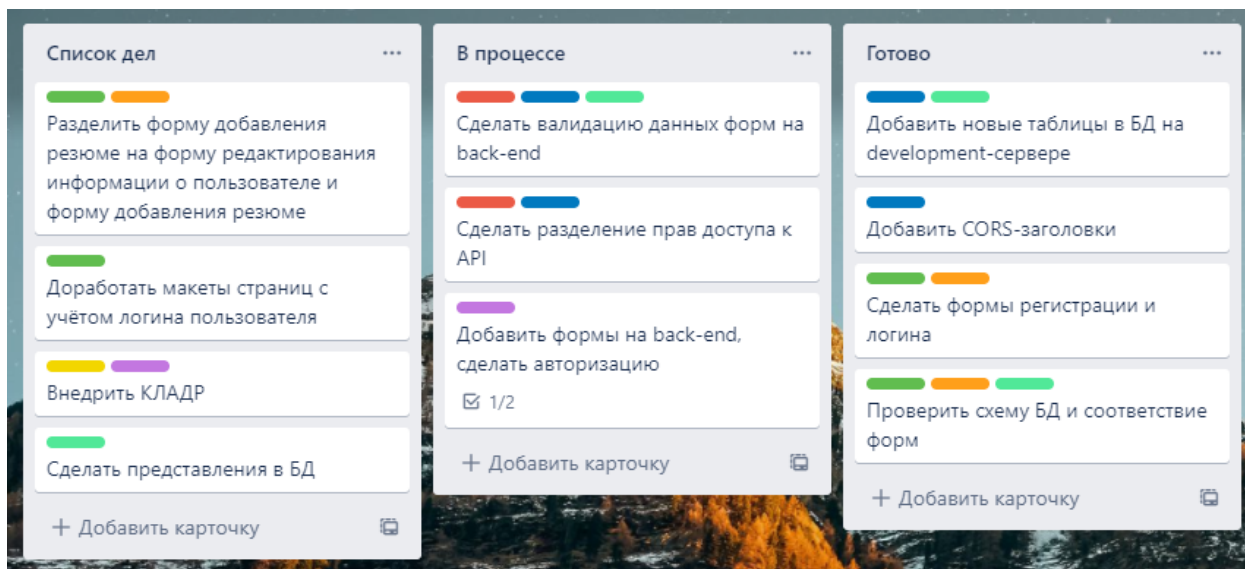


Рисунок 1 – Kanban-доска

Задачи были разделены на следующие группы:

Таблица 1 – группы задач

Метка	Описание
UX/UI	Пользовательский интерфейс, макеты страниц
front-end (JS)	jQuery, AJAX
front-end (вёрстка)	HTML, CSS
security	Безопасность REST API, разграничение прав доступа
back-end (website)	Серверная составляющая сайта
back-end (API)	CRUD, REST API
back-end (database)	База данных (структура, представления)

Метки позволяют структурировать задачи и улучшить визуальное восприятие. По ходу работы задачи вносились в колонку «Список дел», начало работы отмечалось перемещением карточки в колонку «В процессе», выполнение задачи отмечалось перемещением карточки в раздел «Готово».

2.2. Управление временем выполнения задач

Для визуализации времени выполнения задач применена диаграмма Ганта — популярный тип столбчатых диаграмм, который используется для иллюстрации плана, графика работ по проекту.

Задачи были распределены по времени и затем добавлены в Google-таблицу. Задачи отмечены разными цветами для удобства восприятия.

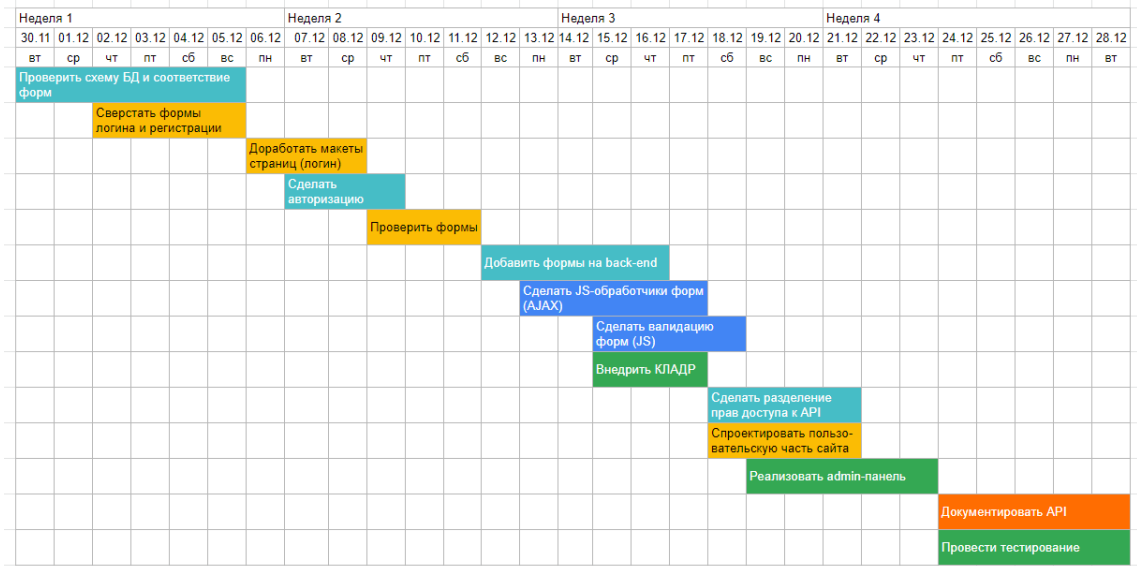


Рисунок 2 — диаграмма Ганта

2.3. Организация работы над базой данных и кодовой базой проекта

Структура базы данных, ER-диаграмма и кодовая база back-end были расположены в репозитории на GitHub.

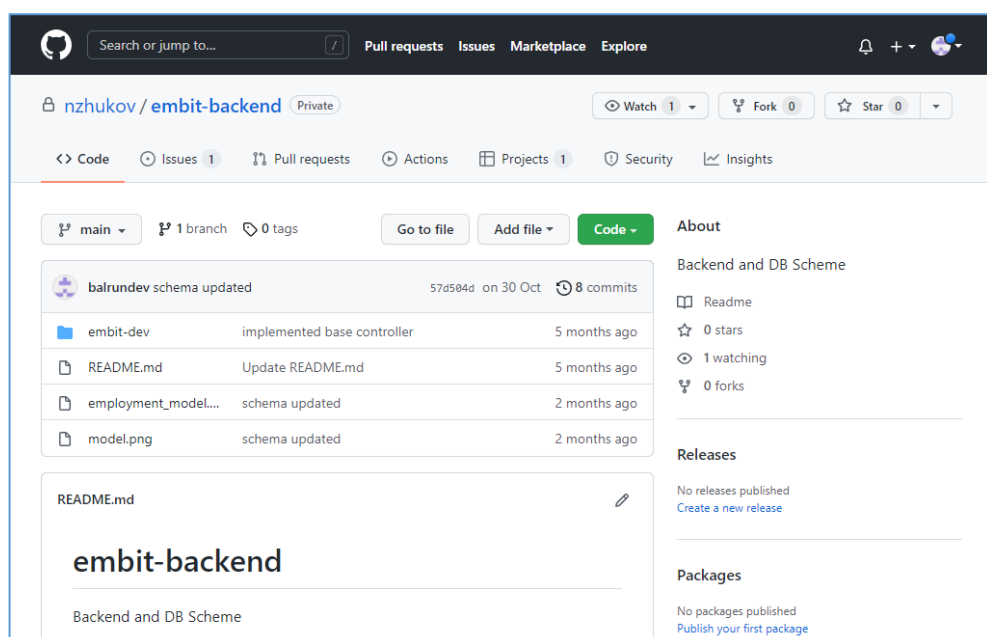


Рисунок 3 — страница репозитория в GitHub

2.4. Выполненные задачи

Система разрабатывалась в соответствии с техническим заданием, представленным в Приложении.

Разработка велась на основе разработанной ранее структуры базы данных и реализованного CRUD (REST API).

Серверная часть написана на языке программирования PHP и реализована с использованием Fat Free Framework 3.7.

В ходе работы над серверной составляющей были поставлены и решены следующие задачи:

- Проверка схемы базы данных;
- Обновление структуры базы данных на development-сервере;
- Добавление выдачи заголовков CORS;
- Реализация механизма валидации данных форм;
- Реализация защиты от XSS-атак;
- Реализация разделения прав доступа к API;
- Проведение модульного тестирования.

Проверка схемы базы данных

В ходе анализа были выявлены следующие недостатки структуры:

- имя пользователя содержится в одном атрибуте с типом недостаточной длины;
- таблица accounts не связана с таблицей users;
- таблица accounts не содержит информации о типе учётной записи;
- отсутствуют поля с информацией о должностных лицах компаний-работодателей.

Структура базы данных была доработана, обновлена ER-диаграмма. Обновленная структура представлена в Приложении.

Обновление структуры базы данных на development-сервере

Структура базы данных была обновлена для обеспечения возможности работы с back-end на сервере.

Добавление выдачи заголовков CORS

Для обеспечения возможности внешнего доступа к API были добавлены CORS заголовки Access-Control-Allow-Origin, Access-Control-Allow-Methods, Access-Control-Allow-Headers.

Реализация механизма валидации данных форм

Основная валидация данных должна быть реализована на стороне клиента. В ходе работы был реализован дополнительный уровень валидации на стороне сервера. Основным способом валидации – использование регулярных выражений.

Была реализована валидация данных для данных, добавляемых в таблицу company_hr. Определены следующие критерии валидности данных:

- фамилия, имя, отчество – состоят из символов русского алфавита,
- должность – может состоять из латиницы и кириллицы,
- номер телефона – от 5 до 12 цифр, включает код страны, код города, номер абонента.

Были составлены соответствующие регулярные выражения и реализована проверка данных, отправляемых методом POST. В случае невалидности данных запрос отклоняется с кодом HTTP 422, выдаётся объект JSON с описанием ошибки:

```
{
    "status": "failed",
    "error_string": "Data validity check error"
}
```

Рисунок 4 – ответ сервера при отправке невалидных данных

Реализация защиты от XSS-атак

В ходе реализации валидации данных было установлено, что API уязвим к атакам типа Cross-Site Scripting (XSS). Для исправления уязвимости было добавлено преобразование специальных символов в данных, поступающих на сервер, в HTML-сущности. Все данные из тела запроса преобразуются с помощью стандартной функции языка PHP - htmlspecialchars.

Реализация разделения прав доступа к API

Операции, связанные с изменением данных, недоступны для неавторизованных пользователей. В случае отправки запросов на добавление, изменение или удаление данных неавторизованным пользователем запрос отклоняется с кодом HTTP 401, выдаётся объект JSON с описанием ошибки:

```
{  
    "status": "failed",  
    "error": "Unauthorized"  
}
```

Рисунок 5 – ответ сервера в случае отправки данных неавторизованным пользователем

Введено разделение типов учётных записей пользователей.

Таблица 2 – типы учётных записей

account_type	Тип учётной записи
1	Администратор
2	Оператор
3	Пользователь (соискатель)
4	Пользователь (должностное лицо работодателя)

Реализовано следующее разделение прав доступа к данным:

Таблица 3 – права доступа к данным

Тип учётной записи	Операции	Доступные таблицы
Администратор	SELECT	Все таблицы
	INSERT, UPDATE, DELETE	Все таблицы
Оператор	SELECT	Все таблицы
	INSERT, UPDATE, DELETE	companies, vacancies, users, users_data, resumes, company_hr
Пользователь (соискатель)	SELECT	Все таблицы
	INSERT	users, users_data, resumes
	UPDATE	
	DELETE	resumes
Пользователь (должностное лицо работодателя)	SELECT	Все таблицы
	INSERT	companies, vacancies
	UPDATE	
	DELETE	vacancies

При попытке запроса к таблицам, доступ к которым запрещён в соответствии с правами доступа, он отклоняется с кодом HTTP 403, выдаётся объект JSON с описанием ошибки:

```
{  
  "status": "failed",  
  "error": "not allowed"  
}
```

Рисунок 6 – ответ сервера в случае попытки запроса к таблицам, доступ к которым запрещён в соответствии с правами доступа

Модульное тестирование

Разработаны тесты для проверки разделения прав доступа к API. Для выполнения тестов используется Test – модуль Fat Free Framework. Реализован интерфейс для отображения результатов тестирования.



Рисунок 7 – результаты тестирования модуля разделения прав доступа к API

2.5. Диаграмма прецедентов

Диаграмма прецедентов — диаграмма, отражающая отношения между акторами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

Диаграмма прецедентов информационной системы студенческой биржи труда:

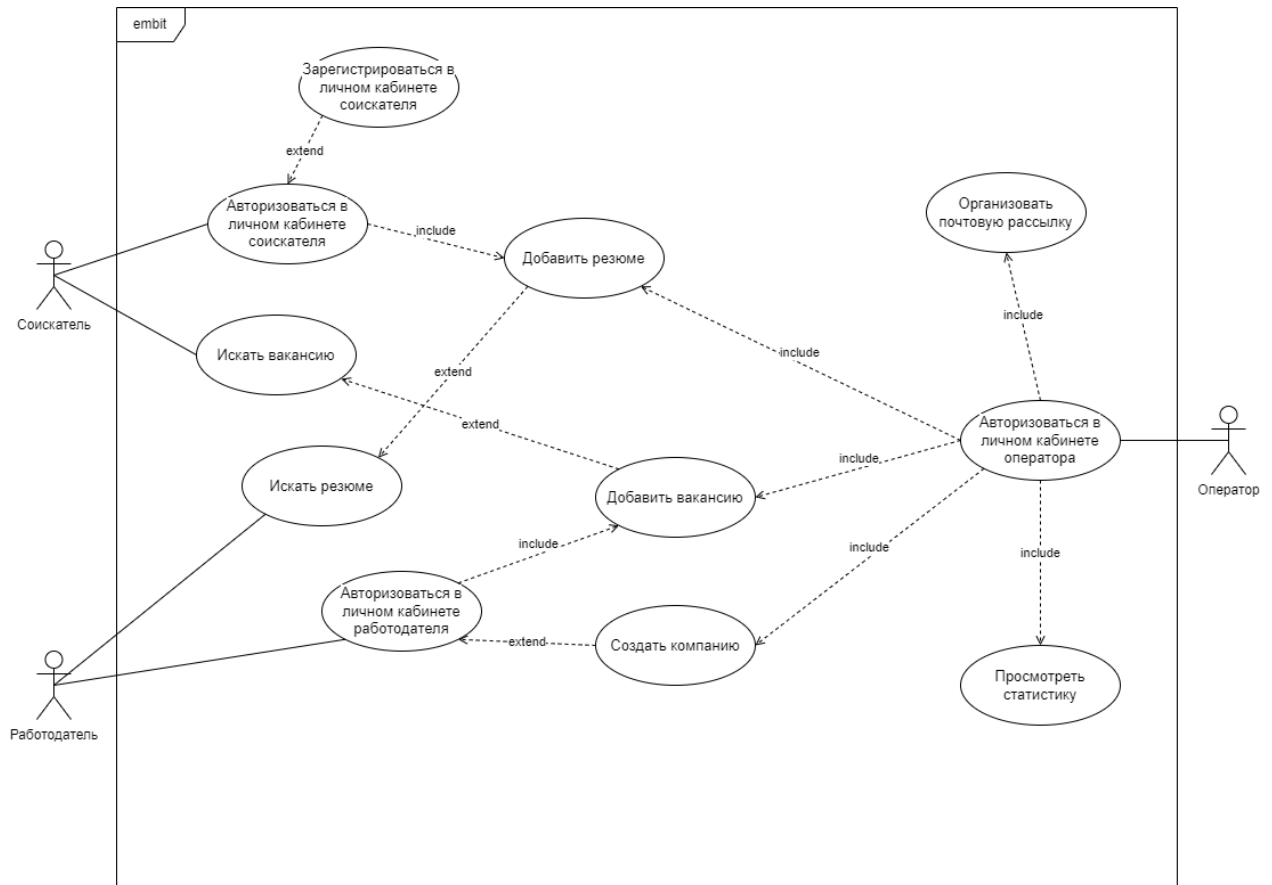


Рисунок 8 – диаграмма прецедентов

Заключение

В ходе работы были выполнены задачи по разработке серверной составляющей информационной системы для студенческой биржи труда.

Работа над проектом велась с использованием методологии Kanban. Использовались инструмент управления проектами Trello, диаграмма Ганта для управления задачами и контроля времени их выполнения.

В новой версии системы доработана структура базы данных, реализована валидация данных и разделение прав доступа к API.

Список источников

1. Иванов, Д. Моделирование на UML / Д. Иванов, Ф. Новиков. — Санкт-Петербург : НИУ ИТМО, 2010. — 200 с.
2. Лещева И.А., Страхович Э.В. Основы управления проектами / И.А. Лещева, Э.В. Страхович; Высшая школа менеджмента СПбГУ. — СПб.: Изд-во «Высшая школа менеджмента», 2011. — 96 с.
3. Управление программными проектами: учебник / Ю.П. Ехлаков. — Томск : Изд-во Томск. гос. ун-та систем управления и радиоэлектроники, 2015. — 216 с
4. Ручкин А.В., Трофимова О.М. Управление проектами: Основные определения и подходы // Вопросы управления. 2017. №3 (46). URL: <https://cyberleninka.ru/article/n/upravlenie-proektami-osnovnye-opredeleniya-i-podhody> (дата обращения: 21.12.2021).
5. Малиновская В.В. Методы управления проектами, актуальные для разработки программного обеспечения // Science Time. 2020. №4 (76). URL: <https://cyberleninka.ru/article/n/metody-upravleniya-proektami-aktualnye-dlya-razrabotki-programmnogo-obespecheniya> (дата обращения: 23.12.2021).

Техническое задание на разработку веб-системы для Центра содействия трудоустройству и адаптации к рынку труда выпускников учебных заведений

1. НАЗНАЧЕНИЕ И ЦЕЛИ СОЗДАНИЯ СИСТЕМЫ

1.1. Назначение системы

Назначением системы является автоматизация деятельности Регионального центра содействия трудоустройству выпускников.

1.2. Цели создания системы

- повышение эффективности деятельности Центра;
- приведение данных к единой форме;
- обеспечение возможности работы с резюме и вакансиями в электронном виде.

2. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

Разработка системы будет разбита на два этапа.

2.1. Первый этап

На первом этапе необходимо реализовать следующие разделы технического задания:

- Статические страницы (главная, контакты, партнеры, помощь, о системе, о бирже).
- Динамические страницы (поиск персонала, поиск вакансий, регистрация соискателей, регистрация компаний, добавление резюме, добавление вакансий).
- Личный кабинет оператора (авторизация, модерация резюме, модерация вакансий, модерация компаний).
- Личный кабинет соискателя.
- Личный кабинет работодателя.

2.2. Второй этап

На втором этапе необходимо реализовать:

- Создание системы почтовой рассылки для соискателей и работодателей.
- Создание механизма автоматического резервного копирования базы данных.
- Создание раздела статистики.

3. ТРЕБОВАНИЯ К ПРОГРАММЕ ИЛИ ПРОГРАММНОМУ ИЗДЕЛИЮ

3.1. Требования к функциональным характеристикам

Информационная система должна обеспечивать выполнение следующих задач:

- добавление резюме;
- добавление вакансии;
- поиск резюме и вакансий;
- представление информации в виде, удобном для обработки оператором.

Система должна предусматривать следующие роли пользователей:

- посетитель сайта;
- соискатель вакансии;
- главный оператор;
- оператор.

Таким образом, система может быть разделена на следующие части:

1. Публичная (в данный раздел входят статичные и динамичные страницы):
 - a. Статичные страницы (информационные страницы):
 - i. главная страница
 - ii. контакты
 - iii. партнеры
 - iv. помощь
 - v. о бирже
 - vi. о системе
 - b. Динамические страницы (содержащие в себе постоянно изменяющееся содержимое или вообще не относящиеся к информационным страницам):
 - i. статистика (по соискателям и работодателям)
 - ii. события (новости о бирже, партнерах и событиях в области поиска работы)
 - iii. поиск вакансий (работы)
 - iv. поиск персонала (резюме)
 - v. регистрация (соискателя/работодателя или компании)
 - c. Личный кабинет оператора:
 - i. сводка информации по вакансиям и резюме
 - ii. статистика по вакансиям и персоналу
 - iii. настройки оператора (смена логина, пароля, контактных данных и т.д.)
 - iv. почтовая рассылка
 - v. операции с резюме
 - vi. операции с вакансиями
 - vii. операции с компаниями
 - d. Личный кабинет соискателя
 - i. редактирование личной информации
 - ii. размещение резюме
 - e. Личный кабинет работодателя
 - i. редактирование информации о компании
 - ii. размещение вакансий

3.2 Требования к информационной и программной совместимости

Информационные структуры должны быть представлены в виде базы данных MySQL.

Система должна выполняться на сервере со следующим программным обеспечением:

- Apache 2
- PHP 5.6
- MySQL Server

Таким образом, система должна быть разработана на языке программирования PHP.

Обновление программного обеспечения в среде выполнения не представляется возможным.

Информационная система должна выполняться в среде с разделением прав доступа:

- запрещено изменение конфигурации сервера Apache
- отсутствуют права записи в системные директории, в т.ч. в /tmp.

Структура базы данных

