

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и Информационных Технологий

Кафедра «Информационные системы»

УТВЕРЖДАЮ

Зав. кафедрой ИС

_____ С. А. Виденин

подпись инициалы, фамилия

« _____ » _____ 2016 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии

Построение системы мониторинга распределения мест при поступлении
абитуриентов в СФУ ИКИТ

Руководитель

подпись, дата

Т.Н. Сизова

Выпускник

подпись, дата

А.Ю. Ежелый

Нормоконтролер

подпись, дата

Ю.В. Шмагрис

Красноярск 2016

РЕФЕРАТ

Выпускная квалификационная работа по теме «Построение системы мониторинга распределения мест при поступлении абитуриентов в СФУ ИКИТ» содержит 46 страниц текстового документа, 25 рисунков, 3 таблицы, 13 использованных источников.

РАЗРАБОТКА ПРОГРАММОГО ПРОДУКТА, БАЗА ДАННЫХ, СУБД, MVC, ШАБЛОНЫ АРХИТЕКТУРЫ, ПРИЕМНАЯ КОМИССИЯ, PHP

Объект работы – сайт, написанный на языке PHP

Цели проекта:

- анализ существующих систем при поступлении;
- анализ существующих систем «Абитуриент» других ВУЗов;
- построение требуемой системы.

В результате был создан сайт, с помощью которого как абитуриент, так и администратор, может отслеживать положение абитуриента в конкурсной таблице. Был получен ряд рекомендаций для дальнейшей поддержки сайта.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	5
1 Теоретическая часть.....	8
1.1 Приемная кампания 2016 года.....	8
1.1.1 Система приоритетов	8
1.1.2 Система, основанная на договорах	12
1.2 Сравнение систем мониторинга абитуриентов различных вузов.....	15
1.2.1 Система Сибирского Федерального Университета.....	15
1.2.2 АИС КГПУ	18
1.2.3 АИС СибГАУ	19
1.2.4 Оценка выбранных систем.....	21
2 Практическая часть.....	23
2.1 Основные шаблоны архитектуры ПО.....	23
2.1.1 Модель MVC	23
2.1.2 Модель MVP.....	25
2.1.3 Модель MVVM	27
2.1.4 Выбор шаблона	28
2.2 Обзор языков программирования	29
2.2.1 PHP	29
2.2.2 Ruby и Ruby on Rails.....	30
2.2.3 Python и Django	32
2.2.4 Выбор языка программирования.....	33
2.3 Диаграммы.....	34

2.3.1 Схема базы данных	34
2.3.5 Диаграмма развертывания	35
2.3.5 Диаграмма состояний	36
2.3.5 Диаграмма деятельности.....	36
2.3.5 Диаграмма прецедентов	37
2.4 Работа сайта. Реализация шаблона MVC	38
ЗАКЛЮЧЕНИЕ	44
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	45

ВВЕДЕНИЕ

Не секрет, что важнейший этап в жизни любого школьника, заканчивающего свое учебное заведение, не сколько государственные экзамены, сколько поступление в высшее учебное заведение. Сибирский федеральный университет производит поддержку школьников по всем направлениям: это и помощь с подготовкой к экзаменам в виде дополнительных курсов или иных способов подготовки будущих специалистов, а также как подготовка специалистов, которые в период приемной кампании будут работать в Университете, так и создание необходимого комплекса программного обеспечения, требующегося для работы и мониторинга текущей ситуации как сотруднику, так и самому абитуриенту.

Абитуриенту, в первую очередь, важно знать и отслеживать свои шансы на попадание на интересующую его специальность. Ранее для этого использовался сайт, на котором выводилась в не слишком удобной для понимания форме информация, отображающая текущую позицию абитуриента в рейтинге. Неудобства заключались как раз в отображении рейтинга: если отображение в рейтинге с аттестатом можно было понять, какое место занимает абитуриент и какие у него шансы на поступление, то рейтинг без учета аттестата выдавал в большинстве случаев неправдоподобную информацию, на которую нельзя было рассчитывать ни абитуриенту, ни работнику приемной комиссии, когда ему это было необходимо.

Для члена приемной комиссии неудобства заключались также в выводе информации об абитуриентах. Например, невозможно было вывести список всех абитуриентов, информацию об отдельном абитуриенте можно было просмотреть исключительно через pdf-файл, который печатался для личного дела, что невероятно тормозило работу и создавало очень много неудобств, а также в невозможности в удобном для понимания образе вывести текущую ситуацию по распределению мест по специальностям. Для этого приходилось использовать сторонние программы, которые также не всегда работали должным

образом в ассоциации с системой, с которой приходится работать во время приемной кампании.

Также проблемой стало то, что в приемной кампании 2016 года абитуриентов стали принимать в Университет по новым правилам приема. Вместо привычной системы приоритетов, хоть и имеющей свои недостатки, о которых речь пойдет в соответствующем разделе, да и которая, к тому же, использующейся повсеместно во множестве высших учебных заведений, а потому понятной и доступной для большей части абитуриентов, в этом году вводят систему договоров, с которой может сходу разобраться не только не каждый абитуриент, но и не каждый сотрудник приемной кампании, имевший честь работать в предыдущие года работы Университета, а также опыт и привычки от работы с прошлой системой.

Поэтому перед приемной комиссией встала задача построить такую систему, которая бы решала проблемы, приведенные выше. Данная система должна соответствовать актуальным правилам приема (система договоров и иные вещи, вроде предоставления дополнительных мест абитуриентам, поступающим из республики Крым и города-героя Севастополя, как было во время приемной кампании 2015 года).

Также нужно учесть, что в последствии с системой и ее внутренней структурой будут работать люди, возможно, незнакомые с ней, для чего нужно создать низкий порог вхождения в устройства системы. Нужно создать базу данных, а также саму систему, используя известные широкому кругу пользователей программы, языки и прочее.

Выбирая язык, на котором будет строиться система, как уже было сказано абзаце выше, нужно учитывать, что люди, которые впоследствии будут заниматься системой, могут не знать или не уметь работать на языке, с помощью которой построена система. Выбор языка в данном случае достаточно сложен, так как у каждого есть свои плюсы, но главные плюсы - понятно и низкое вхождение - стали решающим фактором. О выборе языка будет сказано в основной части данного документа.

Оценив приведенные проблемы, была начата разработка системы, отвечающая заданным требованиям и актуальности, а также подробное изучение материала, часть из которого была изучена, однако, во время обучения в Университете.

1 Теоретическая часть

1.1 Приемная кампания 2016 года

Как уже было написано во введении, приемная кампания для абитуриентов, которые поступают в Университет в 2016 году, претерпела значительные изменения. Можно даже больше и с уверенностью сказать, что она изменилась совершенно кардинально, так как от системы приоритетов, которой придерживались многие высшие учебные заведения по всей Российской Федерации, в нашем Университете решили отказаться. И для того, чтобы понять, как устроена нынешняя система поступления, следует, для начала, рассмотреть старую, более привычную систему приоритетов.

1.1.1 Система приоритетов

Данная система достаточно проста для визуального понимания, да и сама по себе достаточно просто устроена, но, на самом деле, таит в себе множество сложностей.

Перед тем, как с головой окунуться в приемную кампанию, абитуриент должен принести некоторый набор документов для того, чтобы он имел возможность участвовать в конкурсе, а также соответствовать некоторым условиям [7]:

1. Прохождение Единого государственного экзамена (ЕГЭ) по минимальным допустимым по правилам приема в ВУЗ баллам (в 2015 году в СФУ, например, нужно было набрать не менее, чем 31 балл по математике; абитуриенты, набравшие меньшее количество баллов, к участию в конкурсе не рассматривались). Баллы ЕГЭ, хоть и заносятся сначала вручную работником приемной комиссии в базу данных, но потом автоматически сверяются с онлайн-базой, подготовленной Министерством образования. Итоговые результаты считаются именно по базе Министерства.

2. Кроме минимальных баллов, абитуриент для поступления в ВУЗ обязан сдать определенные экзамены (в СФУ ИКИТ, например, кроме обязательных математики и русского языка, требуется информатика).

3. Оригинал или копия документа, подтверждающего успешное окончание образовательного учреждения и получение среднего или средне-специального образования (за редчайшим исключением, это аттестат либо его копия). Следует уточнить, что участвовать в конкурсе можно как по оригиналу, так и по копии аттестата, но при поступлении будут рассматриваться исключительно те кандидаты, которые имели в приемной комиссии ВУЗа на момент окончания приемной кампании оригинал аттестата.

4. Паспорт, чтобы сверить личность и занести данные в БД. Прием абитуриентов осуществляется только при наличии паспорта.

Далее наступает время системы приоритетов. Ее суть состоит в том, что абитуриент, желая поступить в высшее учебное заведение, выбирает из всего множества направлений всего три, при этом цифрами от 1 до 3 отмечая желаемый приоритет, который будет учитываться во время конкурса на места. Пока идет приемная кампания, абитуриент, если он видит, что не сможет поступить на определенную специальность, имеет право поменять свои приоритеты, естественно, не меняя свои конкурсные баллы.

После окончания приемной кампании, начинается зачисление абитуриентов на выбранные ими специальности. На каждое направление есть определенное количество конкурсных мест. Возникает несколько ситуаций:

- Если абитуриент имеет достаточное количество баллов и проходит по специальности, которую он выбрал с первым приоритетом, он поступает в ВУЗ. В противном случае, смотрится на его второй приоритет, и если он не поступает и по нему, то третий. Если абитуриент не проходит и по третьему приоритету, то он не поступает в ВУЗ.

- Если у абитуриентов, которые поступают на одну специальность, оказалось одинаковое количество баллов (они занимают одну строчку в

рейтинге), то для того, чтобы честно определить их положение в списке, используется система: выше в рейтинге будет тот, у кого больше баллов по приоритетному для поступления в ВУЗ предмету (в случае СФУ ИКИТ, это информатика), если баллы вновь совпали – то по второму предмету, и так далее. Если баллы совпали по всем трем предметам, то оцениваются индивидуальные заслуги абитуриента (участие в олимпиадах, спортивные достижения и т.д.).

- Также есть абитуриенты, у которых есть преимущественное право на зачисление. Это, например, победители определенных олимпиад, льготники, инвалиды, сироты, обладатели высоких наград в спорте и т.д. Преимущественные права дают весомые бонусы при поступлении, начиная от приоритета при одинаковом количестве баллов и заканчивая поступлением без экзаменов (например, победители всероссийских олимпиад).

Для того, чтобы разобраться в системе, следует рассмотреть пример.

Табл.3. ЗАЧИСЛЕНИЕ ПО ПРИОРИТЕТАМ

ВУЗ №1				ВУЗ №2				ВУЗ №3			
№	Фамилия	Балл	Приоритет	№	Фамилия	Балл	Приоритет	№	Фамилия	Балл	Приоритет
1	Дубен	205,4	1	1	Губко	203,0	3	1	Иванов	203,0	1
2	Дубинский	202,0	2	2	Скрипник	201,4	1	2	Рябый	200,8	1
3	Шах	198,7	2	3	Лысенко	201,0	1	3	Кулида	200,7	1
4	Ерема	197,0	1	4	Семенова	200,0	2	4	Кравченко	200,1	2
5	Сакун	196,5	1	5	Закревская	199,2	1	5	Губенко	198,5	1
6	Глуховский	195,3	1	6	Петренко	199,0	1	6	Небота	198,5	1
7	Сокол	195,2	2	7	Марыгина	197,3	2	7	Пухов	197,9	2
8	Пухов	195,0	1	8	Подольнюк	197,1	1	8	Черкив	197,3	3
9	Приходько	194,5	3	9	Фокив	196,0	3	9	Сокол	197,3	3
10	Губко	194,3	2	10	Дубинский	195,9	1	10	Губко	196,0	1
11	Шемет	193,9	2	11	Кравченко	195,1	1	11	Фокив	195,8	1
12	Семенова	193,6	1	12	Куцовый	194,5	2	12	Приходько	195,7	1
13	Фокив	193,2	2	13	Глуховский	193,9	3	13	Подольнюк	195,3	2
14	Небота	192,5	3	14	Сокол	193,7	1	14	Смолич	195,2	1
15	Куцовый	192,3	1	15	Шемет	192,6	1	15	Шах	194,9	1
16	Черкив	192,1	2	16	Небота	192,2	2	16	Шемет	194,0	3
17	Марыгина	190,0	1	17	Черкив	192,1	1	17	Глуховский	193,8	2
18				18	Приходько	191,4	2	18			
19				19	Смолич	190,5	2	19			

Рисунок 1 – Пример конкурсной таблицы системы приоритетов

Как видно из рисунка, абитуриенты Глуховский, Подольнюк и Пухов проходят по своим первым приоритетам, поэтому их остальные приоритеты не рассматриваются. Абитуриент Сокол не смог набрать необходимое количество баллов для своих первых двух приоритетов, но прошел по третьему. Также, так как Пухов уже проходит по первому приоритету на свою специальность, то

таблица смещается вверх, поэтому абитуриент Губко хоть и имеет первое место в рейтинге по второй специальности, но поступит на третью специальность последней в рейтинге, так как имеет первый приоритет на нее.

Также в ВУЗах абитуриенты зачисляются не сразу, а по системе «80 на 20». Иными словами, поступление делится на две «волны»: в первую волну попадают первые 80% из списка, во вторую – остальные 20%. Такая система помогает сразу отобрать тех, кто у кого достаточное количество баллов и отдан оригинал аттестата, а также дает возможность нерешившимся сдать оригинал или забрать его, дабы поступить в другой ВУЗ. Следует, однако, отметить, что 80% считается не от количества поступающих, а от количества оставшихся мест. Иными словами, если на специальность 30 мест, из которых 10 заняли льготники, то 80% будет считаться от 20 мест, а не полного количества.

Основные плюсы данной системы:

- Множество вариантов и возможностей при поступлении в ВУЗ, так как можно выбрать сразу три специальности и иметь возможность поступить по каждой из них.
- Динамически изменяемая ситуация в конкурсных списках, что дает внезапную возможность поступить тому, кто на поступление уже не рассчитывал;
- Простая для понимания, так как имеет под собой не особо сложную механику.

Но есть так же и серьезные минусы:

- Хаотичность, возникающая во время конкурсной борьбы. Динамически изменяемая ситуация, которая была описана как плюс, может так же являться и минусом, особенно для работников приемной комиссии, так как уследить за постоянно меняющимися приоритетами и позициями в конкурных списках невероятно сложно, особенно когда идет первая волна зачисления (те, кто проходят по первому приоритету, смещают таблицы, заставляя вновь

пересчитывать абитуриентов, которые поступили, после ее смещают те, кто поступил по второму и т.д.).

- Сложность с расставлением приоритетов у абитуриентов, имеющих средний среди остальных студентов балл ЕГЭ, так как существует возможность не поступить на ту специальность, куда он хотел.

1.1.2 Система, основанная на договорах

Данная схема во многом опирается на предыдущую, по приоритетам. Абитуриент, изъявивший желание поступить в университет, также должен принести определенный набор документов, описанный в прошлом пункте, все так же выбрать три направления, на которые он хочет поступить, все так же действует система 80/20, но на этом сходство и заканчивается.

После того, как абитуриент выбрал три (или менее) направления, он, вдобавок к остальным документам, должен написать согласие на поступление. Следует уточнить, что согласие подписывается не просто для поступления в Университет, а именно для поступления на интересующую абитуриента специальность. Если к концу приемной кампании у абитуриента будет достаточное количество баллов, он поступит именно на ту специальность, согласие на поступление к которой он подписал.

Иными словами, кроме аттестата, который всегда был обязательным для поступления, появился новый документ, без которого нельзя поступить – согласие (оно же договор) на поступление. Зачисление что во время первой волны приема, что во время второй происходит только среди тех абитуриентов, которые приложили к своим заявлениям оригинал аттестата и согласие на поступление.

Также у абитуриента есть возможность забрать свое согласие на поступление и подписать другое, на другую специальность. В сумме можно написать не более двух согласий, причем два согласия одновременно в действии быть не может. Кроме того, если абитуриент изъявляет желание

поменять выбранные направления, то работник приемной комиссии обязан это сделать, причем, в отличие от согласий, сделать так абитуриент может неограниченное количество раз.

Чтобы разобраться в данной системе, лучше всего разобрать небольшой пример. На три специальности – металлургия, горное дело и информатика – есть по два свободных места. Будем считать, что зачисление в данный ВУЗ идет в одну волну (зачисляются сразу все абитуриенты, а не по частям), а также, что данные абитуриенты прикрепили к заявлениям о приеме оригиналы аттестата и что во время приемной кампании они не будут менять свои согласия. Информация представлена в таблице 1.

Таблица 1 – Пример конкурсной ситуации с согласиями на поступление

Фамилия	Сумма баллов	Металлургия	Горное дело	Информатика
Петров	160		Согласие	
Иванов	161			Согласие
Сидоров	162		Согласие	
Кручинин	164			Согласие
Васечкин	159	Согласие		
Кустов	158			Согласие
Кружев	165		Согласие	

Так как на каждое направление всего по два свободных места, то примерный список поступивших на данные три направления будет выглядеть так, как представлено в таблице 2.

Таблица 2 – Пример результатов поступления с согласиями на поступление

Направление	Фамилия поступившего	Сумма баллов
Металлургия	Васечкин	159

Окончание таблицы 2

Направление	Фамилия поступившего	Сумма баллов
Горное дело	Кружев	165
	Сидоров	162
Информатика	Кручинин	164
	Иванов	151

Как видно из таблицы, на направление «Металлургия» произошел недобор – лишь один кандидат оставил согласие на поступление, поэтому второе место осталось невостребованным. Абитуриент Кустов не прошел на направление «Информатика», так как у абитуриентов Кручинина и Иванова, которые так же оставили согласия, было больше баллов. Аналогичная ситуация возникла у абитуриента Петрова. Если бы один из них поменял свой выбор и использовал возможность оставить второе заявление на направление, например, «Металлургия», кто-нибудь из них бы точно поступил.

Очевидно, что процесс зачисления в случае с согласиями на поступление приводит к кардинально иному результату, нежели при использовании системы приоритетов. Рассмотрим плюсы и минусы данного способа.

К плюсам можно отнести:

- Данный способ практически исключает чехарду, которая обычно возникает в конце приемной кампании, когда абитуриенты, которые имеют шаткую позицию в рейтинге, в спешке меняют приоритет.
- Данный способ невероятно удобен для мониторинга как работнику, так и студенту, так как в действительности гораздо очевидней, нежели способ с приоритетами.

Но есть очень большой минус для абитуриентов: шанс поступить становится гораздо меньше. Причина в том, что если у абитуриента всегда был шанс поступить по одному из трех направлений, то теперь он может поступить исключительно туда, куда он оставит согласие. Есть вероятность, что такая

система может даже отпугнуть потенциального студента, что отразится минусом на Университете.

1.2 Сравнение систем мониторинга абитуриентов различных вузов

1.2.1 Система Сибирского Федерального Университета

Одна из главных обязанностей членов приемной комиссии является заполнения электронных заявлений. Данные заявления формируются в автоматизированной информационной системе «Абитуриент». Система «Абитуриент» - это уникальная автоматизированная электронная база данных по всем абитуриентам СФУ, специально разработанная для проведения приемной кампании Сибирского федерального университета.

Эта система – огромная база данных, которая действует на территории всего Сибирского федерального университета. Она предназначена для:

- создания электронных личных заявлений абитуриентов;
- поиска абитуриентов в базе данных;
- получения сводных данных по количеству поданных заявлений и формирование отчетов;
- получения статистики для подготовки отчетов;
- создания собственных отчетов в Microsoft Excel и добавление их в программу.

К АИС «Абитуриент» имеют доступ абитуриенты, которым выдаются логины и пароли для просмотра рейтинга. Также доступ имеют и работники приемной комиссии для редактирования данных.

Важнейшая обязанность членов приемной комиссии – заполнение электронного заявления в соответствии с документами, которые принесли абитуриенты. В это заявление вводятся личные данные поступающего (Ф.И.О, данные паспорта, результаты вступительных испытаний и многое другое).

Следует помнить, что личные данные поступающего – конфиденциальная информация, и разглашать ее сторонним лицам без письменного подтверждения самого абитуриента строжайше запрещено, о чем всем членам приемной комиссии рассказывают на обязательном инструктаже.

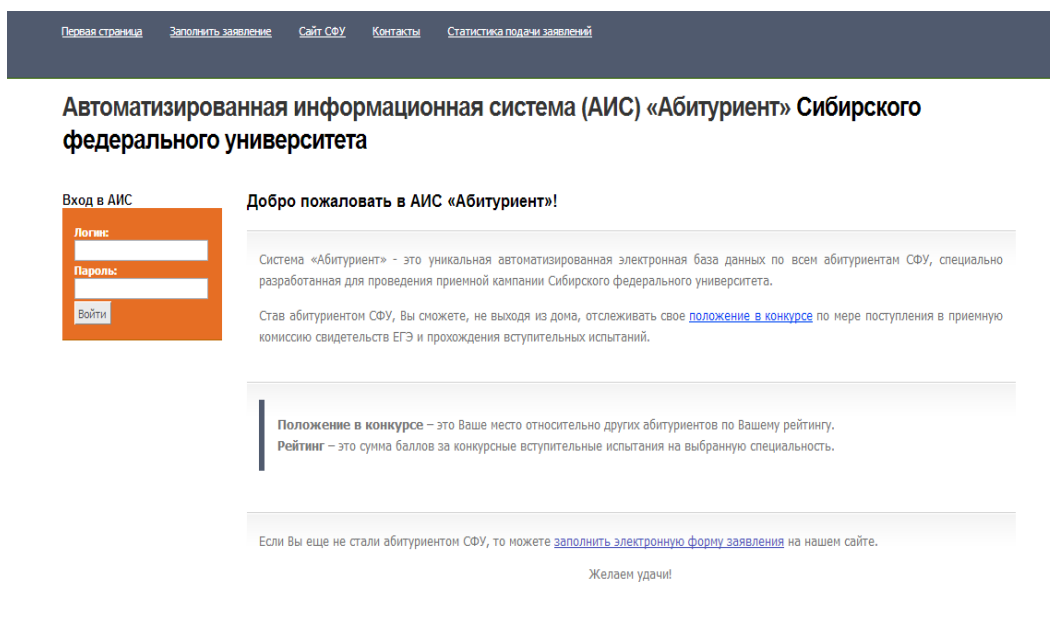


Рисунок 2 - Главная страница системы «Абитуриент» СФУ

На базе СПО/ВО	Направление	Институт	Очная	Зачочная	Очно-зачочная	Бюджет	Договор	Диплом олимпиады	Преимущество
<input type="checkbox"/>	01.03.01 Матем	ИМИФИ	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	На общих осно	<input type="checkbox"/>	Математика(2 ст.) 54. Отч	Не выбрано
<input type="checkbox"/>	08.03.01 Строи	ИСИ	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	На общих осно	<input checked="" type="checkbox"/>	Не выбран	Не выбрано
<input type="checkbox"/>	15.03.05 Констр	ХТИ СФУ	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	На общих осно	<input type="checkbox"/>	Не выбран	Не выбрано

Рисунок 3 - Бланк заявления системы «Абитуриент» СФУ

Преимущественное право зачисления:

Тип: дети-сироты | Другое: | Добавить

Подтверждающие документы:

Номер документа	Название документа	Дата выдачи	Кем выдан
		ДД.ММ.ГГТТ	

Способ возврата оригиналов поданных документов в случае непоступления на обучение/отзыва документов:
 Передача лицу, отозвавшему поданные документы, или доверенному л

Сведения о предоставленной медицинской справке:
 у-086 выдана Городской поликлиникой №1 г. Канска 18.04.2015

Необходимость создания специальных условий для проведения вступительных испытаний в связи с ограниченными возможностями здоровья или инвалидностью:
 Нуждаюсь Не нуждаюсь

Документы, подтверждающие необходимость создания специальных условий для проведения вступительных испытаний в связи с ограниченными возможностями здоровья или инвалидностью(наименование, кем, когда выдан, серия и номер при наличии):

Соотечественники

Основания: | Документы: |

Оригинал документа о предыдущем образовании предоставлен

Дата предоставления: ДД.ММ.ГГТТ | Дата заполнения: 17.06.2015

Вы можете распечатать бланк заявления только после сохранения изменений.
 Для печати Вам потребуется установленная на компьютере программа Adobe Acrobat Reader.

Сохранить | Распечатать | Закрыть окно

Copyright © 2007-2015

Рисунок 4 - Бланк заявления системы «Абитуриент» СФУ

При вводе данных в заявление следует быть крайне внимательным, так как изменить данные после отправки электронного заявления сможет лишь ответственный секретарь приемной комиссии. Поэтому, для исключения такого рода ошибок, абитуриент самостоятельно, после заполнения, должен проверить правильность данных и подписать, что он согласен с ними.

Далее бланки отправлялись на сервер, где обрабатывались системой. Также осуществлялась проверка введенных данных о ЕГЭ Федеральной базой сертификатов (ФБС), о которой было написано в предыдущем параграфе. Зачисление происходило исключительно по ФБС.

Абитуриенты же, в свою очередь, имеют доступ к своим данным через данный веб-сайт, после их регистрации техническим секретарем в базе данных. Благодаря данной возможности, каждый абитуриент, подавший заявление на поступление в один из институтов СФУ, имеет возможность отслеживать свое положение в рейтинге на поступление, используя свою уникальную учетную информацию: фамилия в качестве логина и серия и номер паспорта в качестве

пароля. Такая комбинация позволяет однозначно идентифицировать абитуриента.

С другой стороны, эта система имеет крайне неудобную систему поиска, которая не позволяет вывести, например, сводную информацию по всем абитуриентам на определенном направлении, а также не соответствует текущей системе поступления в СФУ.

1.2.2 АИС КГПУ

«АИС Абитуриент» (Автоматизированная информационная система «Абитуриент» КГПУ) представляет собой веб-сайт, включающий в себя инструменты для работы со специализированной защищенной базой данных, а также веб-формы для ее обновления и редактирования.

Для работы в данной системе в качестве администратора, необходимы учетные данные технического секретаря приемной комиссии.

АИС «Абитуриент» КГПУ

Введите ваше имя пользователя и пароль.

Имя пользователя

Пароль

КГПУ им. В.П. Астафьева, 2014.

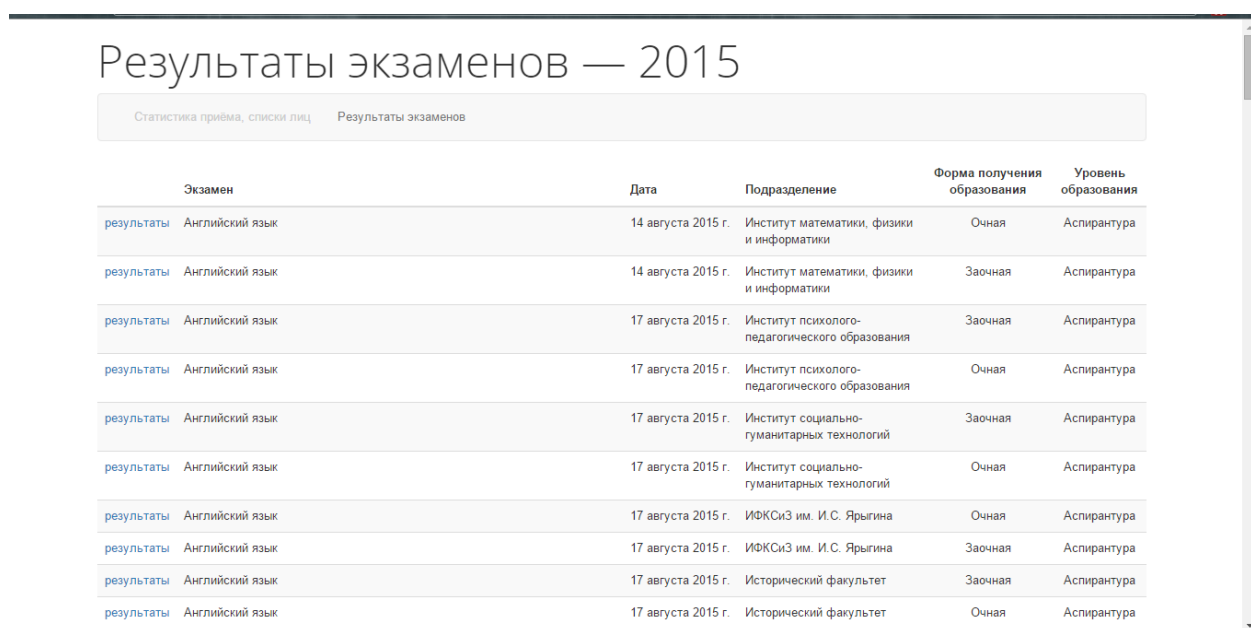
Рисунок 5 – Страница входа в АИС «Абитуриент» КГПУ

Исходя из общедоступных данных о АИС «Абитуриент» КГПУ, можно сделать вывод, что система работает на подобии АИС «Абитуриент» СФУ. На странице входа на веб-сайт пользователю предлагается ввести свои данные для

входа в систему. Дальнейшая работа с системой возможна, если пользователь является сотрудником приемной комиссии КГПУ или его абитуриентом.

Очевидно, что сотрудники приемной комиссии также имеют возможность добавлять данные в базу, на основе информации и документов поступающих, тем самым обновляя ее.

Кроме того, на главной странице АИС «Абитуриент» КГПУ размещена информация о результатах экзаменов поступающих по направлениям магистратуры.



Экзамен	Дата	Подразделение	Форма получения образования	Уровень образования
результаты Английский язык	14 августа 2015 г.	Институт математики, физики и информатики	Очная	Аспирантура
результаты Английский язык	14 августа 2015 г.	Институт математики, физики и информатики	Заочная	Аспирантура
результаты Английский язык	17 августа 2015 г.	Институт психолого-педагогического образования	Заочная	Аспирантура
результаты Английский язык	17 августа 2015 г.	Институт психолого-педагогического образования	Очная	Аспирантура
результаты Английский язык	17 августа 2015 г.	Институт социально-гуманитарных технологий	Заочная	Аспирантура
результаты Английский язык	17 августа 2015 г.	Институт социально-гуманитарных технологий	Очная	Аспирантура
результаты Английский язык	17 августа 2015 г.	ИФКСиЗ им. И.С. Ярыгина	Очная	Аспирантура
результаты Английский язык	17 августа 2015 г.	ИФКСиЗ им. И.С. Ярыгина	Заочная	Аспирантура
результаты Английский язык	17 августа 2015 г.	Исторический факультет	Заочная	Аспирантура
результаты Английский язык	17 августа 2015 г.	Исторический факультет	Очная	Аспирантура

Рисунок 6 – Страница «Результаты экзаменов» АИС «Абитуриент» КГПУ

1.2.3 АИС СибГАУ

Автоматизированная информационная система государственного аэрокосмического университета (АИС СибГАУ) имеет несколько другой функционал в сравнении с приведенными ранее системами.

АИС СибГАУ представляет собой веб-сайт, через который можно получить информацию о количестве поданных заявлений, рейтинге абитуриентов, а также другую полезную информацию для поступающих.

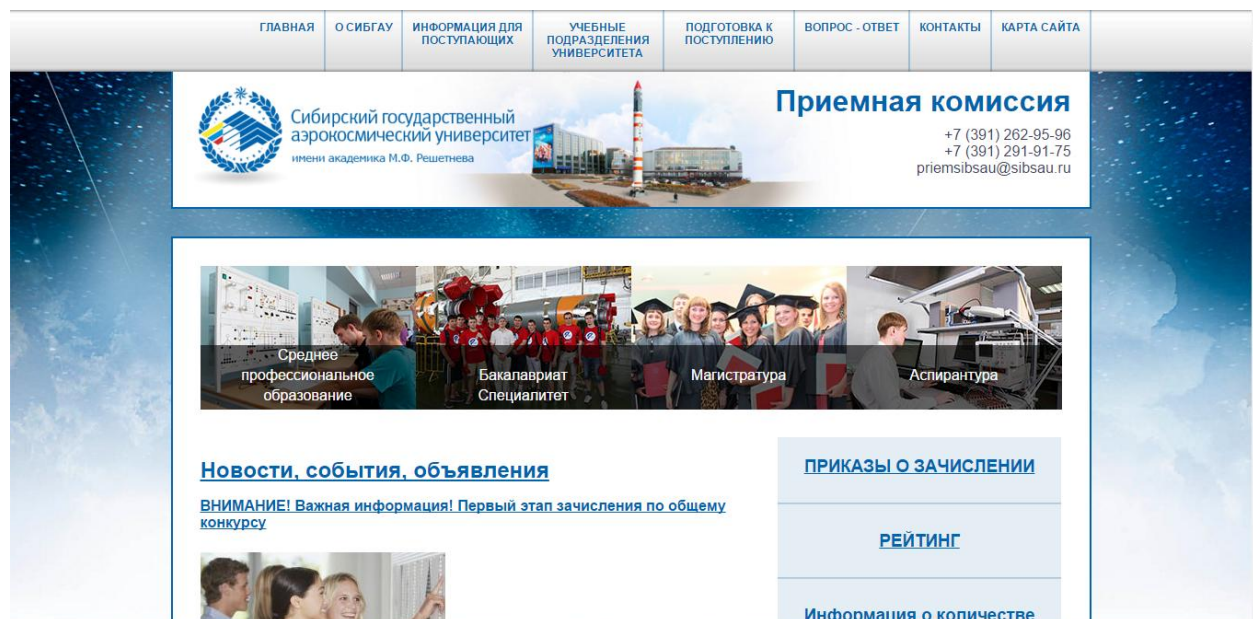


Рисунок 7 – Главная страница АИС «Абитуриент» СибГАУ

Можно сделать вывод, что редактирование базы данных, содержащей данные об абитуриентах СибГАУ, осуществляется администраторами системы через внутренние, недоступные для остальных, ресурсы. Иного доступа к ресурсам сайта у абитуриентов нет.

Администраторы системы с некоторой периодичностью обновляют данные на веб-сайте, в виде структурированной информации в PDF и EXCEL документах, что позволяет сотрудникам и абитуриентам получать актуальную статистическую информацию.

Обновлением информации администратором и стоит выделить как главный минус данной системы, ведь для того, чтобы увидеть свое конкурсное положение, пользователю приходится ждать обновления информации от администраторов, тогда как в предыдущих системах имелась возможность просмотра в любое удобное время, так как обновления производились автоматически.

1.2.4 Оценка выбранных систем

После сравнения вышеприведенных систем, можно сделать вывод, что принцип работы, на котором они основаны, отличаются, в той или иной степени. Система «Абитуриент» СФУ и Система «Абитуриент» КГПУ схожи в механизме работы с базой данных. Сотрудники приемной комиссии и абитуриенты данных университетов имеют возможность работать с системой напрямую, получая в итоге наглядную информацию, которую достаточно легко освоить.

В то же время АИС «Абитуриент» СибГАУ представляет собой закрытую площадку, и все данные выкладываются на сайт администраторами. Такая система, хоть и имеет право на жизнь, но является не совсем удобной по следующим причинам:

- Увеличение точности в электронных заявлениях и контроль за ними. Большим плюсом систем, в которых пользователю предоставляется возможность отслеживать свои данные, является то, что пользователь самостоятельно следит за правильностью этих данных и в случае, если он заметит какую-либо ошибку, то у него имеется возможность связаться с администратором системы, который, в свою очередь, исправит данную ошибку. На практике такие случаи встречаются достаточно часто: например, работник приемной комиссии, заполняя электронное заявление, сделал опечатку в фамилии абитуриента, а сам абитуриент, проверяя данные, также не обнаружил эту опечатку. В случае, если система бы была доступна исключительно администраторам, то опечатка могла бы дорого стоить при поступлении в университет, так как обнаружилась бы только к концу приемной кампании, при составлении списков.

- Возможность отслеживать свое положение в любое время, благодаря динамическому обновлению данных. Обновление системы происходит тогда, когда в базу данных заносится новый пользователь. Система в тот же момент

пересчитывает позиции абитуриентов и, когда определенному абитуриенту это необходимо, выдает ему его текущую конкурсную позицию. Это важный элемент системы, который требуется для удобного мониторинга информации.

Оценив данные системы, которые уже много лет используются в пределах ВУЗов Красноярска и других городов, была начата работа над системой, учитывающей вышеприведенные факторы.

2 Практическая часть

2.1 Основные шаблоны архитектуры ПО

2.1.1 Модель MVC

Часто при разработке любого ПО или веб-приложения, особенно когда программу или сайт разрабатывает команда высококвалифицированных людей, возникает проблема изменения уже написанного кода. А именно, что изменение одной, уже готовой части, может критически повлиять на работу другой, с первого взгляда, казалось бы, совсем не относящейся к изначальной. Например, изменяя что-либо в базе данных, нам придется менять полностью вывод информации по очевидным причинам.

Программисты сформировали такую проблему: как отделить бизнес-логику приложения, в которой происходят основные действия и вычисления, от пользовательского интерфейса так, чтобы изменение одной части не влияли на другую? Иными словами, требовалось создать такую архитектуру, которая уменьшила бы связанность модулей до минимально возможного значения. Схему, решающую эту проблему, описал Трюгве Реенскауг в 1979 году в статье «Applications Programming in Smalltalk-80: How to use Model-View-Controller». Как видно из названия статьи, такую схему построения программ он назвал «Модель – представление – контроллер» [9].

В основе этой модели заложена идея разделения представления данных, их обработку и сами данные на три части [1, 3].

- Модель (Model) – хранит в себе бизнес-логику приложения, содержит некие данные, действия и прочее, с чем будет впоследствии работать пользователь.

- Представление (View) – наоборот, данные само по себе не хранит, но знает, как информацию представить и вывести на экран, будь то в виде таблицы, текста, изображения и пр.

- Контроллер (Controller) – отвечает за обеспечение связи между представлением и моделью. Обрабатывает запросы, поступающие из представления, и отправляет их в модель.

Чтобы лучше понять устройство, следует обратиться к графическому описанию данной модели:

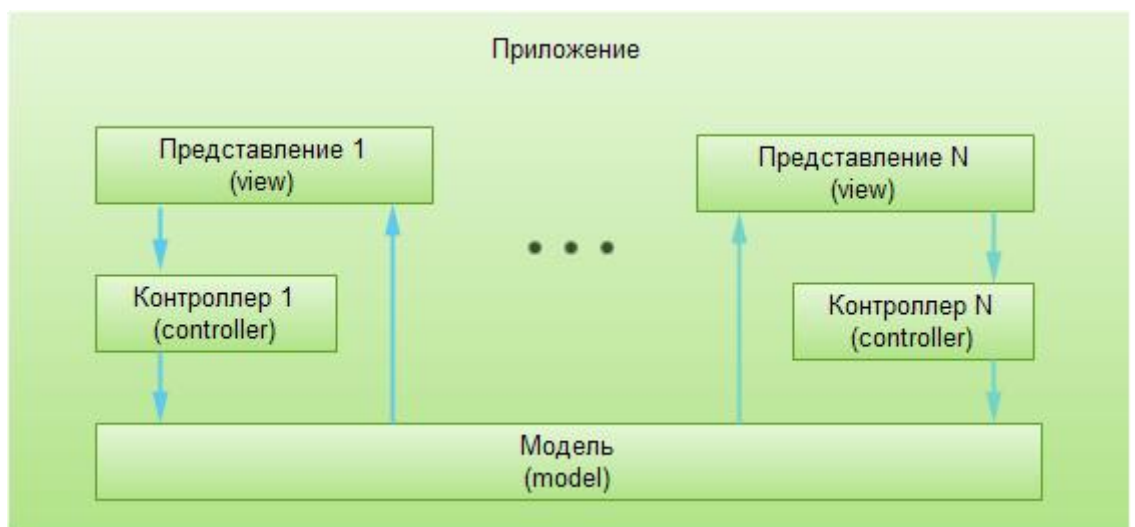


Рисунок 8 – Графическое представление модели MVC

1. Пользователь хочет вызвать какую-либо информацию на экран. Он видит то, что отображает модуль View.

2. Пользователь отправляет запрос, который обрабатывает контроллер.

3. Контроллер отправляет данные в модель, которая уже отправляет требуемые данные в View.

4. Модуль View получает данные и отображает их на экран требуемым образом.

Важно отметить, что зависимость в модели MVC односторонняя: и представление, и контроллер зависят от модели, но модель не зависит ни от контроллера, ни от представления. Таким образом, становится понятно, что

модель можно разрабатывать независимо от визуального представления, а также то, что к любой модели можно создать несколько визуальных представлений.

Следует выделить основные плюсы схемы MVC [4,5]:

- очень низкая связанность модулей. Так как разные модули имеют разные цели, то, например, дизайнер, разрабатывающий представление, может совсем не знать, как устроен код модели. И наоборот, программист, составляющий модель, может не знать, как будет выглядеть конечный продукт. Из этого следует, что при создании MVC-приложения можно нанять высококвалифицированных специалистов в узком плане;

- из предыдущего пункта следует еще один важный вывод: благодаря низкой связанности кода, изменение одной части программы не будут как-либо касаться другой ее части.

- модель MVC не привязана ни к какому языку программирования. Модель можно построить на любом языке. Некоторые, вроде Ruby on Rails, уже держат в основе эту модель;

- огромные возможности по повторному использованию кода, что сильно его упрощает. Следует помнить про возможности инкапсуляции, так как модель может инкапсулировать другие модели. В идеале, контроллер должен иметь всего пару строчек кода, которые перенаправляют запросы к другим элементам системы.

2.1.2 Модель MVP

Также существует шаблон MVP – Model-View-Presenter. В отличие от классического MVC, контроллер, который тут зовется представителем (Presenter), сам дает знать представлению о изменениях [12].

Чтобы подробнее разобраться в устройстве этого шаблона, следует обратить внимание на схемы, показанные на рисунках 3 и 4.

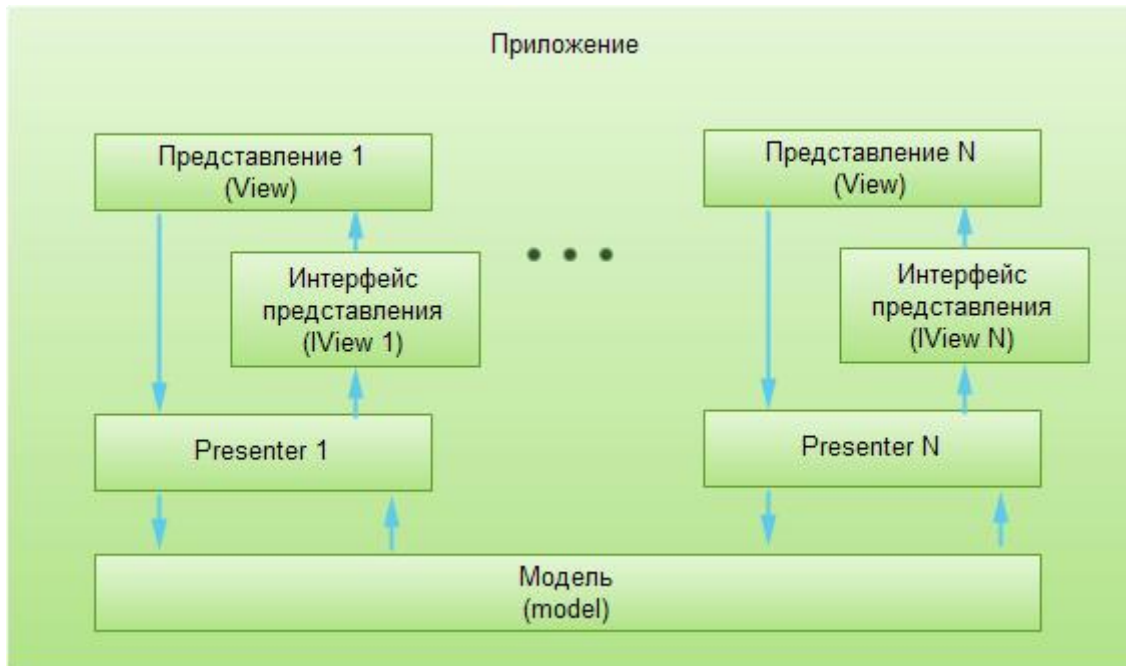


Рисунок 9 – Схема MVP

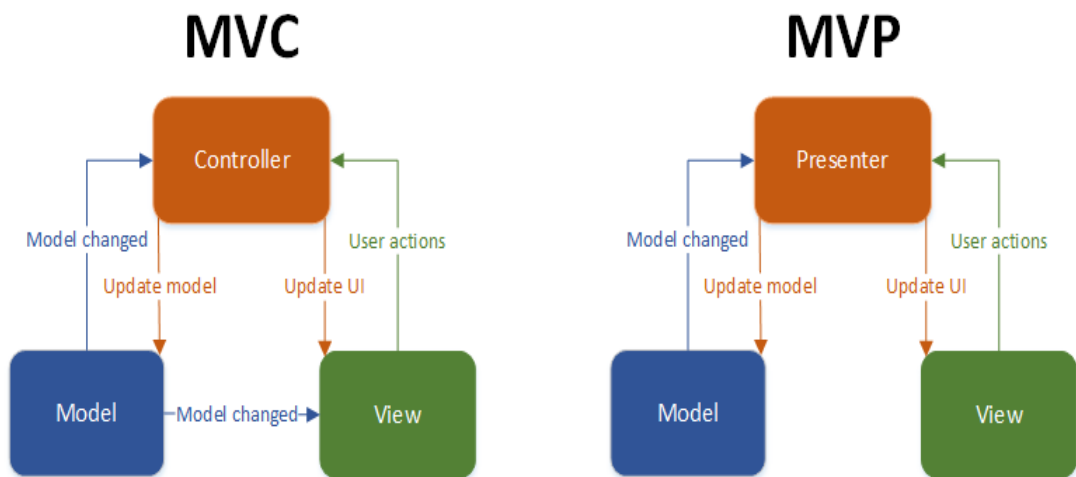


Рисунок 10 – Сравнение MVC и MVP

Как показано на рисунках, в отличие от MVC, где модель напрямую отправляет информацию представлению, в MVP информация сначала отправляется представителю. Представитель же отправляет информацию представлению через дополнительный интерфейс, который может пригодиться для тестирования приложения.

Самому представлению нет надобности получать информацию. Удобство MVP состоит в том, что оно позволяет «разгрузить» модель, переместив логику представления в модуль Presenter. Также следует добавить, что представление, как и в MVC, не зависит от модели, поэтому у каждой модели может быть множество представлений (но также и множество представителей). У каждого представления будут интерфейсы с определенными наборами методов и свойств, необходимых представителю.

Подход удобен тем, что позволяет использовать методологию TDD, так известную как «разработка через тестирование», когда сначала пишется тест, а потом уже логика, которая пройдет этот тест. Данный подход, как и следующий по списку, используется, в основном, при построении WPF-приложений.

2.1.3 Модель MVVM

Шаблон, разработанный на основе MVC. Основным отличием от классической MVC является то, что взаимодействие между компонентами идет не только через контроллер, как в MVC, но и в обратном порядке. Такие ухищрения сделаны для того, чтобы использовать т.н. «связывание данных» - процесс установления связи между графическим интерфейсом и бизнес-логикой приложения. Такой процесс используется в современных технологиях, вроде WPS или Silverlight [10].

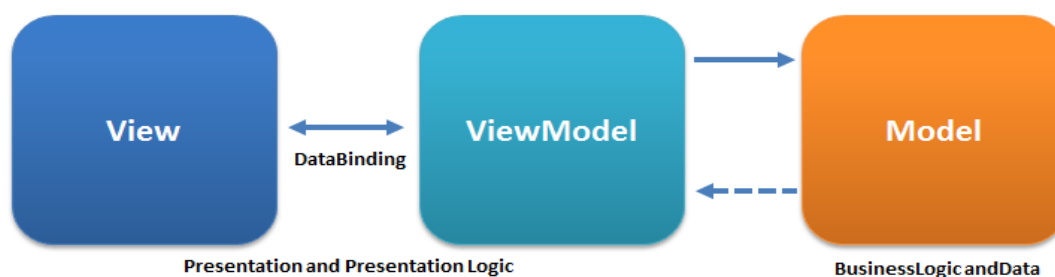


Рисунок 11 – Схема MVVM

- Модель – так же, как и в MVC, содержит в себе данные, необходимые для работы системы (часто это БД или репозиторий).

- Представление – аналогично представлению MVC. Содержит в себе графический интерфейс.

- Модель представления – берет на себя больше функций, чем контроллер в MVC. В схеме MVVM модель представления содержит и модель, которая преобразована для представления, и также содержит набор управляющих команд, с помощью которых можно запрашивать или изменять данные, находящиеся в модели.

Из плюсом MVVM можно выделить:

- Великолепная основа для модульного тестирования. Разрабатывая MVVM-приложения, у разработчика всегда будет возможность тестировать программу по модулям, а не всю сразу. К тому же, появляется, как и в случае с MVP, написать сначала тест, и только затем логику, которая пройдет этот тест.

- Меньшее количество кода, а также больше возможностей для его рефакторинга.

- Разделение обязанностей разработчика бизнес-логики и дизайнера, так как работа одного точно не будет влиять на работу другого.

2.1.4 Выбор шаблона

Рассмотрев все доступные варианты, выбор почти сразу же пал на шаблон MVC и на это есть весомые причины:

- Низкая сложность реализации.

- Большая ориентированность на веб-разработку, в отличие от MVP и, тем более, MVVM, который в вебе предпочтительней не использовать.

Также следует упомянуть, что модель MVC наиболее известна и распространена, поэтому, если у администраторов, обслуживающих сайт,

возникнут проблемы с работой сайта, разобраться в устройстве данного шаблона будет гораздо легче, чем с иными шаблонами.

2.2 Обзор языков программирования

На сегодняшний день существует немало способов, с помощью которых можно создавать веб-приложения. Более того, существуют готовые фреймворки, благодаря которым на создание сайта уйдет всего пара кликов мышки. Однако, наиболее популярными до сих пор остаются три языка программирования, с помощью которых создаются веб-приложения: PHP, Ruby и Python [8].

2.2.1 PHP

PHP (PHP: Hypertext Preprocessor) на сегодняшний день является наиболее популярным скриптовым языком для разработки веб-приложения. Создан в 1994 году датчанином Расмусом Лерфордом. История этого языка началась с создания самых простых сайтов (что заложено в изначальном названии – «Personal Home Page Tools») и продолжается до сих пор такими сайтами, как Wikipedia, Google и другие. Множество корпораций выбирает этот язык, когда требуется разработать веб-приложение и вот почему:

- свободен в плане лицензии, имеет открытый код;
- прост в освоении, что подкрепляется большим подспорьем материала для обучения и большой пользовательской базой;
- имеет развитую поддержку баз данных;
- непривередлив к серверу, поэтому сайт, написанный на PHP, можно разместить почти на любом сервере;
- по синтаксису очень похож на не менее популярные Perl и C/C++, что лишь упрощает работу пользователю;

- огромное количество подключаемых библиотек, позволяющих настраивать сайт «под себя».

- на текущий момент PHP используется практически на любой платформе, будь то смартфоны, handheld-девайсы и другие устройства.

Но в каждой бочке меда найдется ложка дегтя. PHP, к сожалению, также имеет оные:

- изначально не предназначен для работы с MVC, что не мешает, однако, исправить эту проблему вручную;

- предназначен исключительно для разработки веб-приложений и только;

- отсутствует поддержка многопоточности.

Подводя итог, PHP является очень хорошим выбором как для начинающего веб-программиста, так и для профессионала. Если от приложения не будут требоваться какие-либо диковинные функции (та же многопоточность), то этот язык будет оптимальным при выборе.

2.2.2 Ruby и Ruby on Rails

Язык программирования за авторством японца Мацумото Юкихиро, увидевший свет в 1995. Создавался на основе языка Perl и задумывался как «Настоящий ООП-язык». Именно на этом языке написан популярнейший фреймворк для создания веб-приложений Ruby on Rails, который также рассматривался для создания требуемого веб-приложения. Данный фреймворк обладает очень большими возможностями, но самое главное – что в его сути, уже по умолчанию реализуется шаблон MVC.

Следует сначала рассказать про плюсы данного фреймворка:

- открытая разработка, а также поддержка многих платформ и систем;

- код, как и в PHP, гармонично внедряется в HTML-разметку;

- реализует концептуально чистую объектно-ориентированную парадигму, основывается на шаблоне MVC;

- предоставляет продвинутые методы манипуляции строками, текстом и массивами (начиная с простых команд, вроде reverse);

- легко интегрирует серверы баз данных, вроде MySQL, для последующего использования;

- простой, чистый и понятный без особых знаний о программировании синтаксис (команда reverse, например, как уже было сказано выше, отражает текст задом наперед) значительно облегчает программистам первые шаги в обучении этому языку;

- возможность создания многопоточных приложений.

- простой, чистый и понятный без особых знаний о программировании синтаксис (команда reverse, например, как уже было сказано выше, отражает текст задом наперед) значительно облегчает программистам первые шаги в обучении этому языку;

- возможность создания многопоточных приложений;

Из минусов этого замечательного языка можно выделить:

- материалов, книг и прочих источников, из которых можно почерпнуть информацию по этому языку, особенно в русском сегменте Сети, крайне мало, отчего знакомство с ним – достаточно сложный процесс;

- хоть и при изучении основ этого языка он может показаться простым, но материал уровня выше уже крайне сложно понять новичку, что вкупе с предыдущим пунктом сразу убивает желание изучать этот язык у большинства новичков;

- из этого следует один еще один, пожалуй, главный минус – администраторы, которые впоследствии будут работать с сайтом, могут не знать синтаксис данного языка, поэтому им будет сложно работать и осуществлять поддержку сайта.

2.2.3 Python и Django

Язык, разработка которого началась еще в конце 1980-х годов, голландцем Гвидо ван Россумом. Как и Ruby, Python следует принципам уменьшения кода ради его читабельности. Но мало – не значит плохо. Это очень мощный язык, со своими плюсами и минусами.

Django же – популярный фреймворк, в основе которого также лежит язык Python. Как и Ruby on Rails, в саму идею этого фреймворка заложен шаблон MVC. Важное отличие от фреймворка на Ruby – сайт на Django создается из приложений, которые следует делать подключаемыми и отчужденными. Основной принцип построения строится на идее того, что если один модуль отключить, то остальные полностью работать не перестанут.

Следует выделить плюсы этого фреймворка:

- довольно прост в изучении на начальном этапе, чему также способствует большое и позитивное сообщество, которое всегда готово помочь новичкам;
- особенности синтаксиса дают программисту стимул писать читабельный код;
- множество полезных библиотек и расширений языка;
- изначальная основа на шаблоне MVC;
- абсолютно всё в Python является объектами в смысле ООП, но при этом объектный подход не навязывается программисту.

Недостатки Python:

- не слишком удачная поддержка многопоточности;
- изначальная ограниченность средств для работы с базами данных;
- несколько меньшая производительность по сравнению с другими фреймворками;
- профессионалов, которые могли бы обслуживать сервер и иметь знания об этом фреймворке, не так уж и много;

- не слишком удобные возможности для расположения сервера, так как Django очень требователен к нему.

2.2.4 Выбор языка программирования

Выбирая между тремя вариантами – PHP, Django и Ruby on Rails, выбор был остановлен на PHP по следующим причинам:

1. Простота в установке сайта на сервер. Не каждый хостинг поддерживает Django и Ruby, но очень сложно найти сервер, который бы не поддерживал PHP - распространенность языка и его популярность играют здесь немаловажную роль. Существует вариант не покупки хостинга, а установке на локальный сервер Университета, но, однако, также неизвестно, как будет работать сайт, написанный не на PHP.

2. Наличие специальных знаний у людей, обслуживающих сайт. Так как сайт будут обслуживать также сторонние люди, не участвующие в его разработке, то эти люди должны либо обладать знаниями о хотя бы синтаксисе языка, либо должны обучиться этому в короткие сроки, так как обслуживание сайта, структуру которого ты не понимаешь, лучше не обслуживать вообще. Однако, так как обучение в кратчайшие сроки невозможно (это как материальные затраты на обучение, так и слишком большие затраты по времени), то следует выбрать не сложный язык, вроде Ruby, а простой PHP, в основе которого синтаксис Perl и C/C++.

Более наглядная и краткая информация представлена в таблице 3.

Таблица 3 – Сравнение языков программирования и фреймворков

	PHP	Ruby on Rails	Django
Порог вхождения в язык	Низкий	Низкий	Низкий
Сложность в дальнейшем обучении	Низкая	Очень высокая	Высокая

Окончание таблицы 3

Простота в установке и дальнейшей поддержке	Очень просто	Сложно	Сложно
Изначальная поддержка MVC	Отсутствует	Отсутствует	Отсутствует

2.3 Диаграммы

UML— язык графического описания для объектного моделирования в области разработки программного обеспечения, моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

UML является языком широкого профиля, это — открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML-моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. UML не является языком программирования, но на основании UML-моделей возможна генерация кода.

UML позволяет также разработчикам программного обеспечения достигнуть соглашения в графических обозначениях для представления общих понятий (таких как класс, компонент, обобщение, агрегация и поведение) и больше сконцентрироваться на проектировании и архитектуре.

2.3.1 Схема базы данных

Схема БД – это все поля (как ключевые, так и нет), также связи между ними. Данная схема нужна для того, чтобы наглядным образом понимать, какие поля присутствуют в БД, как осуществляется связь между таблицами, какие значения могут принимать поля, какие поля являются ключевыми;

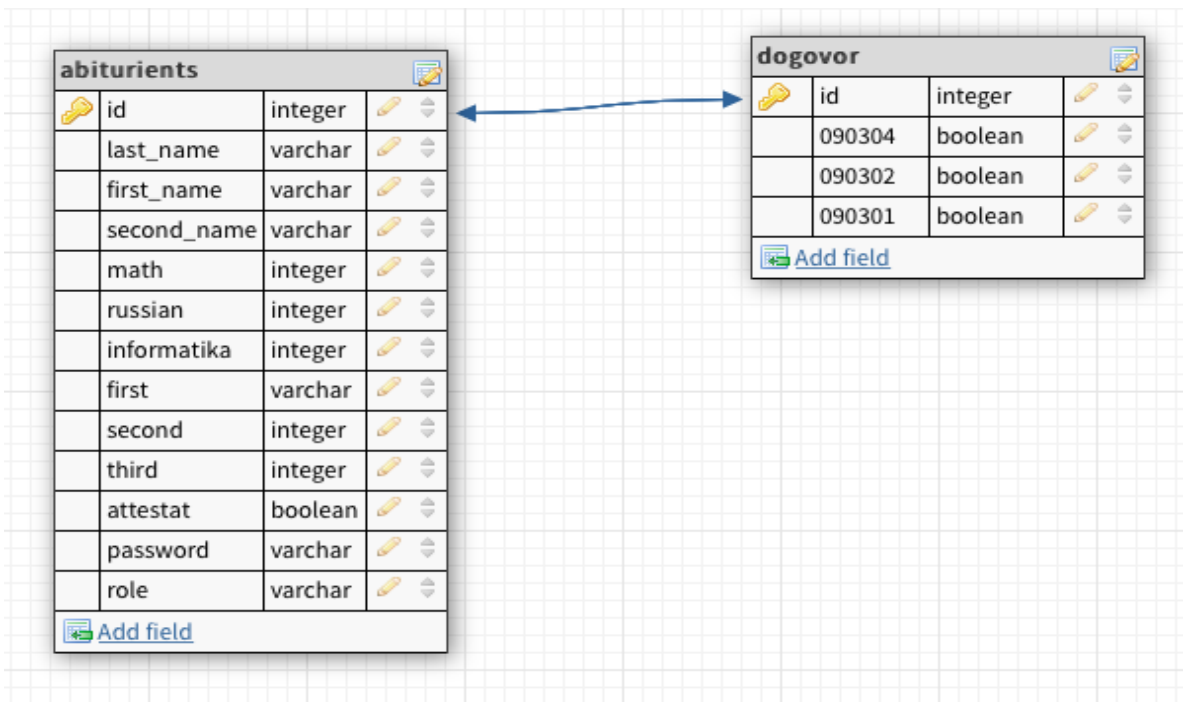


Рисунок 12 – Схема БД

2.3.5 Диаграмма развертывания

На диаграмме развертывания показано, какие аппаратные компоненты существуют (Система, Абитуриент, База данных), какие программные компоненты работают на каждом узле (Браузер, интерфейс сайта и прочее)), и как различные части этого комплекса соединяются друг с другом.

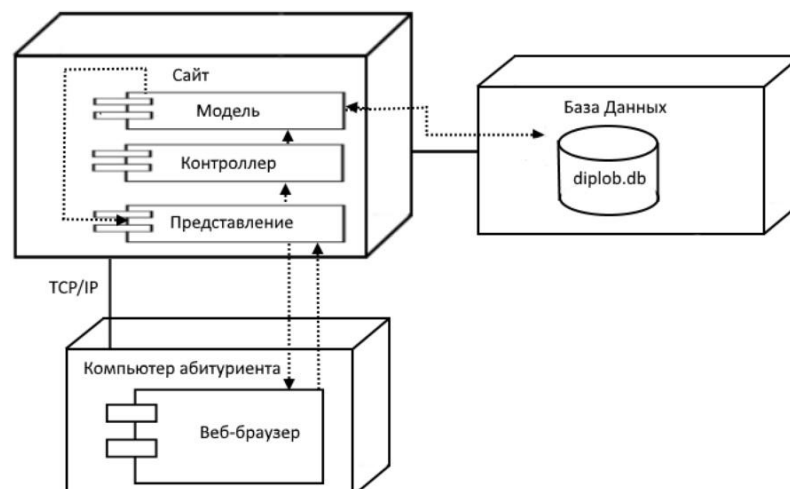


Рисунок 13 – Диаграмма развертывания

2.3.5 Диаграмма состояний

Данная диаграмма описывает все возможные состояния одного экземпляра определенного класса и возможные последовательности его переходов из одного состояния в другое, то есть моделирует все изменения состояний объекта как его реакцию на внешние воздействия. Данная диаграмма, описывающая процесс входа на сайт, представлена на рисунке 14.

Следует сделать оговорку, что на данном сайте самостоятельная регистрация пользователя отсутствует. Создание комбинации логина и пароля для абитуриента создает только администратор сайта.

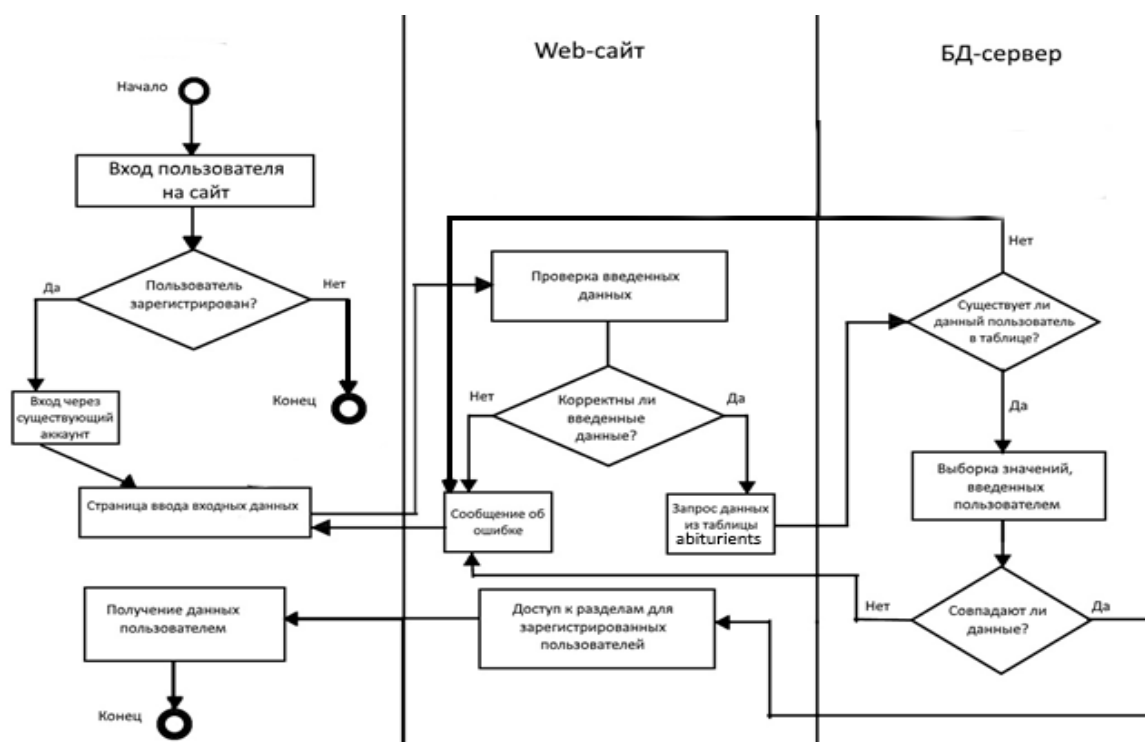


Рисунок 14 – Диаграмма состояний, описывающая процесс входа на сайт

2.3.5 Диаграмма деятельности

UML-диаграмма, на которой показано разложение некоторой деятельности на её составные части. Под деятельностью понимается спецификация исполняемого поведения в виде координированного

последовательного и параллельного выполнения подчинённых элементов - вложенных видов деятельности отдельных действий, соединённых между собой потоками, которые идут от выходов одного узла к входам другого.

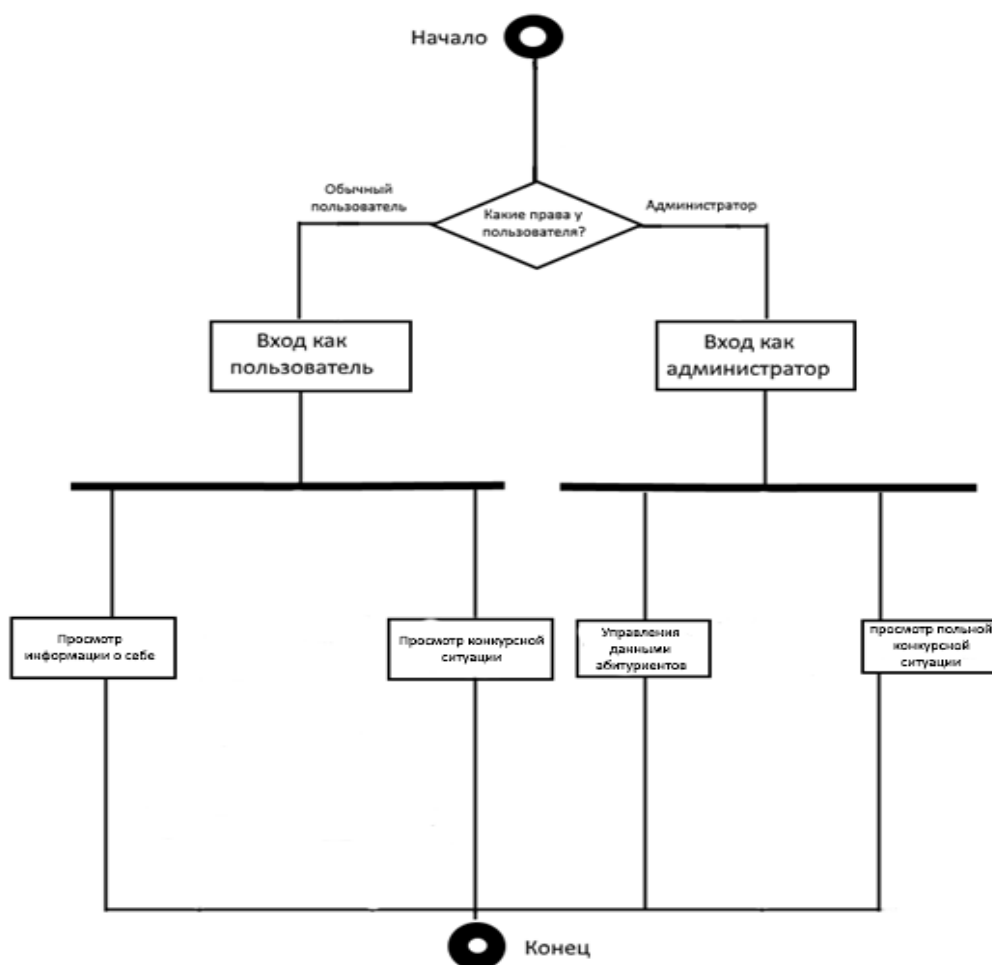


Рисунок 15 – Диаграмма деятельности

2.3.5 Диаграмма прецедентов

На диаграмме прецедентов (она же Use-case диаграмма, она же – диаграмма вариантов использования) показаны отношения администраторов и обычных пользователей, позволяющая описать модель сайта на концептуальном уровне. Такая диаграмма создается для того, чтобы понять, как должна работать система, какие функции должны и будут выполнять ее пользователи.

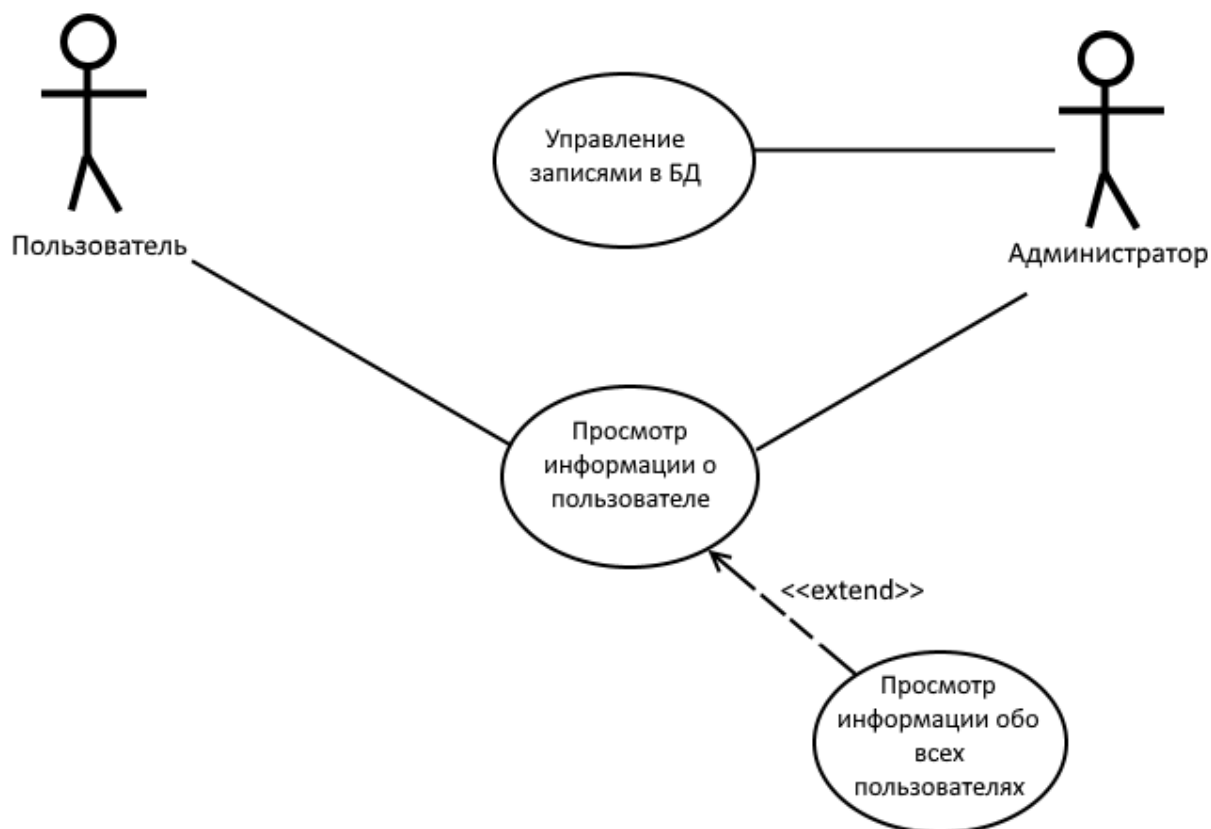


Рисунок 16 – Диаграмма прецедентов

2.4 Работа сайта. Реализация шаблона MVC

PHP изначально не предназначен для этого шаблона, но это не значит, что его нельзя реализовать на этом языке, хотя это и будет гораздо сложнее, чем, например, на фреймворках Django или Ruby on Rails.

Для начала были созданы файлы, из которых и будет состоять шаблон MVC – модель, контроллер и представление. Тут следует уточнить, что, во-первых, для каждой страницы должно быть свое представление, но не для каждого представления будет отдельный контроллер или отдельная модель.

В качестве примера работы рассмотрим процесс входа на сайт.

Работа происходит таким образом: пользователь, переходя по ссылке на сайт, попадает на представление страницы. На странице при этом могут выполняться какие-либо действия. Скриншот страницы входа можно увидеть на рисунке 17.

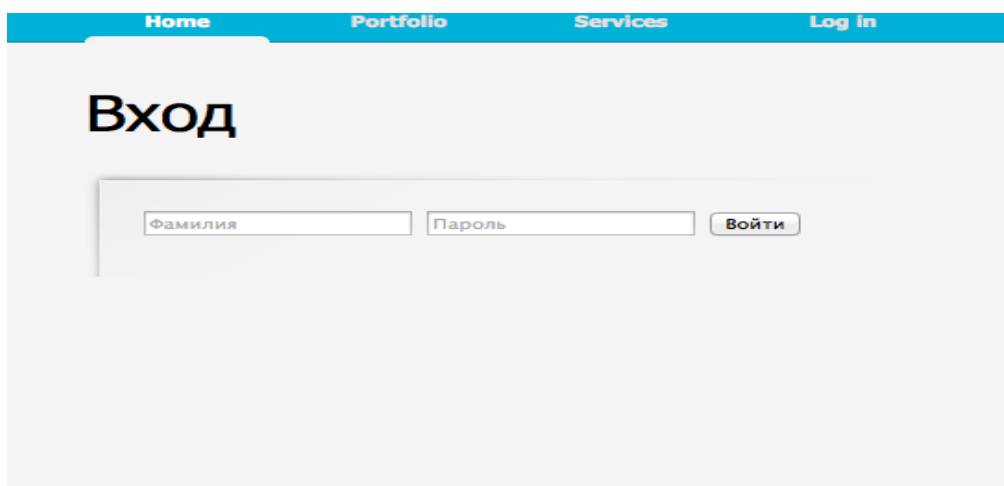


Рисунок 17 – Страница входа

В данном случае, под «действиями» подразумевается внос своих личным данных (логин и пароль, который состоит из серии и номера паспорта) и нажатие кнопки «Войти». Как только необходимые действия завершены, данные отправляются в контроллер, связанный с этим представлением. Контроллер проверяет, что ему отправили, и пересылает эти данные в модель, где они уже обрабатываются в требуемом методе. Набор файлов, из которых состоит сайт, можно увидеть на рисунке 18. В этот набор входят все необходимые контроллеры, представления и модели.

components	--	05 Jun 2016 22:56
config	--	06 Jun 2016 19:34
db_params.php	4 KB	18 May 2016 23:20
routes.php	4 KB	06 Jun 2016 19:34
controllers	--	13 Jun 2016 00:19
AdminController.php	4 KB	06 Jun 2016 17:05
AdminUserController.php	4 KB	06 Jun 2016 19:35
CabinetController.php	4 KB	13 Jun 2016 00:19
NewsController.php	4 KB	02 Jun 2016 00:16
PageNotFoundController.php	4 KB	07 May 2016 12:08
ProductController.php	4 KB	07 May 2016 08:45
UserController.php	4 KB	06 Jun 2016 01:46
index.php	4 KB	06 Jun 2016 01:33
models	--	13 Jun 2016 00:55
News.php	4 KB	08 Jun 2016 23:01
User.php	8 KB	13 Jun 2016 00:55
template	--	02 Jun 2016 01:00
test.jpg	25 KB	07 May 2016 08:58
views	--	08 Jun 2016 13:47

Рисунок 18 – Файлы, из которых состоит сайт

Из этого скриншота также видно, что модели всего две, хотя контроллеров гораздо больше. Это одна из ключевых особенностей MVC, о которой говорилось выше – представление и контроллер зависят от модели, но модель от них зависимости не имеет. Из этого следует, что у одной модели может быть множество контроллеров и представлений.

Также очень важной задачей было создание ЧПУ («человеко-понятный URL»). Гораздо ведь удобней, если вместо странного /user?ud1274214@813 и прочего пользователь увидит лаконичное /user/cabinet/ в адресной строке своего браузера. Звучит на словах все легко, но на деле это невероятно трудоемкий процесс, который, к сожалению, необходимо сделать для успешной реализации MVC.

Для начала, следует ввести общее именование для всех контроллеров и методов в них. В каждом контроллере есть класс, имя которого выглядит как ___Controller, где вместо пропуска стоит имя контроллера. Этот класс имеет методы, которые обязательно имеют имя action___, где вместо пропуска также стоит имя метода. Сделано это для того, чтобы при переходе на ссылку выбирался определенный контроллер, а в этом контроллере – определенный метод. Все доступные ссылки приведены в файле routes; также в этом файле приведены все контроллеры и методы этих контроллеров, использующиеся при переходе по ссылке. Все это наглядно можно посмотреть на рисунках 19 и 20.

```
<?php
class UserController
{
    public function actionLogin()
    {
        $last_name = '';
        $password = '';
        if (isset($_POST['submit']))
        {
            $last_name = $_POST['last_name'];
            $password = $_POST['password'];

            $errors = false;

            if (!User::checkPassword($password))
            {
                $errors[] = 'Слишком короткий пароль!';
            }

            $userId = User::checkUserData($last_name, $password);
            if (!$userId)
            {
                $errors[] = 'неправильные данные для входа на сайт';
            }
        }
    }
}
```

Рисунок 19 – Контроллер и методы в нем


```

<?php
return array(
    'news/([0-9]+)' => 'news/view/$1',
    'news.' => 'news/index',
    'product' => 'product/list',
    '404' => 'PageNotFound/test',
    'cabinet' => 'cabinet/index',
    'user/login' => 'user/login',
    'user/logout' => 'user/logout',
    'admin/user/delete/([0-9]+)' => 'adminUser/delete/$1',
    'admin/user.' => 'adminUser/index',
    'admin' => 'admin/index',
    '' => 'user/login',
);

```

Рисунок 20 – Файл routes.php

Routes.php – это массив, который хранит как ключ (то, что до знака ==>) ссылку, по которой будет переходить пользователь, а как значение – имя контроллера (до знака слеша) и имя метода в этом контроллере, который будет обрабатывать запрос (то, что после слеша).

Однако самое сложное и главное – заставить сайт обрабатывать эти ссылки и открывать нужный файл, дабы пользователь увидел то, что он хочет. Выполняется это все в методе run файла router.php, скриншот которого можно увидеть на рисунке 21.

```

public function run()
{
    $uri = $this->getURI();
    foreach ($this->routes as $uriPattern => $path)
    {
        if (preg_match("~$uriPattern~", $uri))
        {
            $internalRoute = preg_replace("~$uriPattern~", $path, $uri);

            $segment = explode('/', $internalRoute);
            $controllerName = array_shift($segment).'Controller';
            $controllerName = ucfirst($controllerName);
            $actionName = 'action'.ucfirst(array_shift($segment));

            $parameters = $segment;
            $controllerFile = ROOT.'/controllers/'.$controllerName.'.php';

            if (file_exists($controllerFile))
            {
                include_once($controllerFile);
            }
            $controllerObject = new $controllerName;
            $result=call_user_func_array(array($controllerObject, $actionName), $parameters);
            if ($result!=null)
            {
                break;
            }
        }
    }
}

```

Рисунок 21 – Метод run в файле router.php

С помощью функций, работающих со строками, из нужного запроса выделяется имя контроллера, имя метода, после чего выбирается файл, который будет иметь, во-первых, имя, которое мы выбрали, и использоваться метод, который требуется для обработки функции.

После этого можно начинать работать с сайтом. Например, так выглядит страница пользователя, на которой он видит свое место на интересующей его специальности (которую он выбрал). У пользователя Иван Иванов оставлено согласие на специальность 090304, в то время как у пользователя Милли Пухлян его нет, поэтому она видит другой текст. Скриншоты этих страниц можно увидеть на рисунках 22 и 23.

Кабинет пользователя	
Здравствуйтесь, Иван Иванов	
Аттестат	присутствует
Математика	100
Русский	100
Информатика	100
Специальность	Место в рейтинге с договором
09.03.04	1 из 24

Рисунок 22 – Кабинет пользователя Иван Иванов, оставившего согласие

Здравствуйтесь, Алексей Васечкин	
Аттестат	отсутствует
Математика	45
Русский	99
Информатика	98
Специальность	Место в рейтинге с договором
09.03.04	3 из 24, если оставите согласие

Рисунок 23 – Кабинет пользователя Алексея Васечкина, не оставившего согласие

Также на сайте присутствует панель администратора, с помощью которой есть возможность добавить нового абитуриента, отредактировать данные

пользователя, который уже присутствует в базе данных, или же удалить его, а также вывести конкурсную ситуацию по каждому из направлений. Все это можно подробно увидеть на рисунках 24 и 25.

Добавить пользователя

Фамилия	Имя	отчество	Математика	Русский	Информатика	Аттестат	Редактировать данные	Удалить пользователя
Иванов	Иван	Иваныч	100	100	100	Оригинал	Редактировать	Удалить
Петрова	Анастасия	Андреевна	74	87	89	Оригинал	Редактировать	Удалить
Васечкин	Алексей	Павлович	45	99	98	Копия	Редактировать	Удалить

Рисунок 24 – Фрагмент таблицы с абитуриентами

Выйти

Управление пользователями

Вывод по направлениям: [090304](#), [090301](#), [090302](#)

Рисунок 25 – Фрагмент панели администратора

ЗАКЛЮЧЕНИЕ

В ходе данной бакалаврской работы был проведен анализ текущих тенденций в приемных кампаниях различных университетов, выделены их плюсы и минусы. Ввиду ввода новой системы поступления, произведена оценка как старой системы с приоритетами, так и новой системы с согласиями, после чего были разобраны как плюсы, так и минусы обеих. Также была произведена проверка систем мониторинга других вузов, в ходе которой были выделены проблемы, которых нужно было избежать в данной бакалаврской работе.

Произведен анализ актуальных шаблонов архитектуры программного обеспечения, был выбран наиболее подходящий к требованиям, которые поставлены в начале работы. Также было оценено несколько языков программирования, после чего на выбранном языке началась работа над созданием сайта.

В результате был создан сайт, удовлетворяющий всем предъявленным в начале требованиям, актуальности, основная функция которого – мониторинг абитуриентов – выполнена в полной мере. Также присутствует возможность вывода списка абитуриентов, которые проходят в конкурсе на поступление. В дальнейшем планируется постоянное развитие и поддержка сайта, налаживание обратной связи с его администрацией.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Информационный портал коллективного блога (специализированная пресса) [Электронный ресурс] Статья о шаблоне MVC. Режим доступа: <https://habrahabr.ru/post/181772/>
2. Информационный портал коллективного блога (специализированная пресса) [Электронный ресурс] Статья о UML. Режим доступа: <https://habrahabr.ru/post/150041/>
3. Информационный портал коллективного блога (специализированная пресса) [Электронный ресурс] Статья о сравнении разных шаблонов архитектуры приложений. Режим доступа: <https://habrahabr.ru/post/215605/>
4. Персональный блог о разработке ПО [Электронный ресурс] Статья о паттернах. Режим доступа: <https://www.outcoldman.com/ru/archive/паттерны-mvc-mvp-и-mvvm/>
5. Персональный блог о разработке ПО [Электронный ресурс] Статья о сравнении паттернов. Режим доступа: <https://nirajrules.wordpress.com/2009/07/18/mvc-vs-mvp-vs-mvvm/>
6. Русскоязычный сайт, посвященный программированию [Электронный ресурс] Model-View-Controller в .Net. Режим доступа: <http://rstdn.ru/article/patterns/ModelViewPresenter.xml>
7. Сайт Сибирского Федерального Университета [Электронный ресурс] Правила приёма в СФУ на обучение по программам бакалавриата и специалитета на 2016/17 учебный год. Режим доступа: <http://about.sfu-kras.ru/node/9127>
8. Сайтостроение от А до Я [Электронный ресурс] PHP, Ruby, Python – краткая характеристика трёх языков программирования. Режим доступа: http://www.internet-technologies.ru/articles/article_1991.html
9. Свободная общедоступная универсальная интернет-энциклопедия [Электронный ресурс] Статья о шаблоне MVC. Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-Controller>

10. Свободная общедоступная универсальная интернет-энциклопедия [Электронный ресурс] Статья о шаблоне MVVM. Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-ViewModel>

11. Свободная общедоступная универсальная интернет-энциклопедия [Электронный ресурс] Статья о UML. Режим доступа: <https://ru.wikipedia.org/wiki/UML>

12. Свободная общедоступная универсальная интернет-энциклопедия [Электронный ресурс] Статья о MVP. Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-Presenter>

13. СТО 4.2–07–2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности. – Введ. 30.12.2013. – Красноярск: СФУ, 2014. – 60 с.