ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК
КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ПРОГНОЗИРОВАНИЯ РАСХОДОВ ЭЛЕКТРОЭНЕРГИИ С УЧЕТОМ ТРЕБОВАНИЙ ОАО «БЕЛГОРОДЭНЕРГОСБЫТ»

Выпускная квалификационная работа
обучающейся по направлению подготовки
02.03.01 Математика и компьютерные науки
очной формы обучения, группы 07001303
Ушаковой Светланы Николаевны

Научный руководитель
к.т.н., доцент
В.В. Муромцев

БЕЛГОРОД 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3
1 ОБЗОР И АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5
1.1 Проблема прогнозирования 5
1.2 Статистические методы прогнозирования временных рядов 9
1.3 Требования к автоматизированной системе 16
2 ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ 18
2.1 Проектирование архитектуры автоматизированной системы 18
2.2 Проектирование структуры базы данных 19
2.3 Выбор средств разработки 21
3 РАЗРАБОТКА И ТЕСТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ 27
3.1 Реализация базовых компонентов системы и базы данных 27
3.2 Реализация пользовательского интерфейса 37
3.3 Тестирование программного комплекса 44
ЗАКЛЮЧЕНИЕ 52
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 53
ПРИЛОЖЕНИЕ 55

ВВЕДЕНИЕ

В настоящее время реформирование электроэнергетики России привело к конкуренции на оптовом и розничном рынках электроэнергии. В связи с процессом демонополизации в сфере сбыта электрической энергии и возникновением независимых сбытовых компаний энергосбытовой бизнес оказывается в совершенно новых для себя условиях. Постепенное усиление конкуренции на розничном рынке электроэнергии предъявляет новые требования ко всем участникам рынка. Для энергосбытовых компаний увеличивается риск потери своих позиций в конкурентной борьбе. Поэтому сегодня перед энергосбытовыми организациями встает серьезная задача удержания крупных потребителей от ухода на оптовый рынок электроэнергии и защита клиентов от «перехвата» со стороны независимых компаний. Это обуславливает использование новых механизмов в управлении энергосбытовой организации, которые позволят ей сохранить и усилить конкурентное положение компании.

Одним из возможных путей достижения этой цели является планирование и контроль процесса распределения электроэнергии и прогнозирования объемов потребляемой электроэнергии.

Энергосбытовое предприятие является посредником между электростанциями, занимающимися выработкой электроэнергии, и потребителями. Выработанная электроэнергия закупается энергосбытовым предприятием на оптовом рынке и реализуется конечному пользователю: промышленным предприятиям и бытовому сектору. Постоянные изменения на предприятиях и в бытовом секторе оказывают влияние на общее потребление региона в целом, что является проблемой энергосбытовых предприятий, задачей которых является полное удовлетворение потребностей региона в электроэнергии. Энергосбытовое предприятие как посредник между бытовым потребителем и электростанцией должно

проводить постоянное исследование изменения потребностей региона в объемах электроэнергии. Задача состоит в как можно более точном прогнозировании электропотребления для закупа необходимых объемов электроэнергии на оптовом рынке.

Целью данной работы является разработка автоматизированной системы прогнозирования расходов электроэнергии. Для достижения данной цели необходимо решить следующие задачи:

– произвести обзор и анализ современных математических методов построения прогнозов и выбрать наиболее подходящий метод для решения поставленной проблемы;

– сформулировать требования к автоматизированной системе прогнозирования расходов электроэнергии;

– произвести проектирование автоматизированной системы прогнозирования расходов электроэнергии;

– реализовать автоматизированную систему прогнозирования расходов электроэнергии в виде законченного программного продукты;

– произвести тестирование разработанного программного продукта и проверить точность и адекватность составляемых прогнозов.

В первой главе проведен обзор и анализ современных методов прогнозирования, выявлены требования к информационной системе прогнозирования расходов электроэнергии.

Во второй главе описан процесс проектирования структуры информационной системы прогнозирования расходов электроэнергии, архитектуры базы данных и интерфейса пользователей.

В третьей главе описан процесс разработки основных компонентов системы и интерфейса пользователей, описан процесс тестирования системы.

Выпускная квалификационная работа содержит 54 страницы без приложения, 33 рисунка, 6 формул, 1 таблицу, 12 листингов и 1 приложение. В процессе создания было использовано 16 литературных источников.

1 ОБЗОР И АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Проблема прогнозирования

Под прогнозом понимается научно обоснованное описание возможных состояний объектов в будущем, а также альтернативных путей и сроков достижения этого состояния. Необходимость прогноза обусловлена желанием знать события будущего, что невозможно на 100 % в принципе, исходя из статистических, вероятностных, эмпирических, философских принципов. По сроку прогнозы можно разделить на краткосрочные, среднесрочные, долгосрочные, дальнесрочные. По масштабу на частные, местные, региональные, отраслевые, мировые (глобальные) [1].

Различают метод прогнозирования и модель прогнозирования. *Метод прогнозирования* представляет собой последовательность действий, которые нужно совершить для получения модели прогнозирования. *Модель прогнозирования* есть функциональное представление, адекватно описывающее рассматриваемый процесс и являющееся основой для получения его значений в будущем.

Существующие методы прогнозирования принято разделять на интуитивные методы и формализованные методы [1]. Они представлены на рис. 1.1.

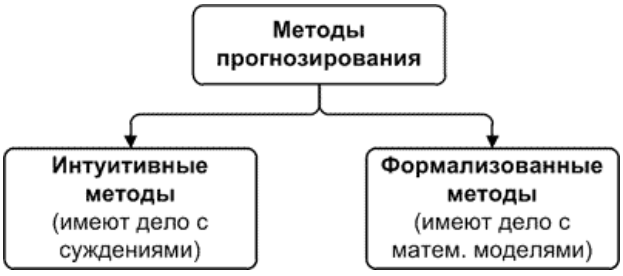

Рис. 1.1. Классификация методов прогнозирования

Интуитивные методы прогнозирования имеют дело с суждениями и оценками экспертов [2]. На сегодняшний день они часто применяются в маркетинге, экономике, политике, так как система, поведение которой необходимо спрогнозировать, или очень сложна и не поддается математическому описанию, или очень проста и в таком описании не нуждается.

Формализованные методы – описанные в литературе методы прогнозирования, в результате которых строят модели прогнозирования, то есть определяют такую математическую зависимость, которая позволяет вычислить будущее значение процесса, то есть сделать прогноз [2].

Рассмотрим классификацию моделей прогнозирования. На первом этапе модели следует разделить на две группы: модели предметной области и модели временных рядов. Наглядно это представлено на рис. 1.2.

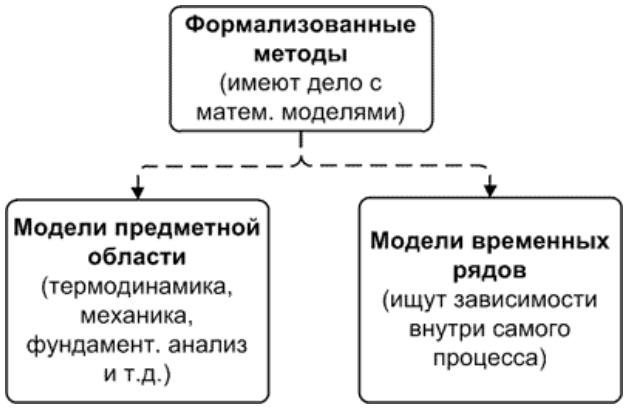

Рис. 1.2. Классификация моделей прогнозирования

Модели предметной области – такие математические модели прогнозирования, для построения которых используют законы предметной области. Например, модель, на которой делают прогноз погоды, содержит уравнения динамики жидкостей и термодинамики [2]. Прогноз развития популяции делается на модели, построенной на дифференциальном уравнении. Прогноз уровня сахара крови человека, больного диабетом, делается на основании системы дифференциальных уравнений. Словом, в таких моделях используются зависимости, свойственные конкретной

предметной области. Такого рода моделям свойственен индивидуальный подход в разработке.

Модели временных рядов – математические модели прогнозирования, которые стремятся найти зависимость будущего значения от прошлого внутри самого процесса и на этой зависимости вычислить прогноз. Эти модели универсальны для различных предметных областей, то есть их общий вид не меняется в зависимости от природы временного ряда [2]. Мы можем использовать нейронные сети для прогнозирования температуры воздуха, а после аналогичную модель на нейронных сетях применить для прогноза биржевых индексов.

В настоящее время существует огромное количество различных моделей прогнозирования временных рядов, однако почти все их можно разделить на две большие группы: статистические модели и структурные модели. Модели временных рядов представлены на рис. 1.3.

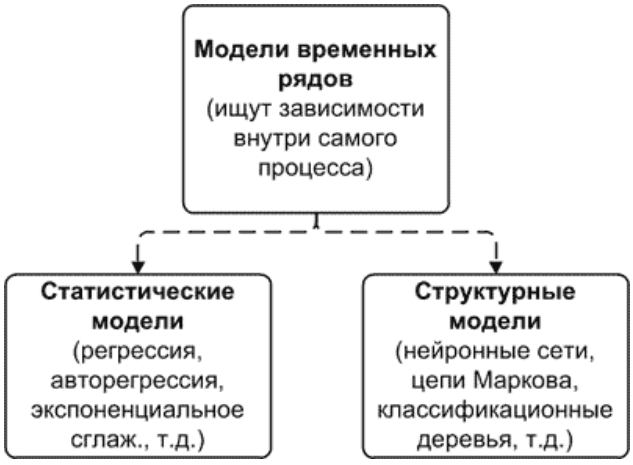

Рис. 1.3. Классификация моделей временных рядов

В статистических моделях зависимость будущего значения от прошлого задается в виде некоторого уравнения [3]. К ним относятся:

– регрессионные модели (линейная регрессия, нелинейная регрессия);

– авторегрессионные модели (ARIMAX, GARCH, ARDLM);

– модель экспоненциального сглаживания;

– модель по выборке максимального подобия.

В структурных моделях зависимость будущего значения от прошлого задается в виде некоторой структуры и правил перехода по ней [3]. К ним относятся:

– нейросетевые модели;

– модели на базе цепей Маркова;

– модели на базе классификационно-регрессионных деревьев.

Общая классификация моделей и методов прогнозирования представлена на рис. 1.4.

Рис. 1.4. Общая классификация методов и моделей прогнозирования

В литературе название метода и соответствующей ему модели прогнозирования, как правило, совпадают.

В рамках данной работы ставится задача построения прогнозов расходов электроэнергии на основе накопленных статистических данных за предыдущие периоды, поэтому для решения данной задачи целесообразно использовать одну из моделей прогнозирования временных рядов, в частности одну из статистических моделей.

1.2 Статистические методы прогнозирования временных рядов

Процессы, перспективы которых необходимо предсказывать описываются *временными рядами*, то есть последовательностью значений некоторых величин, полученных в определенные моменты времени. Временной ряд – это множество наблюдений, генерируемых последовательно во времени. Если время непрерывно, временно ряд называется непрерывным. Если время изменяется дискретно, временной ряд дискретен.

Временные ряды принято разделять на стационарные и нестационарные временные ряды. *Стационарным временным рядом* называется такой ряд, который остается в равновесии относительно постоянного среднего уровня. Остальные временные ряды являются нестационарными [4].

Горизонт времени, на который необходимо определить значения временного ряда, называется временем упреждения. В зависимости от времени упреждения задачи прогнозирования, как правило, делятся на долгосрочное, среднесрочное и краткосрочное прогнозирование.

Метод прогнозирования состоит из определенной последовательности шагов, в результате исполнения которых определяется модель прогнозирования определенного временного ряда. Кроме этого, метод прогнозирования включает в себя шаги по оценке качества прогнозных значений. В итеративный алгоритм к определению конкретной модели прогнозирования включает в себя пять этапов [4].

Первый этап: основываясь на собственном или стороннем опыте, определяется общий класс моделей прогнозирования временного ряда на необходимый период (на заданный горизонт).

Второй этап: выбранный в ходе первого этапа общий класса моделей, как правило, слишком обширен. Поэтому для более точного приближения к исходному временному ряду выбирается подкласс модели прогнозирования.

Третий этап: после того, как на втором этапе выбран определенный подкласс модели, нужно определить ее параметры (для моделей, которые содержат в себе параметры), или структуру (для моделей, которые относятся к классу структурных моделей. В ходе этого шага, как правило, пользуются одним из итеративных способов: производится оценка участка (или всего) временного ряда для различных значений изменяемых величин.

Четвертый этап: производится диагностическая проверка выбранной модели прогнозирования. Для этого берется один или несколько отрезков временного ряда, достаточных по длине для проведения проверочного прогнозирования и оценки качества прогноза. Участки временного ряда, выбранные для диагностики модели прогнозирования, называются контрольными участками или периодами.

Пятый этап: если для рассматриваемой задачи точность диагностического прогнозирования оказалась достаточной, то можно считать, что выбранная модель готова к использованию. Если же для последующего использования прогнозных значений точность прогнозирования оказалась неприемлемой, то необходимой повторить описанные выше шаги необходимое количество раз.

Моделью прогнозирования временного ряда называется функциональное представление, адекватно описывающее временной ряд. В контексте рассмотрения проблемы предсказания временных радов существует два варианта постановки задачи. В первом варианте при расчёте предсказываемых значений исследуемого временного ряда используются только значения этого ряда. Во втором варианте при расчете предсказываемых значений используются не только известные значения этого ряда, но и значения внешних параметров, которые также представляются в виде временных рядов. Зачастую, количество значений во временном ряде параметров может отличаться от количества значений в исследуемом временном ряде. В общем случае внешние факторы могут быть дискретными, т. е. представленными временными рядами, например,

температура воздуха. Также внешние факторы могут быть категориальными, т. е. состоящими из подмножеств, например, в зависимости от веса тела человека можно отнести к трем категориям: «легкий», «средний», «тяжелый». Лишь некоторые модели прогнозирования позволяют учитывать категориальные внешние факторы, большинство моделей позволяют учитывать только дискретные.

Для прогнозирования значений временного ряда необходимо определить функциональную зависимость, которая корректно описывает исследуемый временной ряд. Такая зависимость называется моделью прогнозирования. Конечной целью подбора модели прогнозирования является создание такой модели, при которой среднее абсолютное отклонение истинного значения от прогнозируемого значения стремится к минимальному для выбранного интервала (горизонта). Данный интервал называют временем упреждения. Когда модель прогнозирования исследуемого временного ряда разработана, следует рассчитать предсказываемые значения временного ряда, а также их доверительный интервал [15].

В статистических моделях, как правило, функциональная зависимость между предсказываемыми и фактическими значениями временного ряда, а также внешними параметрами задана аналитически. К классу статистических моделей можно отнести следующие подклассы: регрессионные модели, авторегрессионные модели и модели экспоненциального сглаживания.

Регрессионные модели.

Существует много задач, требующих изучения отношения между двумя и более переменными. Для решения таких задач используется регрессионный анализ. Целью регрессионного анализа является определение зависимости между исходной переменной и множеством внешних факторов (регрессоров). При этом коэффициенты регрессии могут определяться по методу наименьших квадратов или методу максимального правдоподобия.

Линейная регрессионная модель. Самым простым вариантом регрессионной модели является линейная регрессия. В основу модели положено предположение, что существует дискретный внешний фактор X(t), оказывающий влияние на исследуемый процесс Z(t), при этом связь между процессом и внешним фактором линейна [5].

Модель прогнозирования на основании линейной регрессии описывается уравнением (1.1).

$$Z(t) = \alpha_0 + \alpha_1 X(t) + \varepsilon_t \quad (1.1)$$

где α_0 и α_1 — коэффициенты регрессии; ε_t — ошибка модели. Для получения прогнозных значений *Z(t)* в момент времени *t* необходимо иметь значение *X(t)* в тот же момент времени *t*, что редко выполнимо на практике.

Множественная регрессионная модель. На практике на процесс *Z(t)* оказывают влияние целый ряд дискретных внешних факторов $X_1(t),\ldots,X_S(t)$. Тогда модель прогнозирования имеет вид (1.2):

$$Z(t) = \alpha_0 + \alpha_1 X_1(t) + \alpha_2 X_2(t) + \ldots + \alpha_s X_S(t) + \varepsilon_t \quad (1.2)$$

Недостатком данной модели является то, что для вычисления будущего значения процесса *Z(t)* необходимо знать будущие значения всех факторов $X_1(t),\ldots,X_S(t)$, что почти невыполнимо на практике.

Нелинейной регрессионной модели. В их основу положено предположение о том, что существует известная функция, описывающая зависимость между исходным процессом *Z(t)* и внешним фактором *X(t).*

$$Z(t) = F(X(t), A) \quad (1.3)$$

В рамках построения модели прогнозирования необходимо определить параметры функции *A*. Однако на практике редко встречаются процессы, для которых вид функциональной зависимости между процессом *Z(t)* и внешним фактором *X(t)* заранее известен. В связи с этим нелинейные регрессионные модели применяются редко.

Авторегрессионные модели

В основу авторегрессионных моделей заложено предположение о том, что значение процесса *Z(t)* линейно зависит от некоторого количества предыдущих значений того же процесса *Z(t-1),…,Z(t-p)*. В области анализа временных рядов *модель авторегрессии* (autoregressive, AR) и *модель скользящего среднего* (moving average, MA) является одной из наиболее используемых [5]. В этой модели текущее значение процесса выражается как конечная линейная совокупность предыдущих значений процесса и импульса, который называется «белым шумом» и представлена формулой (1.4).

$$Z(t) = C + \varphi_1 Z(t-1) + \varphi_2 Z(t-2) + .. + \varphi_p Z(t-p) + \varepsilon_t \quad (1.4)$$

Формула (1.4) описывает *процесс авторегрессии порядка p*, который в литературе часто обозначается *AR(p)*, здесь *C* — вещественная константа, $\varphi_1,..,\varphi_p$ — коэффициенты, ε_t — ошибка модели. Для определения φ_i и *C* используют метод наименьших квадратов или метод максимального правдоподобия.

Другой тип модели имеет большое значение в описании временных рядов и часто используется совместно с авторегрессией. Он называется *моделью скользящего среднего порядка q* и описывается уравнением (1.5).

$$Z(t) = \frac{1}{q}\big(Z(t-1) + Z(t-2) + .. + Z(t-q)\big) + \varepsilon_t \quad (1.5)$$

В литературе процесс (1.5) часто обозначается MA(q); здесь q — порядок скользящего среднего, εt — ошибка прогнозирования. Модель скользящего среднего является по сути дела фильтром низких частот. Нужно отметить, что существуют простые, взвешенные, кумулятивные, экспоненциальные модели скользящего среднего.

Для достижения большей гибкости в подгонке модели часто целесообразно объединить в одной модели авторегрессию и скользящее среднее. Общая модель обозначается ARMA(p,q) соединяет в себе фильтр в виде скользящего среднего порядка q и авторегрессию фильтрованных значений процесса порядка p [6].

Модели экспоненциального сглаживания

Модель экспоненциального сглаживания (exponential smoothing, ES) применяется для моделирования финансовых и экономических процессов. В основу экспоненциального сглаживания заложена идея постоянного пересмотра прогнозных значений по мере поступления фактических. Модель ES присваивает экспоненциально убывающие веса наблюдениям по мере их старения [6]. Таким образом, последние доступные наблюдения имеют большее влияние на прогнозное значение, чем старшие наблюдения. Функция модели ES имеет вид (1.6).

$$Z(t) = S(t) + \varepsilon_t,$$
$$S(t) = \alpha \cdot Z(t-1) + (1-\alpha) \cdot S(t-1) \qquad (1.6)$$

где α — коэффициент сглаживания, $0 < \alpha < 1$; начальные условия определяются как $S(1) = Z(0)$. В данной модели каждое последующее сглаженное значение $S(t)$ является взвешенным средним между предыдущим значением временного ряда $Z(t)$ и предыдущего сглаженного значения $S(t-1)$.

Достоинства и недостатки статистических методов прогнозирования

К достоинствам регрессионных моделей прогнозирования относят простоту, гибкость и единообразие алгоритма их анализа и проектирования. При применении линейных регрессионных моделей результат прогнозирования, как правило, получается быстрее, чем в случае применения других моделей. Так же к достоинствам моделей данного класса относят прозрачность моделирования. В этих моделях для анализа доступны все промежуточные вычисления. К числу недостатков линейных регрессионных моделей относится слабая адаптивность и практически полное отсутствие возможности моделирования нелинейных процессов. Основным недостатком нелинейных регрессионных моделей является сложность определения вида функциональной зависимости, а также трудоемкость определение параметров модели.

Важными достоинствами авторегрессионных моделей и методов являются их простота и прозрачность моделирования. Еще одним достоинством является единообразие анализа и проектирования. В настоящее время данный класс моделей является одним из самых распространенных. В связи с этим легко найти много примеров решения задач прогнозирования на основе авторегрессионных моделей в открытом доступе для различных предметных областей.

К недостаткам моделей данного класса можно отнести большое количество параметров, процесс подбора которых порой является очень сложной задачей. Данные модели также характеризуются низкой адаптивностью и линейностью и, как следствие, в них отсутствует возможность моделирования нелинейных процессов, которая очень часто встречается на практике.

Достоинствами моделей и методов экспоненциального сглаживания являются простота и единообразие их анализа и проектирования. Данный класс моделей чаще других используется для долгосрочного прогнозирования. Недостатком данного класса моделей прогнозирования является отсутствие гибкости [16].

На основании обзора основных статистических методов прогнозирования можно сделать вывод, что для решения задачи прогнозирования расходов электроэнергии наиболее подходящим подклассом методов является модель экспоненциального сглаживания.

1.3 Требования к автоматизированной системе

В данной выпускной квалификационной работе рассматривается автоматизация процесса прогнозирования расходов электроэнергии в рамках предприятия ОАО «Белгородэнергосбыт». Основными направлениями деятельности компании являются покупка электрической энергии на оптовом рынке электроэнергии (мощности) и последующая ее реализация потребителям области. В сферу деятельности работников компании входит анализ тарифов на электроэнергию, прогноз полезного отпуска электроэнергии на предстоящий год, а также изучение рынка сбыта и реализация мероприятий по его расширению.

В настоящее время в компании существует автоматизированная информационная система учета текущего потребления электроэнергии, данная система поддерживает экспорт данных в формате .xls. По этим данным строится прогноз потребления электроэнергии на будущие периоды (квартал, полгода, год). Прогноз строится экспертным методом. Необходимо разработать информационную систему, которая позволит автоматизировать это процесс.

Требования к функционалу системы

Информационная система прогнозирования расходов электроэнергии должна предоставлять пользователям перечисленные ниже функциональные возможности:

– производить импорт исторических данных потребления электроэнергии из существующей системы учета;

– накапливать информацию о потреблении электроэнергии для ее дальнейшего анализа;

– предоставлять удобный интерфейс для просмотра исторических данных о потреблении электроэнергии;

– производить автоматическое построение прогноза по потреблению электроэнергии на выбранный период (квартал, полгода, год);

– формировать отчеты об истории потребления и прогнозируемом потреблении в удобном формате (графики).

Дополнительные требования к системе

Разрабатываемая система должна представлять собой веб-приложение. Доступ к хранимой информации должен предоставляться посредством сети Интернет. Система должна поддерживать все современные веб-браузеры. В ходе разработки информационной системы должно быть использовано открытые и свободно распространяемые инструменты и технологии.

2 ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

2.1 Проектирование архитектуры автоматизированной системы

В соответствии с требованиями, предъявляемыми к разрабатываемому программному изделию, оно должно иметь клиент-серверную архитектуру, что, прежде всего, означает разделение комплекса программных средств, необходимых для ее функционирования на две группы: клиентскую и серверную, и влечет за собой необходимость использования большого числа стороннего программного обеспечения [7].

На рис. 2.1 приведена схема взаимодействия основных компонентов программного обеспечения, необходимого для функционирования разрабатываемой системы.

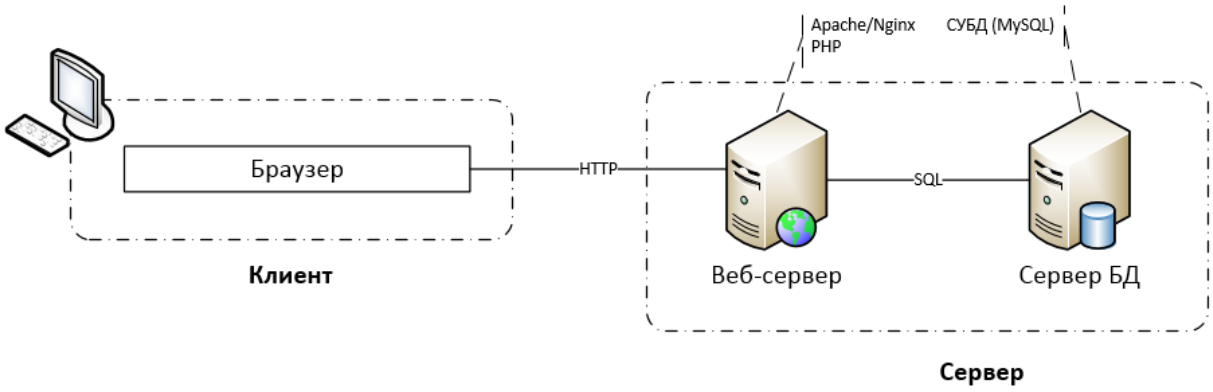

Рис. 2.1. Архитектура программного обеспечения системы

На клиентской стороне необходимо наличие одного из современных веб-браузеров, позволяющих произвести подключение к веб-интерфейсу.

Со стороны сервера работу системы обеспечивает слаженное взаимодействие целого набора программного обеспечения.

В основе всей системы лежит *веб-сервер*. Веб-сервер как правило представляет собой IBM-совместимый персональный компьютер под управлением операционной системы Windows или Linux, и установленным

специализированным программным обеспечением веб-сервера (Apache или Nginx). Веб-сервер хранит компоненты бизнес-уровня, такие как скрипты PHP, статические файлы, изображения, документы и обычные HTML-страницы. Клиент и сервер взаимодействуют посредством протокола HTTP.

Информация, которая обрабатывается системой, хранится в базе данных, находящейся под управлением *системы управления базой данных (СУБД)*. На практике СУБД может быть установлена на том же самом сервере, что и остальное программное обеспечение. Однако, зачастую в целях повышения быстродействия системы его устанавливают на отдельной выделенной системе. СУБД отвечает за обработку запросов от веб-сервера посредством одного из стандартных драйверов. Веб-сервер и сервер базы данных взаимодействуют посредством запросов на языке SQL.

2.2 Проектирование структуры базы данных

Информация, обрабатываемая создаваемой информационной системой должна храниться в базе данных в соответствии с требованиями, сформулированными ранее [8]. В результате анализа предметной области, были выявлены основные сущности и их свойства.

Сущность *«Пользователь»* – отражает информацию о пользователе системы. Её свойства:

- код пользователя;
- фамилия;
- имя;
- код должности;
- имя входа;
- пароль.

Сущность *«Должность»* – отражает информацию о должностях. Её свойства:

– код должности;

– наименование должности.

Сущность «*Подразделение*» – отражает информацию о подразделении, данными которого имеет доступ пользователь, используется для разграничения прав доступа. Её свойства:

– код подразделения;

– наименование подразделения.

Сущность «*Данные*» – отражает информацию данных потребления электроэнергии, поле «ист_значение» имеет значение Истина, если это исторические данные и значение Ложь, если это прогнозируемые данные, по мере поступления исторических данные, прогнозируемые данные заменяются. Её свойства:

– код записи;

– месяц;

– год;

– значение;

– подразделение;

– истинное значение.

На основании анализа описанных выше сущностей предметной области и связей между ними разработана логическая модель базы данных, которая представленная на рис. 2.2. Так же было добавлено несколько дополнительных сущностей в целях нормализации модели.

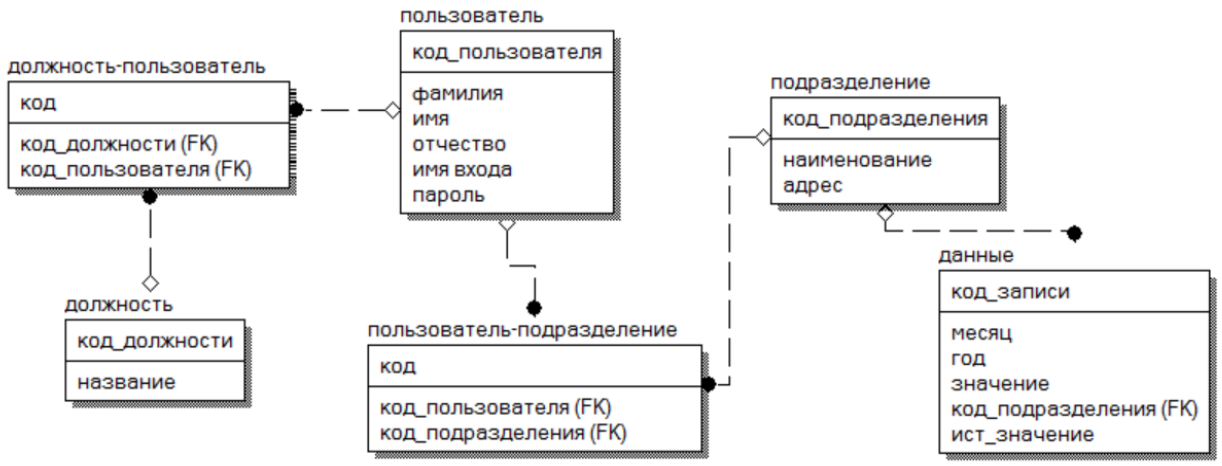

Рис. 2.2. Логическая модель базы данных

В качестве системы управления базой данных была выбрана MySQL, т.к. данная система является свободно распространяемой и простой в использовании. С учетом возможностей выбранной СУБД была спроектирована физическая модель базы данных, приведенная на рис. 2.3.

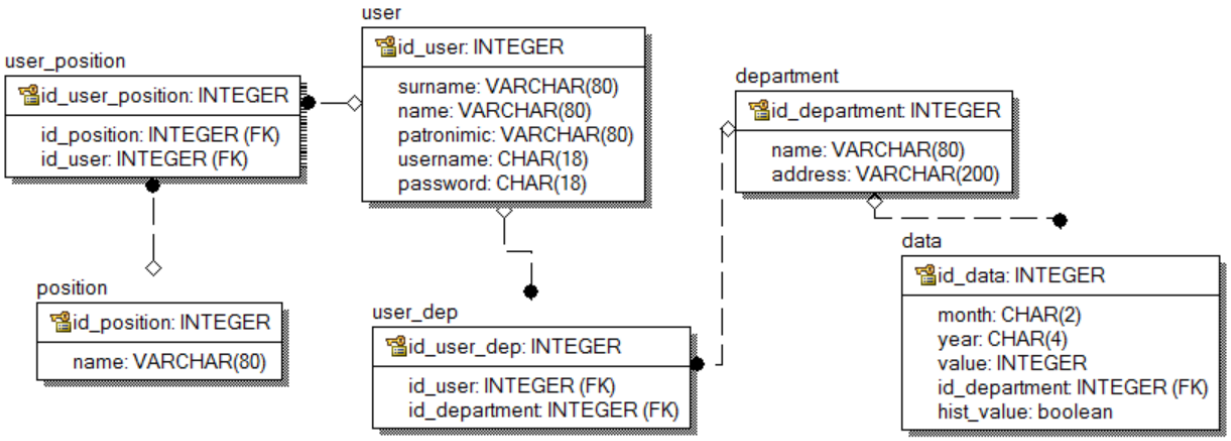

Рис. 2.3. Физическая модель базы данных

2.3 Выбор средств разработки

Интерпретируемый серверный язык PHP

PHP (рекурсивный акроним словосочетания PHP: Hypertext Preprocessor) – это распространенный язык программирования общего

назначения с открытым исходным кодом. PHP сконструирован специально для ведения Web-разработок и его код может внедряться непосредственно в HTML [9]. Сегодня данный язык поддерживается практически всеми хостинг-провайдерерами и является одним из лидеров среди языков программирования, используемых при создании современных веб-сайтов.

Язык и его интерпретатор разрабатываются группой энтузиастов в рамках проекта с открытым кодом. Проект распространяется под собственной лицензией, несовместимой с GNU GPL.

Одной из сильнейших сторон PHP является возможность расширения ядра путем разработки подключаемых модулей, «расширений» для работы с базами данных, динамической графикой, криптографическими библиотеками, документами формата PDF и др.

PHP поддерживает ООП (деструкторы, открытые, закрытые и защищённые члены и методы, интерфейсы и клонирование объектов). PHP поддерживает XML.

При запросе пользователя web-сервер просматривает документ, выполняет найденные в нем PHP-инструкции, а результат их выполнения возвращает пользователю. При этом статическая часть документа, написанная на языке HTML, фактически является шаблоном, а изменяемая часть формируется при исполнении PHP-инструкций. Сами скрипты находятся на сервере и их содержимое посетителю сайта просмотреть невозможно. В итоге пользователь видит обычную веб-страницу, которая от других отличается лишь расширением.

Интерпретатор PHP подключается к веб-серверу либо через модуль, созданный специально для этого сервера (например, для Apache или IIS), либо в качестве CGI-приложения [10].

Основные достоинства PHP:

1. простота модификации внешнего вида готового программного продукта (возможность создания множества шаблонов и переключения между ними);

2. большое количество бесплатных решений на этой платформе;

3. бесплатность самого интерпретатора и всех компонентов платформы LAMP (Linux, Apache, MySQL, PHP);

4. кроссплатформенность.

Основные недостатки PHP:

1. нестрогая типизация;

2. отсутствие поддержки многобайтовых кодировок в ядре языка.

Технология Microsoft ASP.Net

Microsoft .NET Framework – это платформа для создания, развертывания и запуска Web-сервисов и приложений. Она предоставляет высокопроизводительную, основанную на стандартах, многоязыковую среду, которая позволяет интегрировать существующие приложения с приложениями и сервисами следующего поколения, а также решать задачи развертывания и использования интернет-приложений. .NET Framework состоит из трех основных частей - общеязыковой среды выполнения (common language runtime), иерархического множества унифицированных библиотек классов и компонентной версии ASP, называемую ASP.NET.

ASP.NET - это часть технологии .NET, используемая для написания мощных клиент-серверных интернет приложений. Она позволяет создавать динамические страницы HTML. ASP.NET возникла в результате объединения более старой технологии ASP (активные серверные страницы) и .NET Framework. Она содержит множество готовых элементов управления, используя которые можно быстро создавать интерактивные web-сайты [11].

Процесс создания веб-форм ускоряется за счет использования стандартных встроенных компонентов и пользовательских компонентов, помогающих при написании кода страницы.

Основные достоинства технологии ASP.NET:

1. наличие средства визуального программирования позволяет сократить время;

2. реализация объектной модели приложения;

3. выполняемый код – откомпилированное приложение.

Основные недостатки ASP.NET:

1. для работы ASP.NET приложение нужно лицензионное программное обеспечение. (Windows Server + Microsft SQL, для разработки Microsoft Visual Studio);

2. платформозависимость;

3. достаточная сложность освоения модели классов .NET.

Технология Java EE

Java Platform, Enterprise Edition, сокращенно Java EE (до версии 5.0 – Java 2 Enterprise Edition или J2EE) – набор спецификаций и соответствующей документации для языка Java, описывающей архитектуру серверной платформы для задач средних и крупных предприятий [12].

Основная цель спецификаций – обеспечить масштабируемость приложений и целостность данных во время работы системы. J2EE во многом ориентирована на использование её через веб как в интернете, так и в локальных сетях.

Платформа J2EE использует модель многоуровневого распределенного приложения. Логически приложение разделено на компоненты в соответствии с их функциональностью. Различные компоненты, составляющие J2EE-приложение, установлены на различных компьютерах в зависимости от их уровня в многоуровневой среде J2EE, которой данный компонент принадлежит. J2EE-приложения состоят из компонентов. J2EE-компонента представляет собой законченный функциональный программный модуль, встроенный в приложение J2EE с соответствующими классами и файлами и взаимодействующий с другими компонентами. J2EE-спецификация определяет следующие J2EE-компоненты:

1. клиентские приложения и апплеты – это компоненты, работающие на клиентской машине;

2. компоненты технологии Java-сервлет и JavaServer Pages (JSP) – это Web-компоненты, работающие на сервере;

3. корпоративные компоненты – это бизнес-компоненты, работающие на сервере.

J2EE-компоненты пишутся на языке программирования Java и компилируются точно так же, как и любая другая Java-программа.

Спецификация Java EE предоставляет обширные возможности для разработки клиент-серверных приложений и в особенности удобные пути реализации серверной части приложения на языке Java. Клиентская же часть чаще всего основывается на стандартных html-страницах, дополненных java-аплетами и клент-приложениями [12].

Обоснование выбора технологии

Практически все рассмотренные технологии разработки веб-приложений функционально практически идентичны, т.е. все, что можно реализовать, используя одну из них, можно сделать и на любой другой. Основные различия состоят в скорости и общей стоимости разработки, масштабируемости полученного решения и удобства средств разработки. Однако, в каждом конкретном случае, можно выделить ряд существенных критериев, по которым одна из технологий выигрывает на фоне других, а значит более подходит для решения конкретно поставленной задачи и реализации разрабатываемого проекта. Результат такого сравнения приведен в табл. 2.1.

Таблица 2.1

Сравнение технологий разработки веб-приложений

Критерий	Интерпретируемый серверный язык PHP	ASP.NET	Java EE
Кроссплатформенность	+	-	+
Бесплатность	+	-	+
Поддержка ООП	+	+	+
Наличие библиотек	+	+	+
Простота освоения	+	-	-

Как видно из таблицы сравнения, технология ASP.NET не подходит для решения поставленной задачи, т.к. требует дополнительных расходов на приобретение лицензий. Технология Java EE в целом удовлетворяет выдвинутым требованиям, однако, имеет достаточно большой порог входа и требует больше времени на освоение. Интерпретируемый серверный язык PHP подходит для реализации поставленной задачи по всем параметрам.

3 РАЗРАБОТКА И ТЕСТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

3.1 Реализация базовых компонентов системы и базы данных

В качестве основной схемы проектирования приложения был выбран шаблон MVC (model-controller-view). Шаблон архитектуры Model-View-Controller (MVC) разделяет приложение на три основных компонента: модель, представление и контроллер. Шаблон MVC позволяет создавать приложения, различные аспекты которых (логика ввода, бизнес-логика и логика интерфейса) разделены, но достаточно тесно взаимодействуют друг с другом. Эта схема указывает расположение каждого вида логики в приложении. Пользовательский интерфейс располагается в представлении. Логика ввода располагается в контроллере. Бизнес-логика находится в модели. Это разделение позволяет работать со сложными структурами при создании приложения, так как обеспечивает одновременную реализацию только одного аспекта. Например, разработчик может сконцентрироваться на создании представления отдельно от бизнес-логики.

На рис. 3.1 приведена схема взаимодействия основных компонентов приложения, построенного по шаблону MVC.

Рис. 3.1. Шаблон MVC

Для разработки современных веб-приложений в настоящее время, как правило, используются один из фреймворков – программной платформы, определяющей структуру программной системы разработку и объединение разных компонентов большого программного проекта.

В основе большинства современных PHP фреймворков лежит шаблон проектирования MVC. Наиболее распространенными фрейворками, основанными на шаблоне проектирования MVC являются Symphony, Codeigniter, Zend Framework и Yii.

Как основной инструмент для реализации разрабатываемого приложения был выбран фреймворк Codeigniter – популярный MVC фреймворк с открытым исходным кодом, написанный на языке программирования PHP, для разработки полноценных веб-систем и приложений. Этот фреймворк является свободно распространяемым и предоставляет разработчику разнообразные компоненты для работы с различными СУБД, сессиями, организации системы разграничения прав пользователей и реализации других типовых функций.

Архитектура приложения создавалась по требованиям шаблона проектирования MVC. Общий вид модульной структуры приложения приведен на рис. 3.2.

Основой приложения является 6 контроллеров, по одному на каждый функциональный раздел приложения. Они обеспечивают выполнение основных операций по обработке данных, хранимых в системе путем вызова методов соответствующих моделей и представлений.

Так IndexController предназначен для отображения главной страницы приложения. На главной странице не выводится никакой информации из базы данных, поэтому этот контроллер не использует моделей.

Он вызывает представление Index, которое содержит html код главной страницы и представления Header и Footer, которые содержат общий для всех страниц элементы (заголовки html страницы, подключение javascript и css).

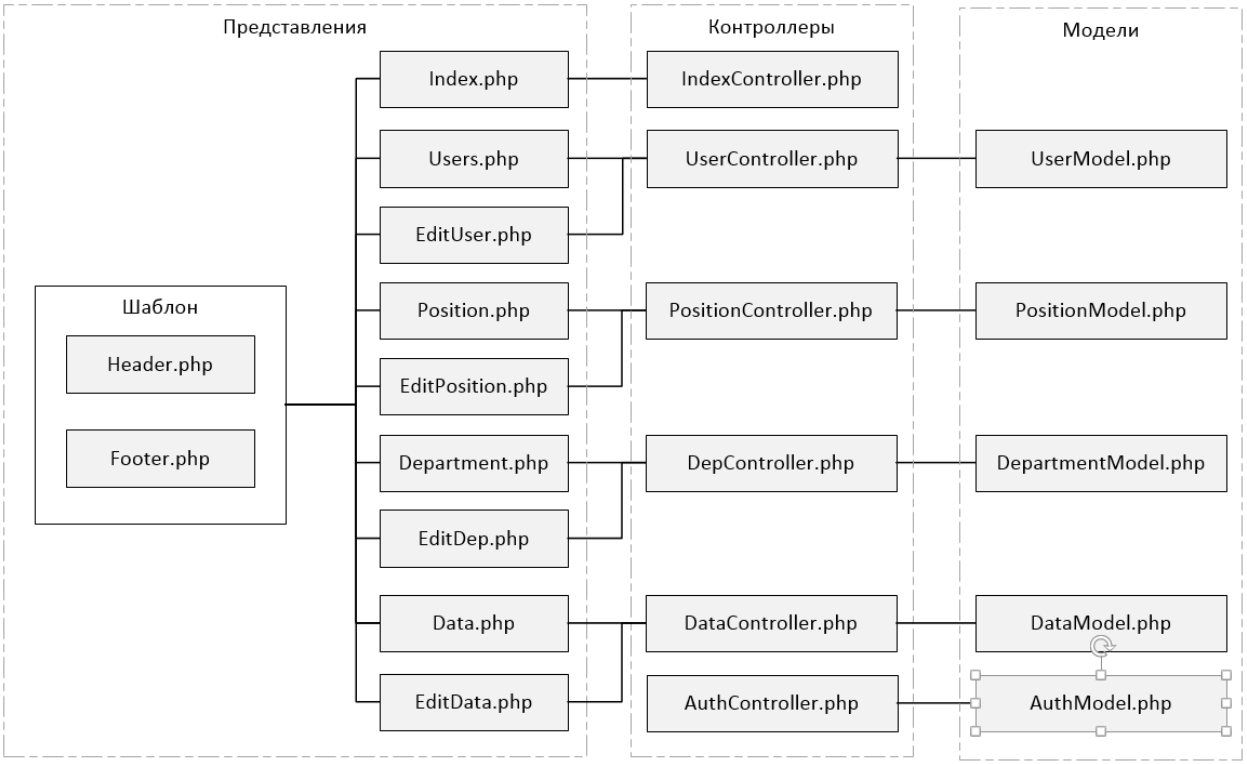

Рис. 3.2. Архитектура приложения в соответствие с шаблоном MVC

Контроллер UserController отвечает за отображение списка пользователей (представление Users) и отображение формы редактирования и данных о пользователе (представление EditSubject). Он использует модель UserModel, которая содержит методы для выбора, удаления и редактирования информации о пользователях в базе данных.

Контроллер PositionController отвечает за отображение списка должностей (представление Position) и формы добавления/редактирования информации о должностях – представление EditPosition. Он использует модель PositionModel, которая содержит методы для выбора, удаления и редактирования информации о должностях в базе данных.

Контроллер DepController отвечает за отображение списка подразделений (представление Department) и формы добавления и редактирования информации о должностях – представление EditDep. Он использует модель DepartmenModel, которая содержит методы для выбора, удаления и редактирования подразделений.

Контроллер DataController отвечает за отображение данных о потреблении электроэнергии (представление Data) и интерфейса добавления/ редактирования данных о потреблении. Он использует модель DataModel, которая содержит методы для выбора, добавления и удаления информации о потреблении электроэнергии.

Механизм авторизации реализуется в контроллере AuthController. Данный контроллер содержит функции проверки достоверности введенных пользователем пары логин-пароль, проверки, авторизован ли текущий пользователь, а также методы для выхода из системы. Данный контроллер пользуется методами модели AuthModel, которая реализует функционал для обращения к данным о пользователях в базе данных.

Механизм авторизации

Механизм авторизации и проверки прав пользователей реализуется на основе использования сессий. Сессии - это специальный механизм веб-сервера, позволяющий организовать передачу данных от одной веб-страницы к другой. Стандартный протокол HTTP не имеет памяти и в нем каждый запрос к серверу рассматривается как отдельный, не связанный с другим объектом.

CodeIgniter поставляется с несколькими драйверами хранения сессий:

– файлы;

– базы данных;

– memcached (кеширование данных в оперативной памяти на основе хеш-таблицы).

Данные сессии – просто массив, связанный с конкретным ID сессии (куки).

Данные куки сохраняются в виде сериализованного массива как показано на примере листинга 1.

Листинг 3.1. Пример сохранения данных куки

```
[array]
(
```

Листинг 3.1. Продолжение

```
        'IDsession'    => случайных хеш,
        'IPaddress'        => 'строковое значение IP адреса
клиента',
        'userAgent'        => 'строковое значение userAgent
клиента',
        'lastActivity' => время последнего обращения к серверу
    )
```

Конец листинга

Для того чтобы получить доступ к элементам массива сессии использовать функцию, представленную на листинге 3.2.

Листинг 3.2. Пример получения доступа к элементам массива

```
$this->session->userdata('ключ');
```

Конец листинга

Где 'ключ' это индекс массива, который соответствует элементу, который необходимо получить. Так, чтобы получить ID сессии, следует воспользоваться методом объекта session.

Листинг 3.3. Пример использования объекта session

```
$sessionId = $this->session->userdata('sessionId');
```

Конец листинга

Чтобы добавить данные в массив сессии, следует передать массив содержащий данные в метод объекта session.

Листинг 3.4. Пример использования session для добавления данных

```
$this->session->set_userdata($arrayData);
```

Конец листинга

Где $arrayData является ассоциативным массивом, который содержит ваши новые данные.

Листинг 3.5. Пример добавления данных в массив сессий

```
$sessiondata = array(
   'user'  => 'michael',
   'emailAddress'     => 'michael@google.com',
   'logeed' => TRUE
);
$this->session->set_userdata($sessiondata);
```

Конец листинга

Если требуется удалить данные из сессии, можно использовать методом объекта session, как представлено на листинге 3.6.

Листинг 3.6. Пример удаления данных из сессий

```
$this->session->unset_userdata('any_key_value');
```

Конец листинга

Работа с базой данных в Codeigniter

Для работы с базой данных в Codeigniter в соответствии с шаблоном MVC используются модели. Модели в codeigniter – это специальные PHP классы, предназначенные для работы с информацией в базе данных под управлением различных СУБД.

Классы моделей codeigniter должны хранится в директории application/models/. Структура каталогов в данной директории может быть вложенная. Базовый прототип для модели класса представлен на листинге 3.7

Листинг 3.7. Базовый прототип для модели класса

```
class Catalogmodel extends Model {
       function Catalogmodel() {
         parent::Model();
       }
     }
```

Конец листинга

После этого нужно подключить библиотеку для работы с базами данных. Для этого необходимо открыть файл «system/application/config/autoload.php» и отредактировать переменную «$autoload['libraries']». Она содержит массив с названиями всех библиотек, которые будут автоматически загружаться при обращении к сайту. Можно добавить загрузку библиотеки database.

Листинг 3.8. Пример редактирования переменной $autoload['libraries']

```
$autoload['libraries'] = array('database');
```

Конец листинга

Одним из преимуществ CodeIgniter является низкое потребление ресурсов. Достигается это в основном за счет того, что по умолчанию загружается только минимально необходимый для работы набор библиотек (ядро). Какие из дополнительных библиотек загружать решает разработчик. И CodeIgniter предоставляет несколько вариантов их загрузки.

В данном случае можно добавить название библиотеки в $autoload, если все методы контроллера будут работать с БД. Если данные из БД нужны только для создания части страниц, то лучше загружать библиотеку непосредственно перед использованием.

Листинг 3.9. Пример загрузки библиотеки database

```
$this->load->database();
```

Конец листинга

CodeIgniter позволяет работать одновременно с несколькими БД, в разрабатываемом приложении эта возможность использоваться не будет. Чтобы настроить подключение к базе данных необходимо открыть файл «system/application/config/database.php» и заполнить массив с параметрами подключения.

Листинг 3.10. Пример работы CodeIgniter с нескольких БД

```
$db['default']['hostname'] = "localhost"; //имя хоста, на котором запущен сервер БД
$db['default']['username'] = "имя_пользователя_БД";
$db['default']['password'] = "пароль";
$db['default']['database'] = "bookcatalog"; //имя базы данных
```

Конец листинга

Модели, как правило, используются из контроллера. Чтобы загрузить модель следует использовать метод, представленный на листинге 3.11.

Листинг 3.11. Пример загрузки модели

```
$this->load->model('myModelName');
```

Конец листинга

Ниже приведен фрагмент контроллера, который загружает модель, затем передает полученные данные в представление на примере отображения списка клиентов.

Листинг 3.12. Фрагмент контроллера, который загружает модель

```
class SubjectsController extends CI_Controller {
     public function index() {
          $this->load->model('SubjectsModel');
          $d['q'] = $this->SubjectsModel->get_all();
          $this->load->view('Subjects', $d);
     }
}
```

Конец листинга

Среда разработки

Для упрощения процесса работки веб-приложений в настоящее время используются интегрированные среды разработки (IDE) – комплекс

программных средств, используемый для разработки программного В области веб-разработки основными конкурентами на рынке являются PhpStorm IDE, NetBeans и Eclipse PDT. Все они имеют различные достоинства и недостатки. В качестве среды для создания разрабатываемого приложения была выбрана IDE PhpStorm. Это коммерческая кроссплатформенная интегрированная среда разработки для PHP. Разрабатывается компанией JetBrains на основе платформы IntelliJ IDEA. Для студентов представляется бесплатная лицензия.

PhpStorm – это интегрированная среда разработки на PHP с интеллектуальным редактором, которая глубоко понимает код, поддерживает PHP 7.1, 7.0, 5.6, 5.5, 5.4 и 5.3 для современных и классических проектов, обеспечивает автодополнение кода, рефакторинг, предотвращение ошибок налету и поддерживает смешивание языков.

На рис.3.3 приведен внешний вид среды разработки.

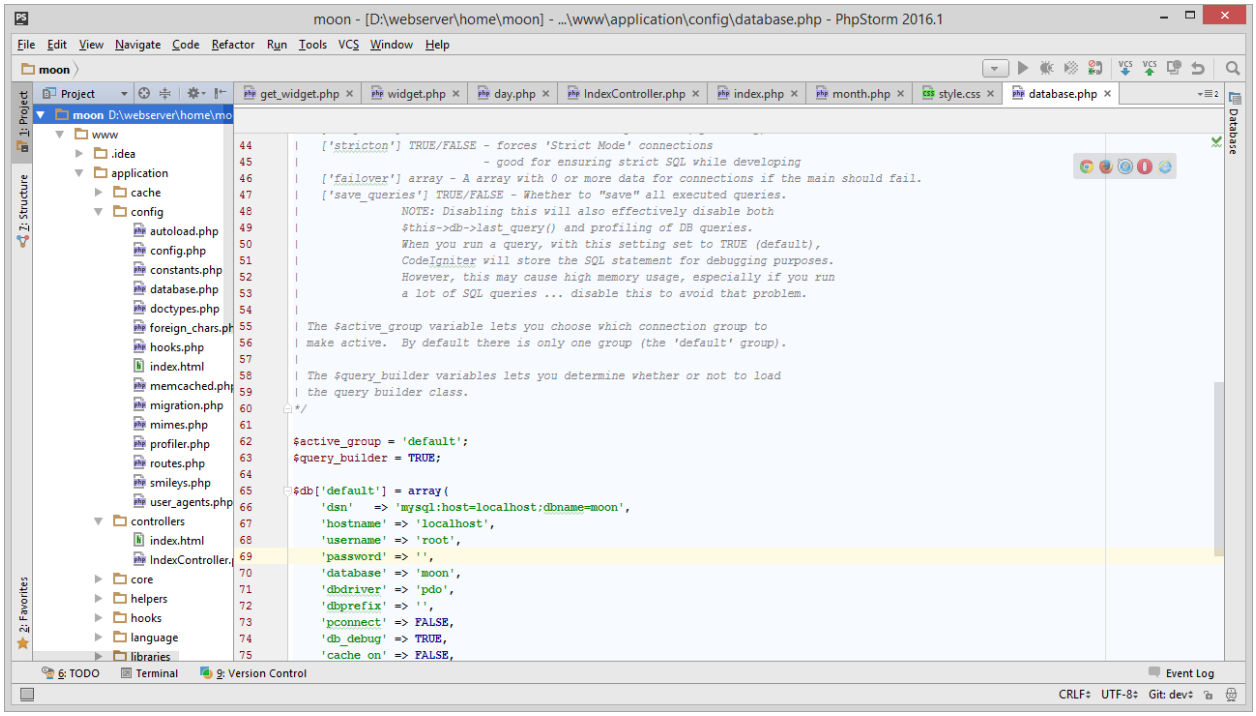

Рис. 3.3. Внешний вид IDE PHPStorm

Реализация базы данных. В качестве СУБД в данной работе используется MySQL. Удобным инструментом для работы с базой данных

под управлением MySQL является среда MySQL Workbench. MySQL Workbench – инструмент для визуального проектирования баз данных, интегрирующий проектирование, моделирование, создание и эксплуатацию БД в единое бесшовное окружение для системы баз данных MySQL. Данная среда позволяет:

– наглядно представить модель базы данных в графическом виде;

– производить Reverse Engineering – восстановление структуры таблиц из уже существующей на сервере БД;

– использовать удобный редактор SQL запросов, позволяющий сразу же отправлять их серверу и получать ответ в виде таблицы;

– возможность редактирования данных в таблице в визуальном режиме.

Внешний вид системы представлен на рис. 3.4.

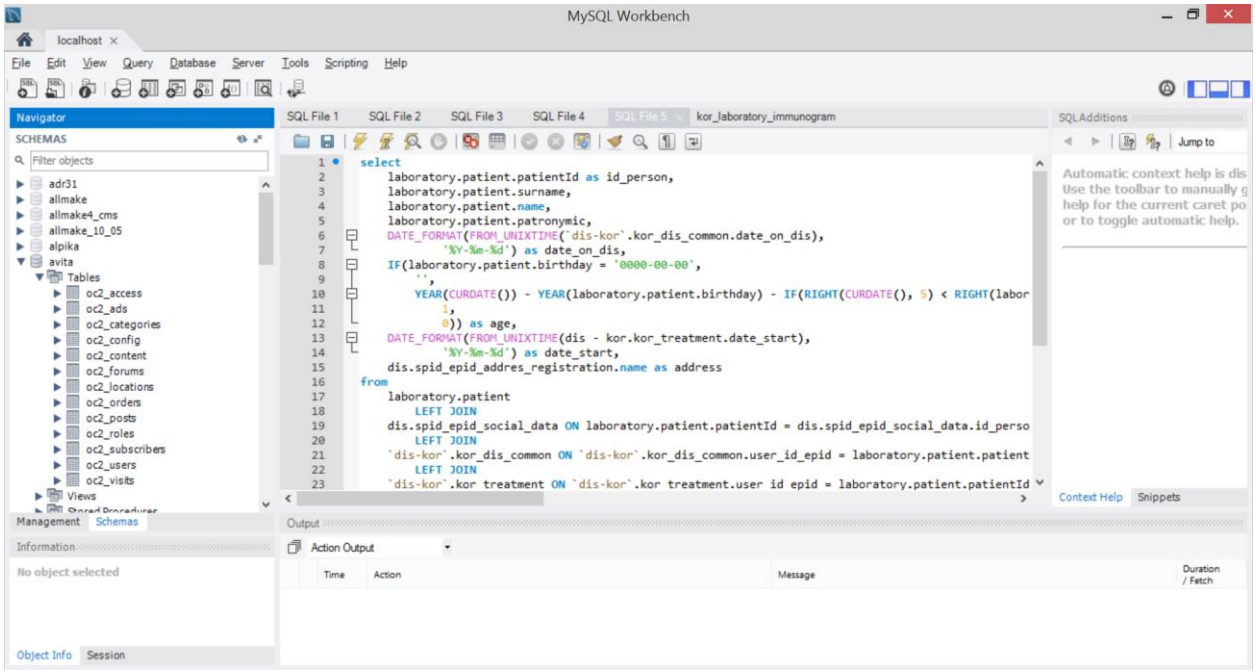

Рис. 3.4. Внешний вид среды MySQL Workbench

3.2 Реализация пользовательского интерфейса

На основе требований к системе документооборота интернет агентства были выделены основные экраны пользовательского интерфейса:

1. главная страница (форма входа);
2. страница просмотра данных о потреблении электроэнергии;
3. форма редактирования данных о потреблении;
4. форма построения прогноза;
5. форма импорта данных;
6. список пользователей;
7. список должностей;
8. форма редактирования информации о должности;
9. список подразделений;
10. форма редактирования информации о подразделении.

Взаимосвязь между основными экранами разрабатываемой системы показана на рис. 3.5.

Рис. 3.5. Схема взаимодействия основных экранов пользовательского интерфейса

Для создания пользовательского интерфейса веб-приложений в настоящее время активно применяются различные фреймворки. Наиболее простым в использовании и распространённым является Bootstrap. Bootstrap – это CSS фреймворк, который изначально создавался для внутреннего использования компанией «Twitter» с рабочим названием «Twitter Blueprint», но в итоге был опубликован в открытый доступ и стал хорошим набором инструментов для разработки клиенткой части веб-пиложений под названием «Bootstrap». Перечислим основные преимущества Bootstrap.

– Высокая скорость разработки макетов страниц сайта. Bootstrap содержит огромный набор готовых решений и элементов.

– Кроссбраузерность и адаптивность сайта. Все элементы фреймворка адаптивны под все устройства и корректно отображаются во всех современных браузерах.

– Простота в обучении. У Bootstrap очень хорошая документация с большим количеством примеров готового кода.

В рамках данной работы использовался бесплатный онлайн-инструмент jetstrap.com. Результатом проектирования является исходный код html, который можно использовать в готовом проекте.

На рис. 3.6 приведен процесс визуального проектирования главной страницы приложения в онлайн-редакторе. Главная страница состоит из заголовка страницы и формы входа. После авторизации пользователь попадет в основное приложение, в котором ему будут доступны весь функционал.

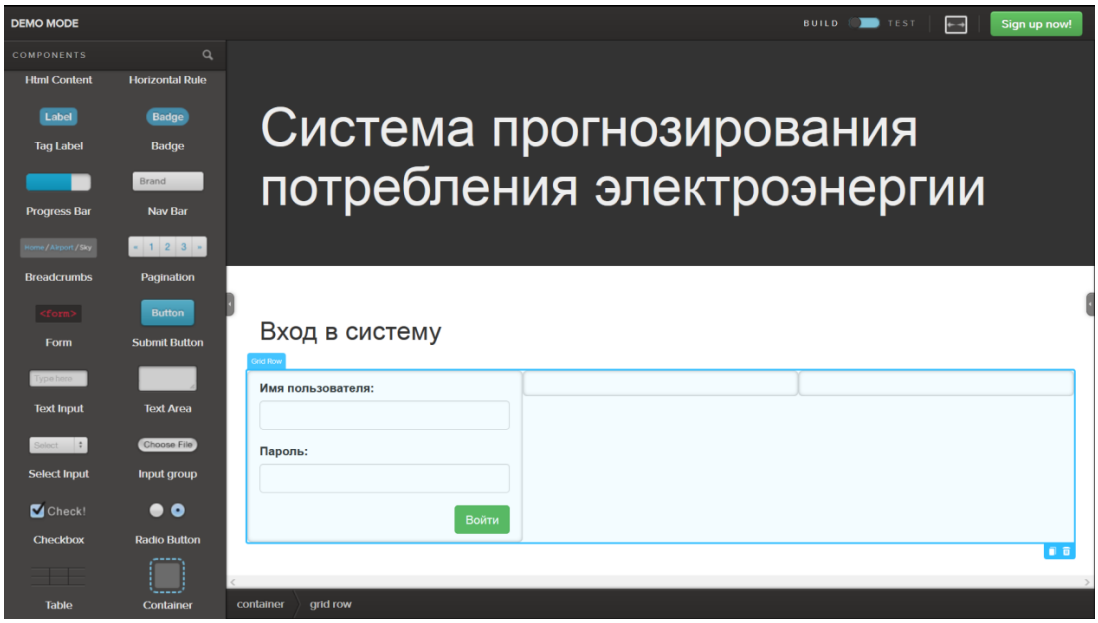

Рис. 3.6. Визуальное проектирование интерфейса главной страницы

На рис. 3.7 показан процесс визуального проектирования основной формы приложения.

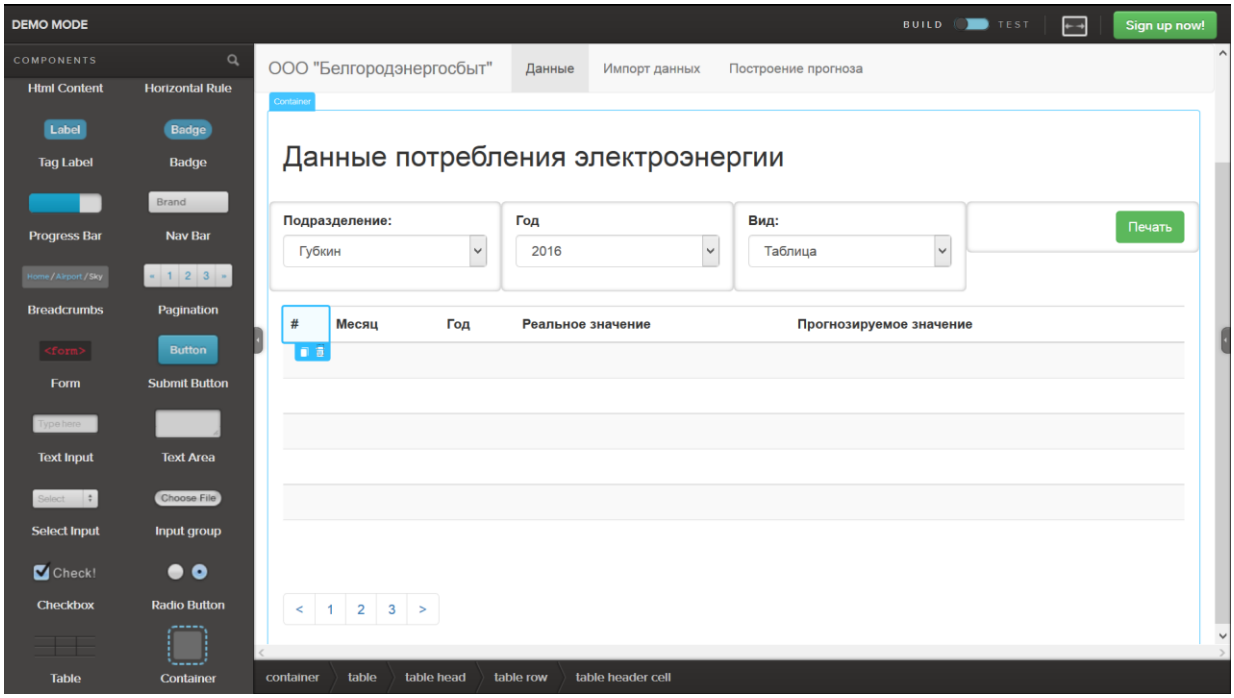

Рис. 3.7. Визуальное проектирование главной формы приложения

На рис. 3.8 показан процесс визуального проектирования формы импорта данных.

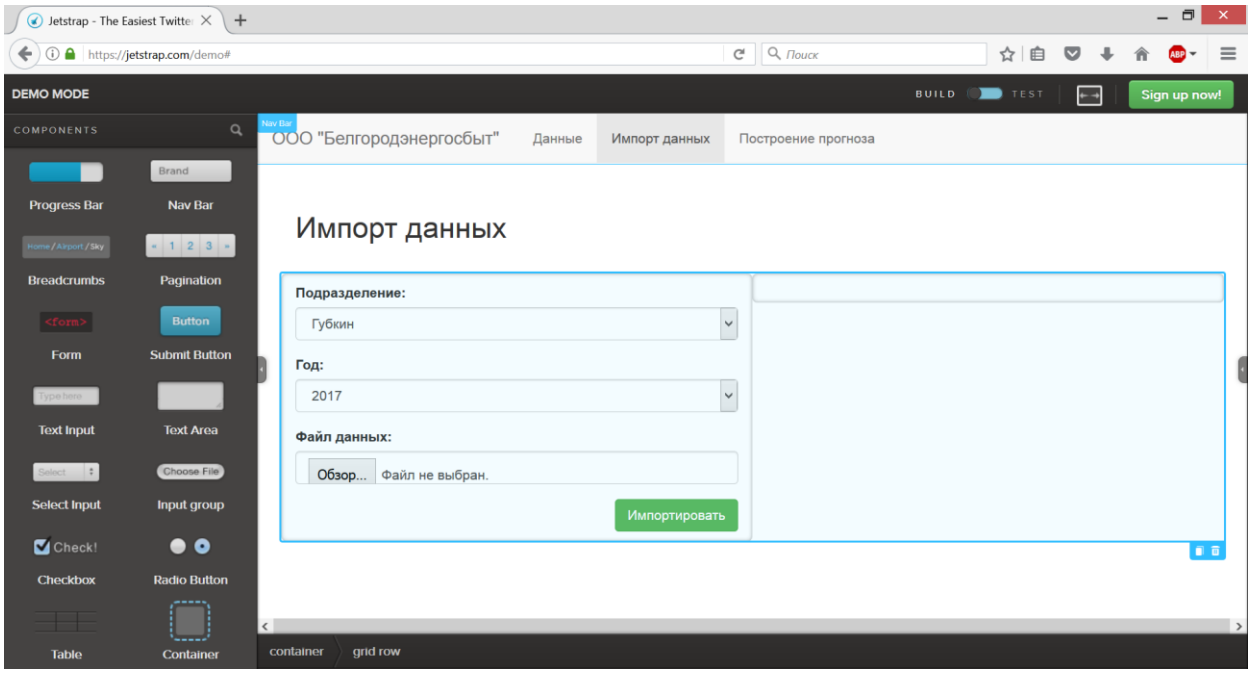

Рис. 3.8. Визуальное проектирование формы импорта данных

На рис. 3.9 показан процесс визуального проектирования формы построения прогноза за выбранный период.

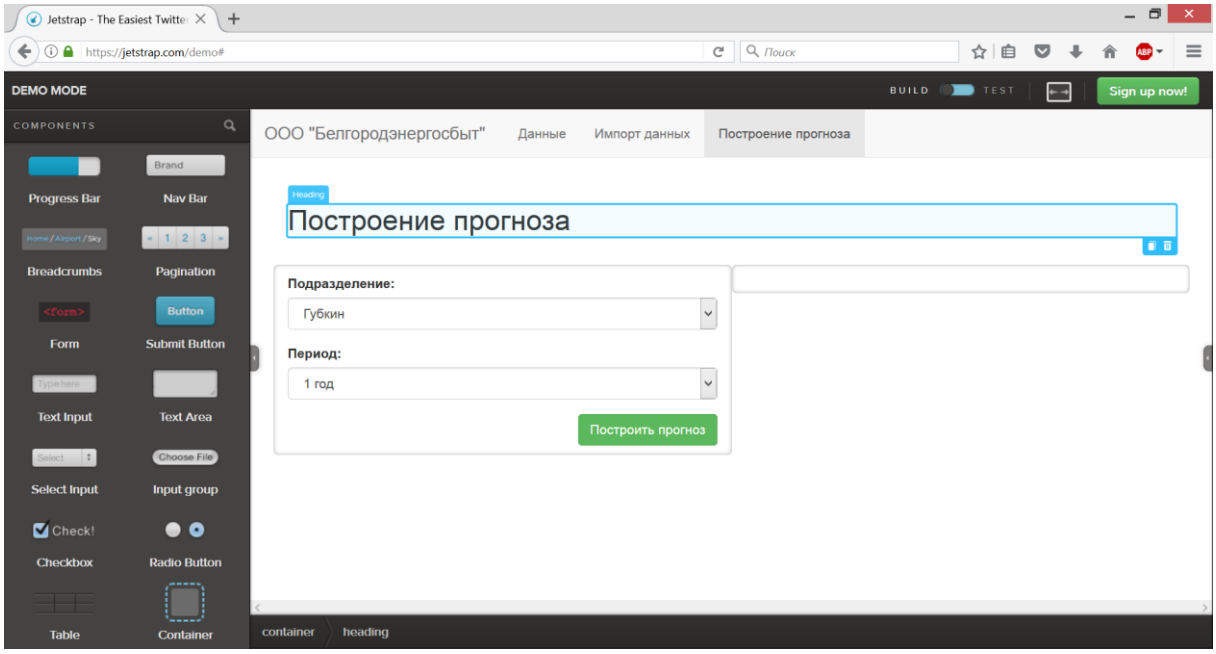

Рис. 3.9. Визуальное проектирование формы построения прогноза

Интерфейс администратора создавался аналогичным образом, были спроектированы формы добавления, удаления и редактирования информации о пользователях, подразделениях и должностях.

На рис. 3.10 представлен процесс визуального проектирования страницы отображения списка пользователей.

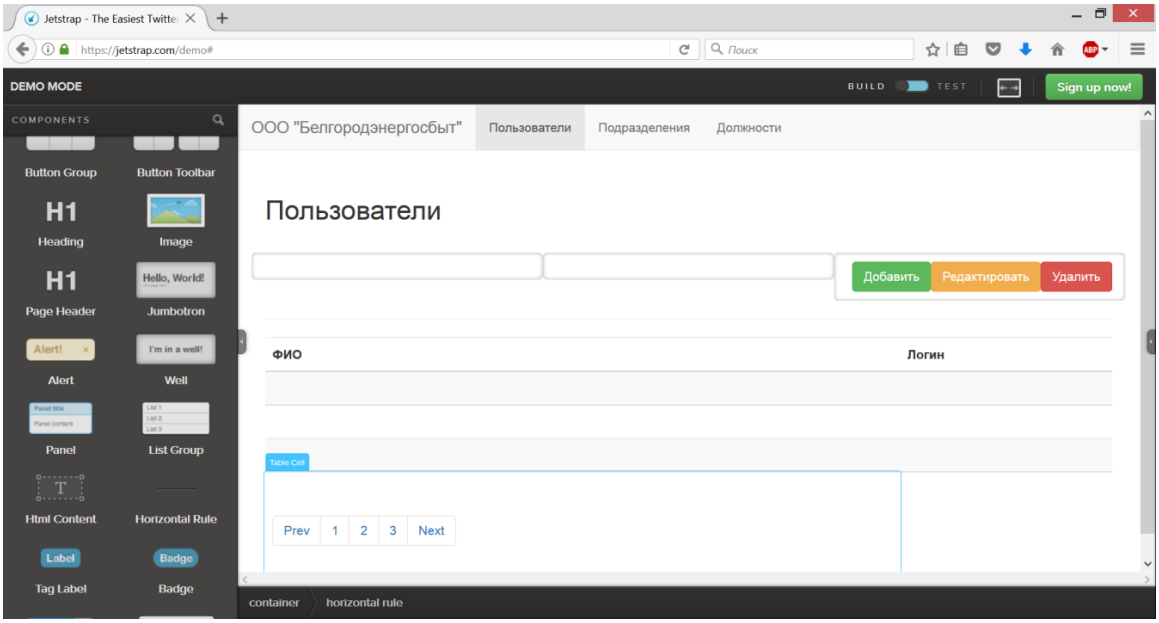

Рис. 3.10. Визуальное проектирование страницы отображения пользователей

На рис. 3.11 представлен процесс визуального проектирования формы редактирования информации о пользователе.

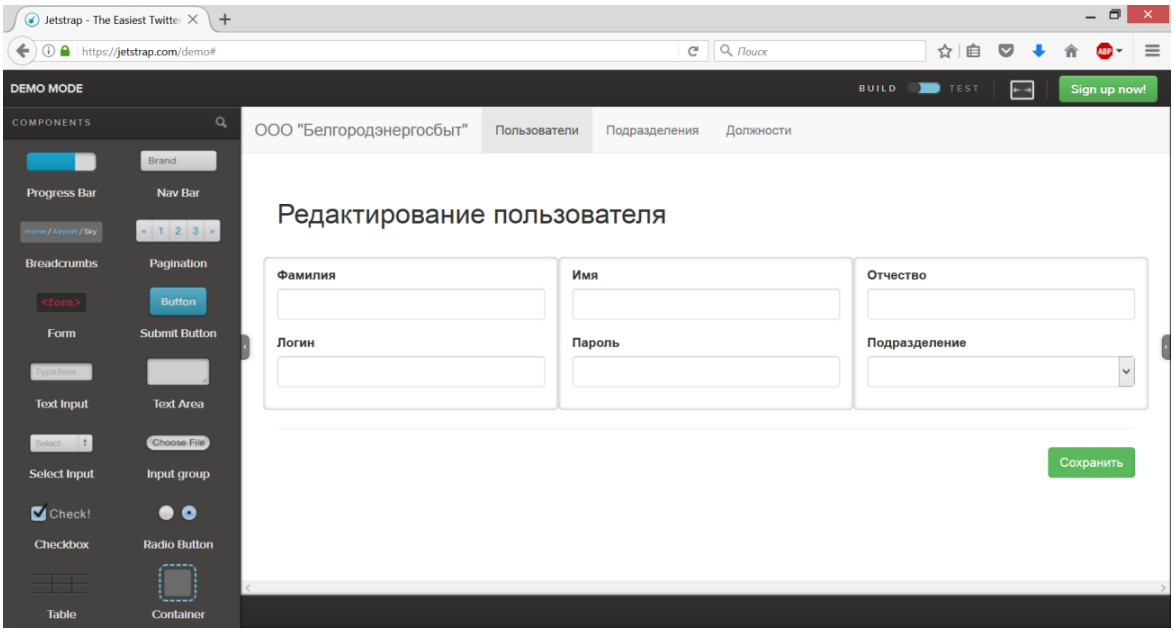

Рис. 3.11. Форма редактирования информации о пользователе

На вкладке «Подразделения» администратор имеет возможность просмотреть список подразделений, добавить новое, удалить и

отредактировать существующие подразделения. На рис. 3.12 представлен процесс визуального проектирования страницы отображения списка пользователей.

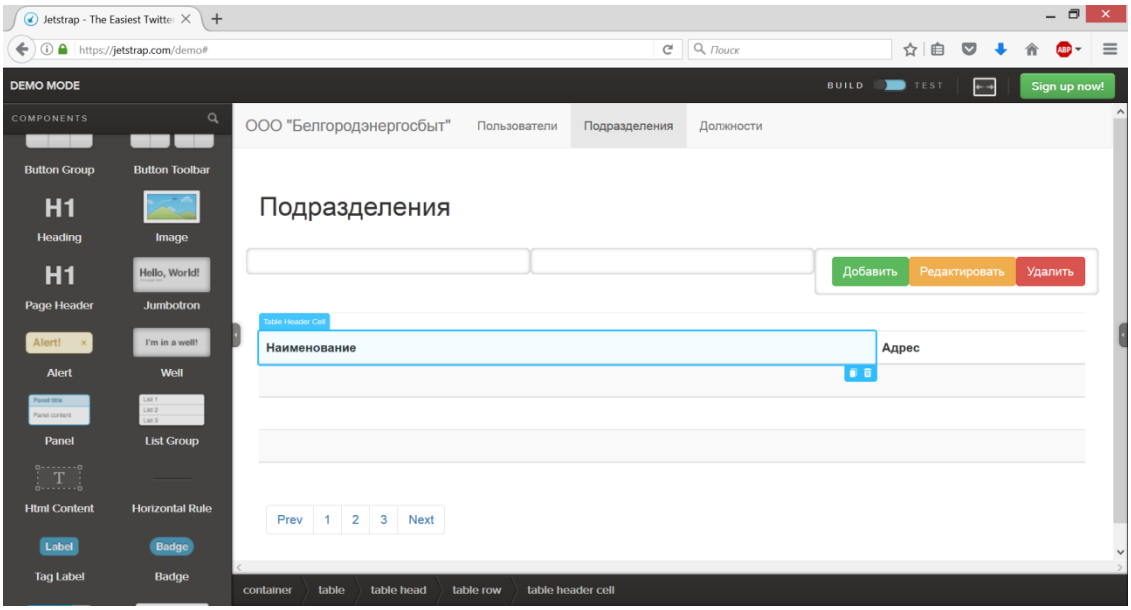

Рис. 3.12. Визуальное проектирование страницы отображения подразделений

На рис. 3.13 представлен процесс визуального проектирования формы редактирования информации о подразделении.

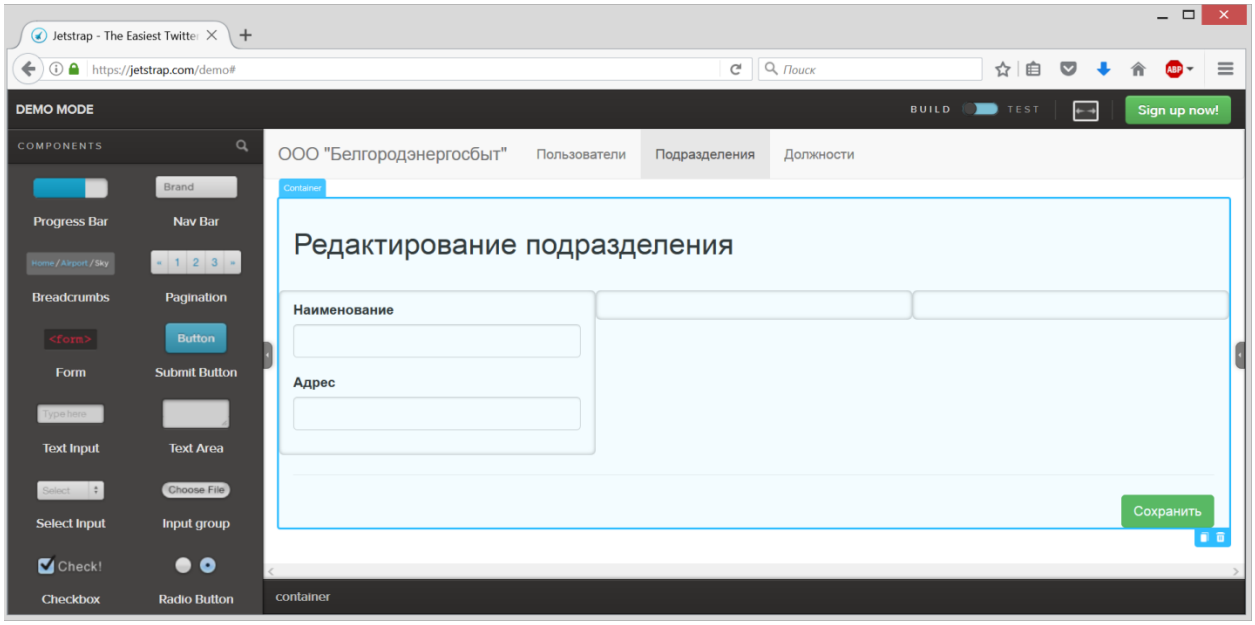

Рис. 3.13. Форма редактирования подразделения

На вкладке «Должности» администратор имеет возможность просмотреть список должностей, добавить новую, удалить и отредактировать существующие должности. На рис. 3.14 представлен процесс визуального проектирования страницы отображения списка должностей.

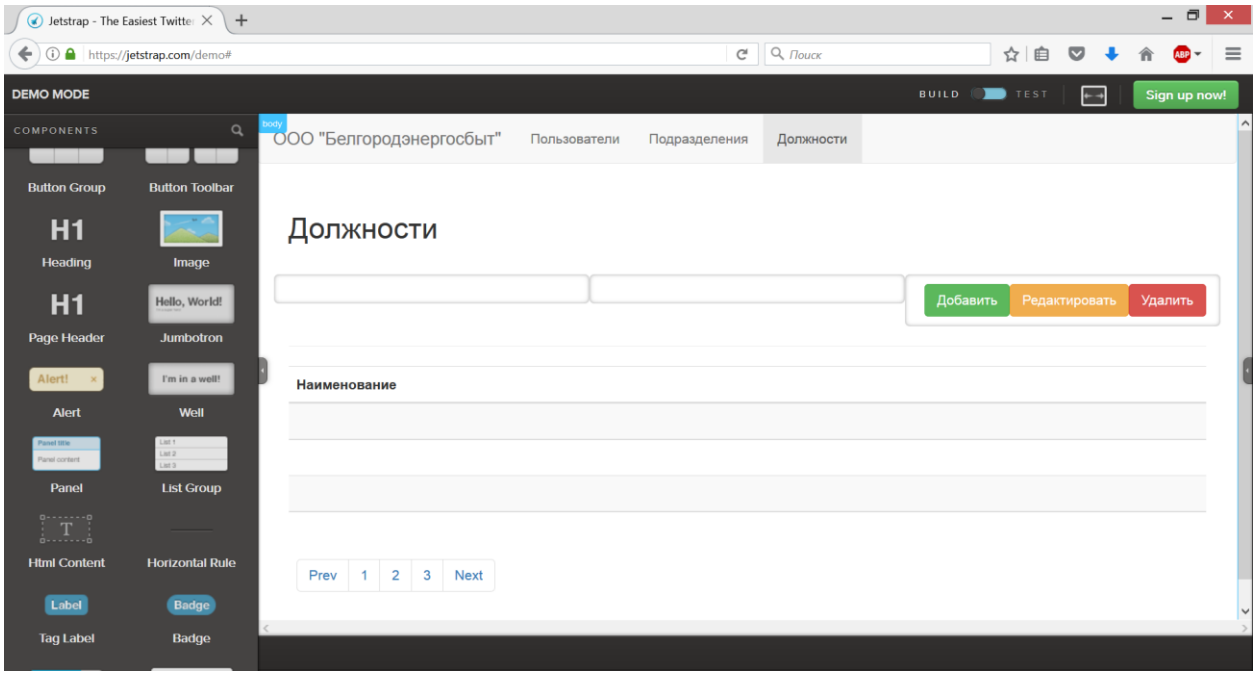

Рис. 3.14. Визуальное проектирование страницы отображения должностей

На рис. 3.15 представлен процесс визуального проектирования формы редактирования информации о должности.

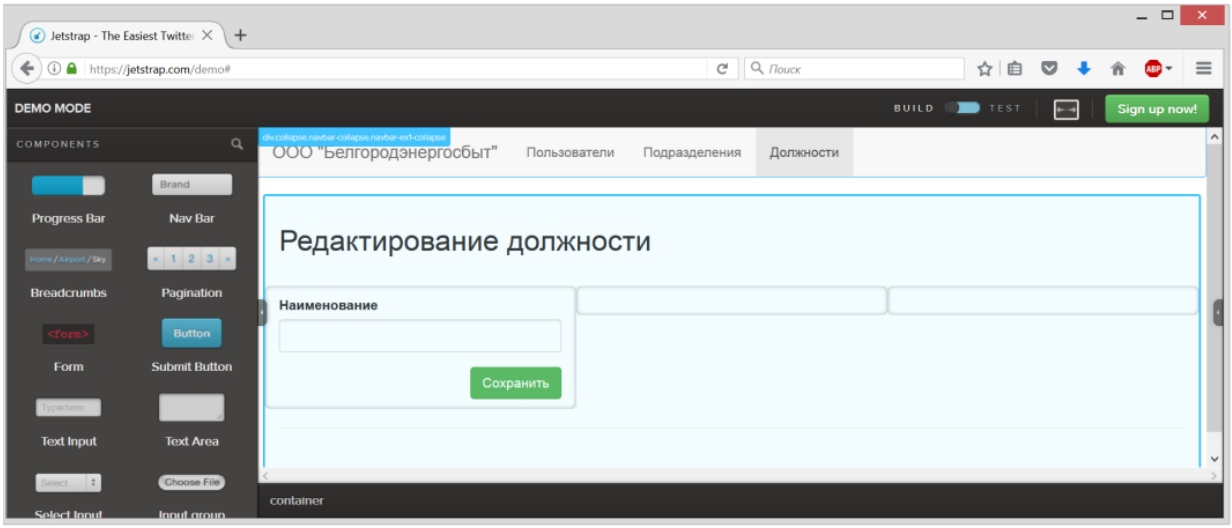

Рис. 3.15. Визуальное проектирование формы отображения должностей

3.3 Тестирование программного комплекса

Покажем основные этапы работы с созданным приложением. Все пользователи приложения делятся на две группы: администратор и пользователи. Каждый из пользователей имеет доступ к ограниченному набору функций в соответствии с группой, к которой он принадлежит.

На главной странице приложения расположена форма авторизации. Для входа в систему пользователь должен ввести имя пользователя и пароль. На рис. 3.16 показан внешний вид главной страницы приложения.

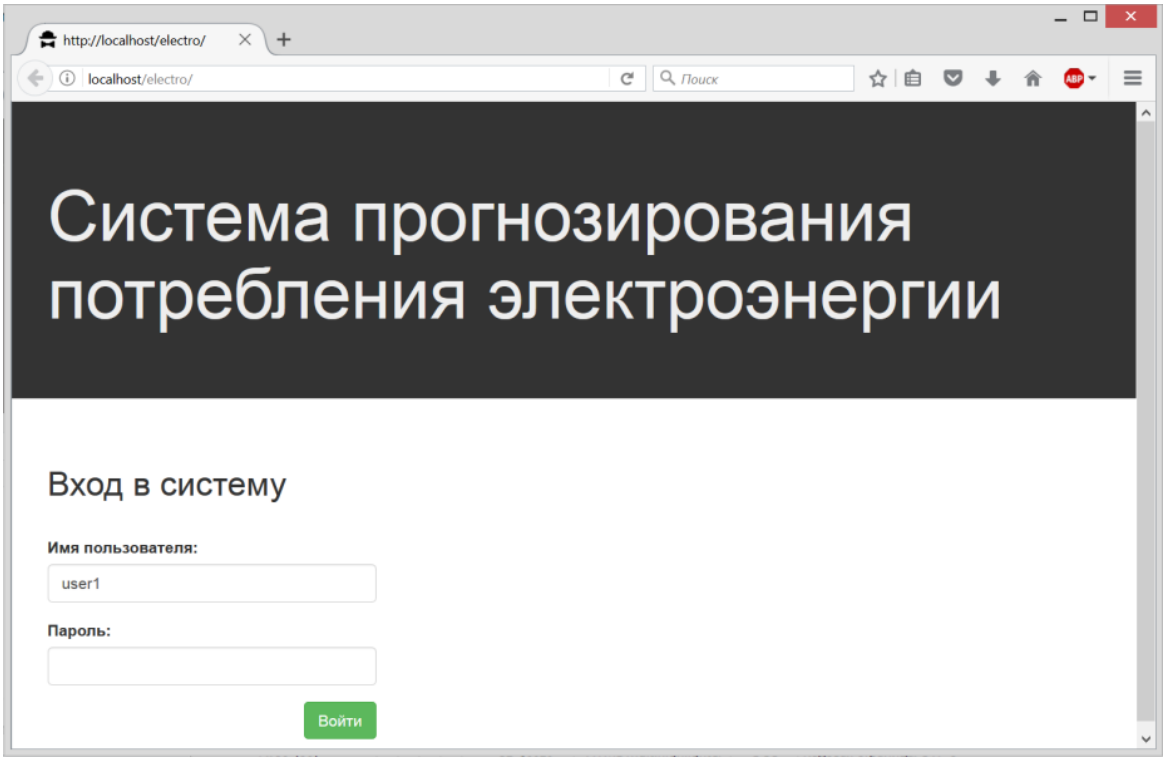

Рис. 3.16. Вход в систему

После авторизации пользователь попадает на страницу личного кабинета. На основном экране личного кабинета пользователь имеет доступ к трем разделам: «Данные», «Импорт данных» и «Построение прогноза».

На вкладке «Данные» пользователь имеет возможность просматривать исторические и прогнозируемые данные по потреблению электроэнергии. На панели в верхней части страницы имеется возможность выбрать

подразделение, для которого выводится информация, период за который выводятся данные и вид отображения (таблица или график).

Справа вверху расположена кнопка «Печать», которая позволяет вывести на печать текущую отображаемую информацию.

Для того чтобы произвести сортировку данных в таблице необходимо нажать на заголовок любого столбца таблицы. По первому щелчку данные сортируются по возрастанию, по второму щелчку по убыванию.

На рис. 3.17 отображена форма «Данные» для периода «2014» год.

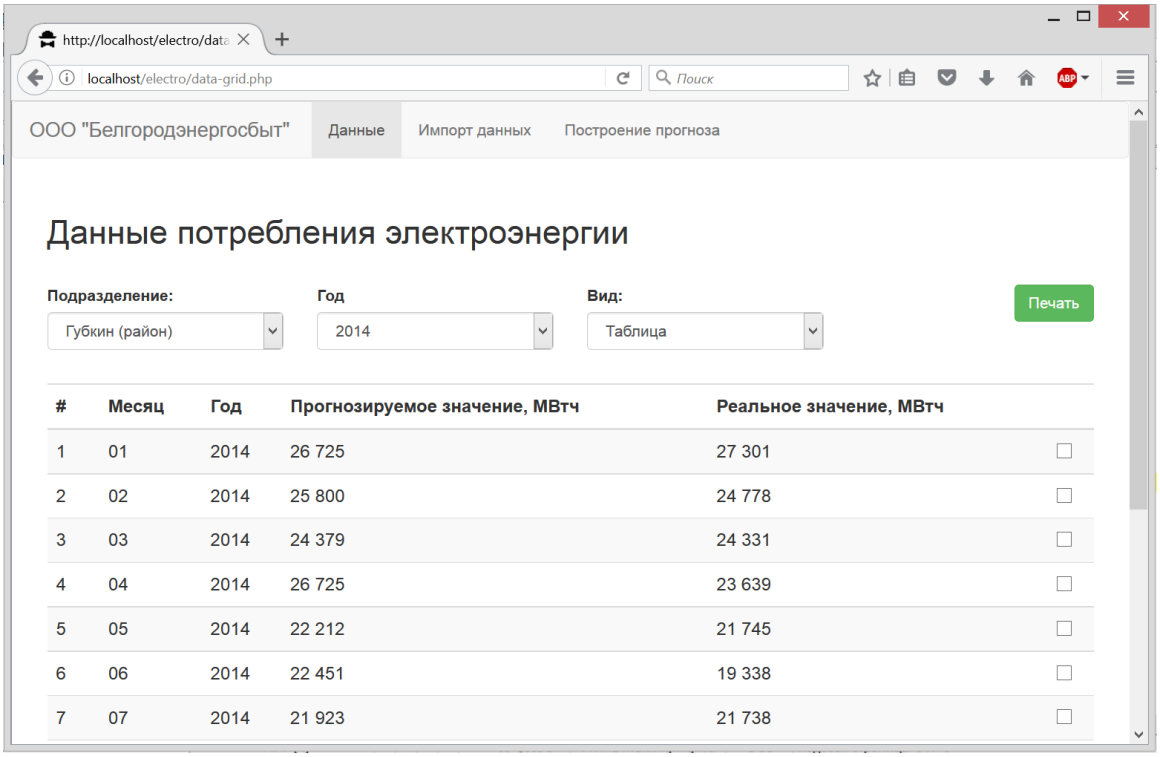

Рис. 3.17. Список данных по потреблению электроэнергии за 2014 год

Отображение данных в виде таблицы затрудняет их анализ. В то же время, отображение данных в виде графика позволяет проследить тренд в данных и наиболее полно отобразить скрытые зависимости.

На рис. 3.18 отображен график фактического и спрогнозированного потребления электроэнергии за 2014 год.

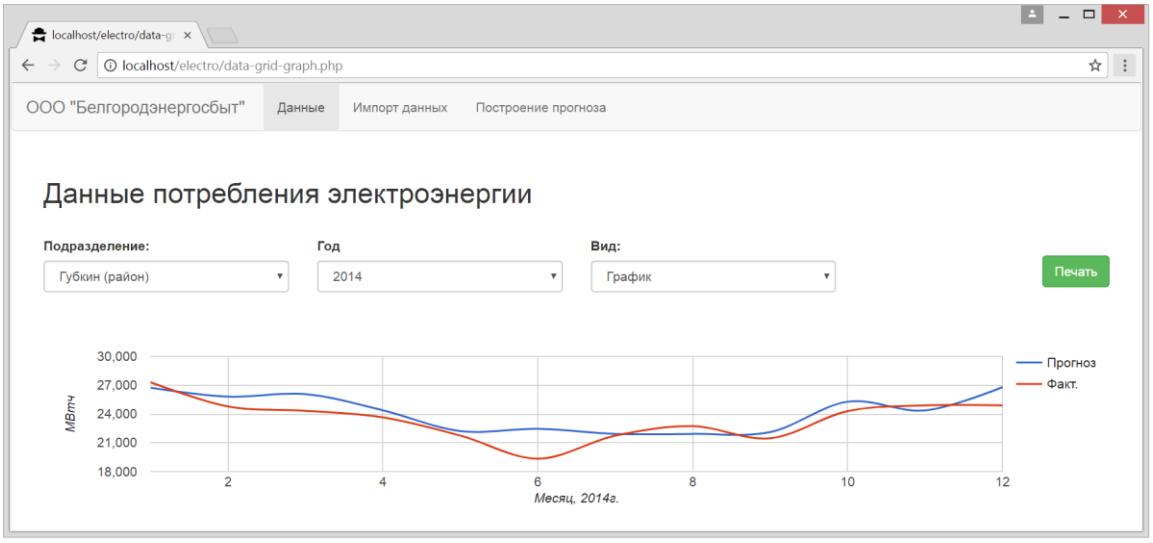

Рис. 3.18. Отображение потребления электроэнергии в виде графика

На вкладке «Импорт данных» располагается форма импорта данных из существующей системы учета. Файлы для импорта предоставляются в формате xls. Для того чтобы импортировать файл с данными, необходимо выбрать подразделение, для которого предполагается произвести импорт и период, за который предоставляются данные. На рис. 3.19 показана форма импорта данных.

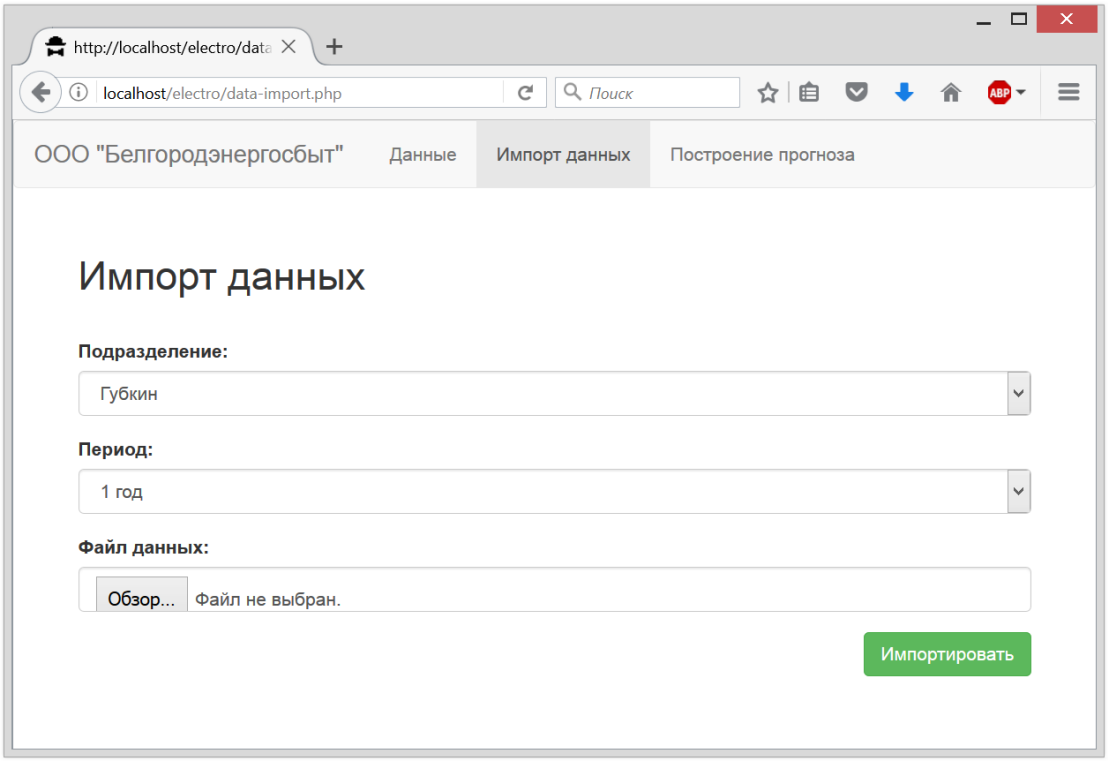

Рис. 3.19. Форма импорта данных

Для построения прогноза по данным необходимо перейти на вкладку «Построение прогноза». Для того чтобы построить прогноз, необходимо выбрать подразделение из списка доступных и период прогнозирования, после чего нажать кнопку «Построить прогноз». На рис. 3.20 представлен внешний вид формы построения прогноза по данным.

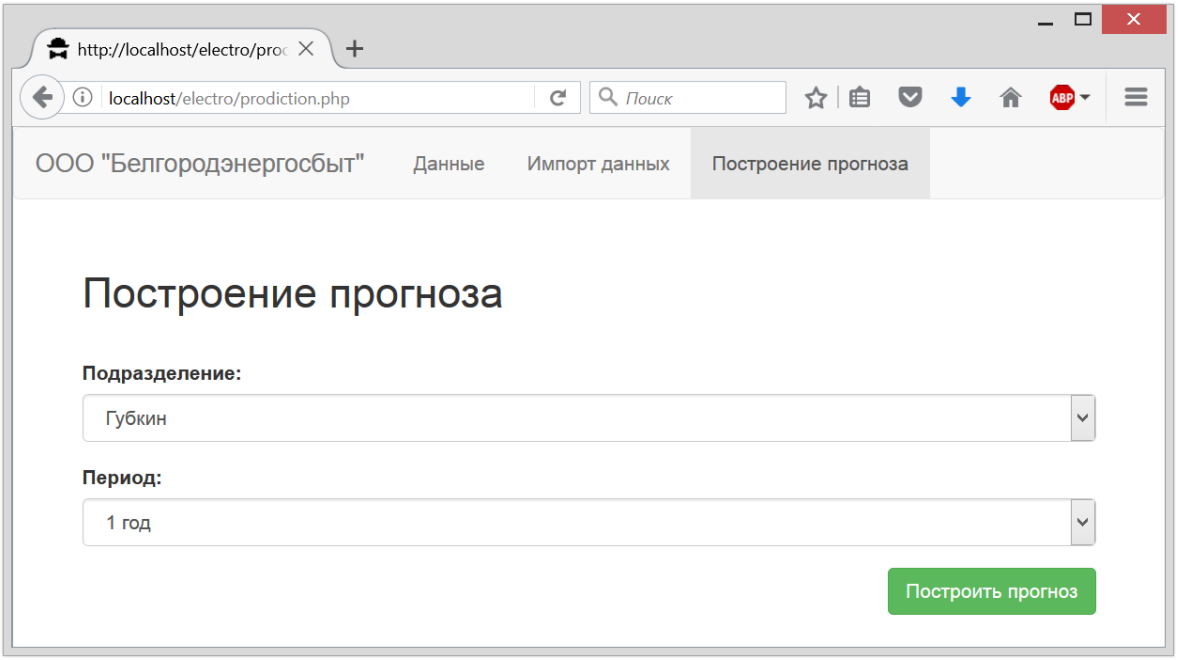

Рис. 3.20. Форма построения прогноза

Интерфейс администратора системы содержит формы для редактирования, добавления и удаления информации. Он построен на базовых формах и представляет собой стандартную реализацию функционала ведения справочников и имеет три вкладки «Пользователи», «Подразделения» и «Должности».

На рис. 3.21 приведен внешний вид вкладки «Пользователи». В этом интерфейсе администратор системы имеет возможность просмотреть список пользователей, добавить нового пользователя, удалить и отредактировать существующих.

Для перехода на форму редактирования необходимо выбрать текущего пользователя с помощью компонента checkbox и нажать кнопку «Редактировать». При нажатии на кнопку «Удалить», текущий выбранный

пользователь будет удален. При нажатии на кнопку «Добавить» откроется форма добавления информации о новом пользователе.

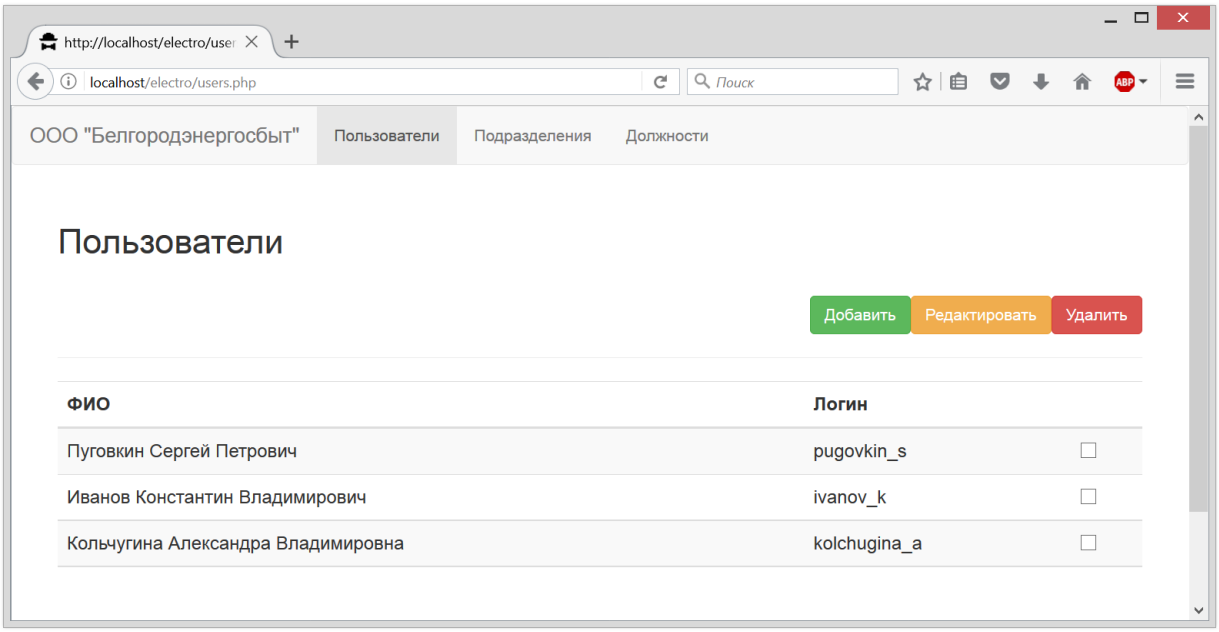

Рис. 3.21. Интерфейс администратора – список пользователей

Внешний вид формы редактирования информации о пользователя представлен на рис. 3.22. Администратор имеет возможность ввести фамилия, имя, отчество, имя пользователя и пароль для входа в систему. Так же можно выбрать подразделение, которое доступно для пользователя.

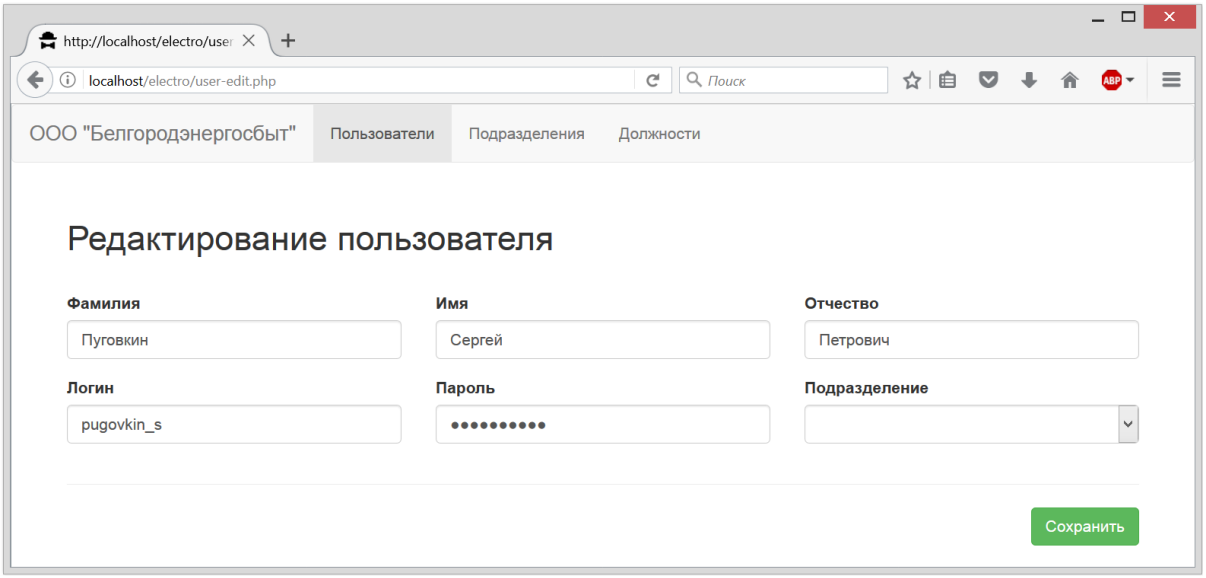

Рис. 3.22. Интерфейс администратора – форма редактирования информации о пользователе

На рис. 3.23 приведен внешний вид вкладки «Подразделения». В этом интерфейсе администратор системы имеет возможность просмотреть список подразделений, добавить новое подразделение, удалить и отредактировать существующие. Администраторам доступны все подразделения, а вот у каждого пользователя свой доступ к каждому участку компании. Если конкретный пользователь работает на Губкинском участке, то ему доступны данные только Губкинского участка.

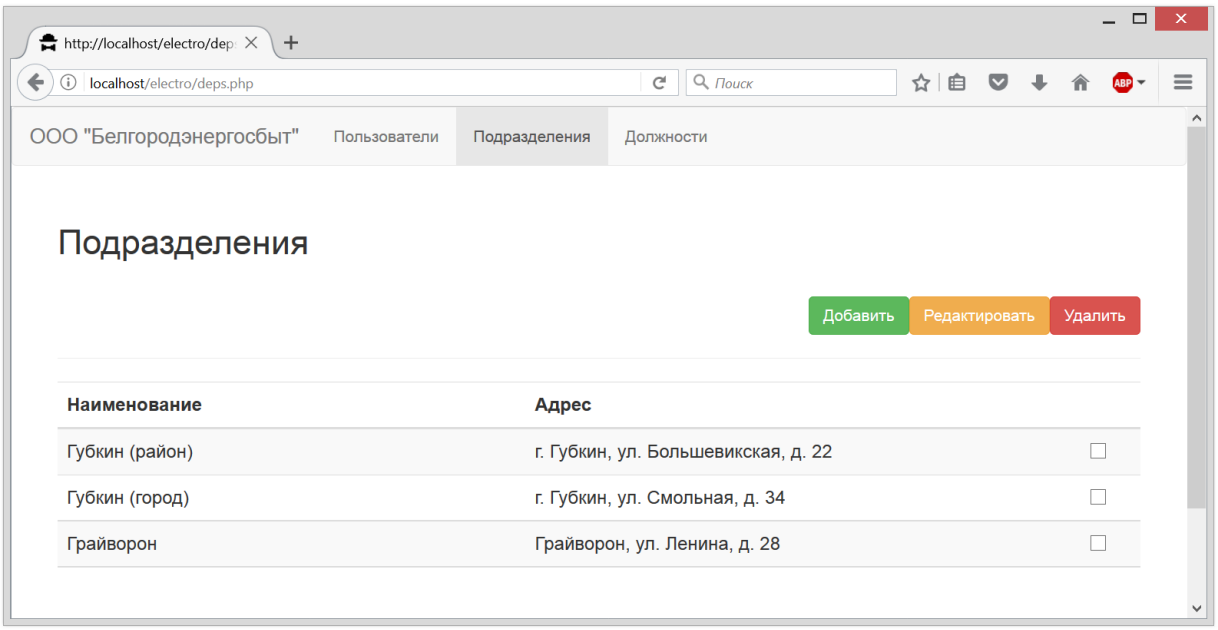

Рис. 3.23. Интерфейс администратора – список подразделений

На рис. 3.24 показан переход на форму редактирования. Для этого необходимо выбрать текущее подразделение с помощью компонента checkbox и нажать кнопку «Редактировать».

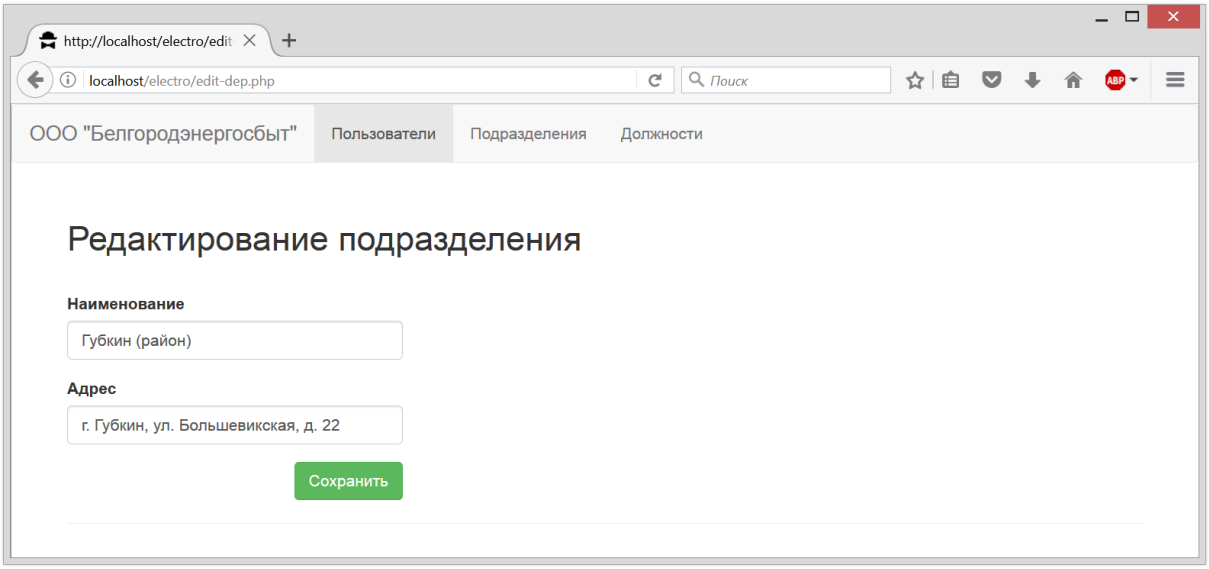

Рис. 3.24. Интерфейс администратора – форма редактирования подразделения

На рис. 3.25 приведен внешний вид вкладки «Должности». В этом интерфейсе администратор системы имеет возможность просмотреть список должностей, добавить новую должность, удалить и отредактировать существующие.

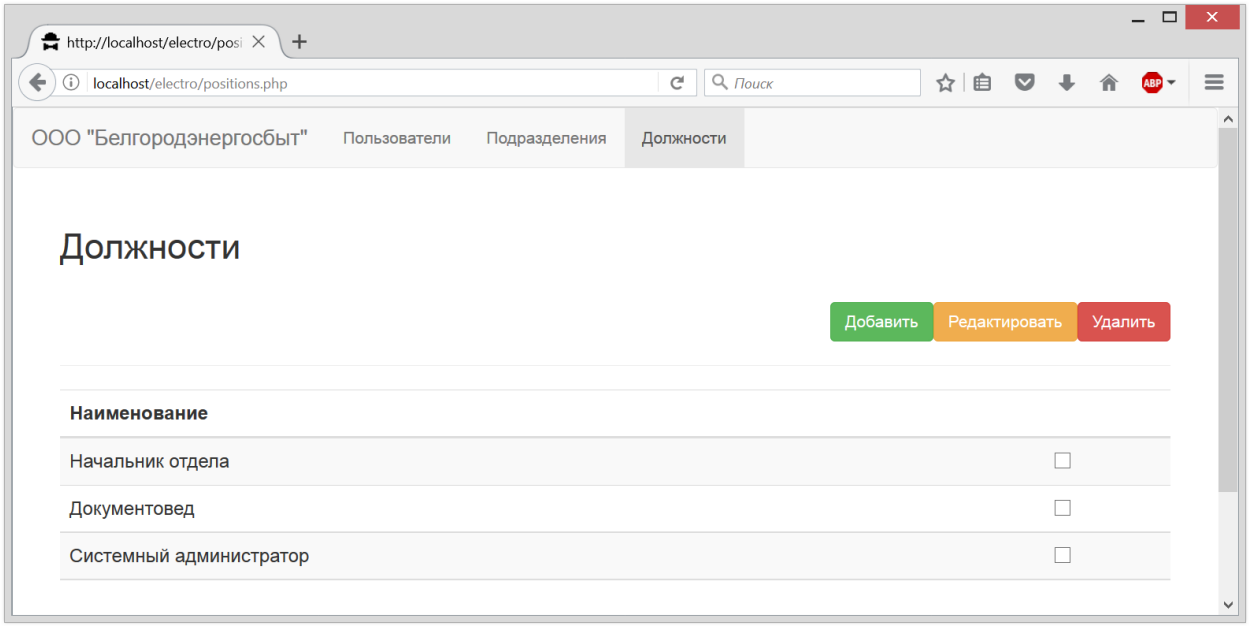

Рис. 3.25. Интерфейс администратора – список должностей

На рис. 3.26 показан переход на форму редактирования. Для этого необходимо выбрать текущую должность с помощью компонента checkbox и нажать кнопку «Редактировать».

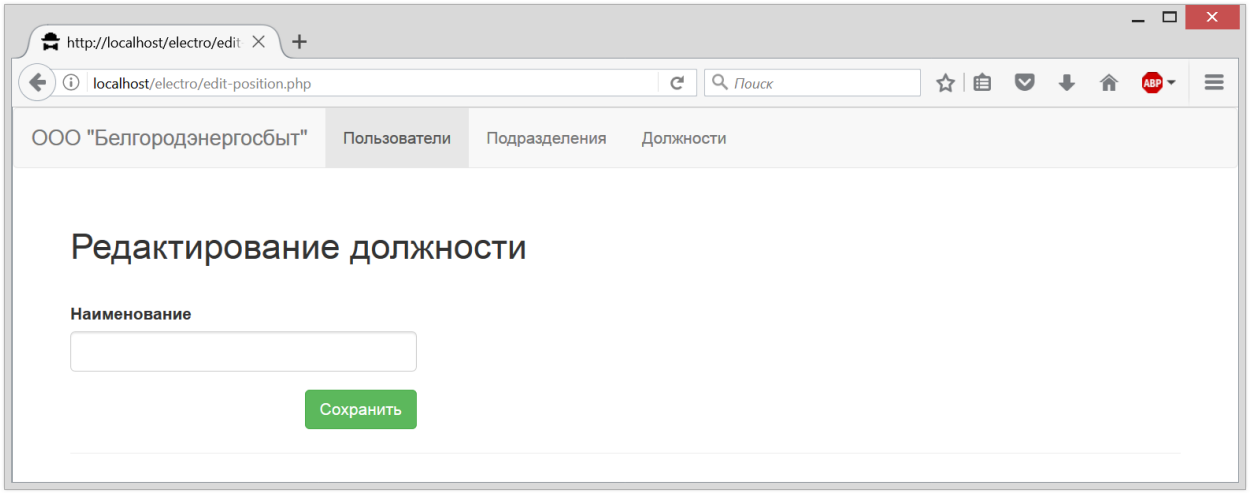

Рис. 3.26. Интерфейс администратора – форма редактирования должности

ЗАКЛЮЧЕНИЕ

В рамках данной работы были рассмотрены современные подходы к разработке автоматизированных систем прогнозирования расходов электроэнергии в виде веб-приложения.

В ходе выполнения данной работы были решены следующие задачи:

– произведен обзор и анализ современных математических методов построения прогнозов и выбрать наиболее подходящий метод для решения поставленной проблемы;

– сформулированы требования к автоматизированной системе прогнозирования расходов электроэнергии;

– произведено проектирование автоматизированной системы прогнозирования расходов электроэнергии;

– реализована автоматизированная система прогнозирования расходов электроэнергии в виде законченного программного продукта;

– произведено тестирование разработанного программного продукта и проверена точность и адекватность составляемых прогнозов.

В результате была реализована автоматизированных систем прогнозирования расходов электроэнергии с применением современных методик разработки веб-приложений.

Можно сделать вывод, что цель данной работы достигнута. В ходе выполнения данной выпускной квалификационной работы были систематизированы и углублены знания, полученные в процессе обучения. Кроме этого, был получен ценный опыт реализации реальных проектов.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Основы Data Science и Big Data. Python и наука о данных / Силен Дэви, Мейсман Арно, Али Мохамед. – Питер, 2017. – 336 с.

2. Э.Орама. Идеальная разработка ПО. Рецепты лучших программистов / Э.Орама, Г.Уилсон. – Питер, 2012. – 592 с.

3. Винстон Уэйн Л. Microsoft Excel 2013. Анализ данных и бизнес-моделирование / Уэйн Л Винстон. – БХВ-Петербург, 2015. – 864 с.

4. Марманис Х. Алгоритмы интеллектуального Интернета. Передовые методики сбора, анализа и обработки данных / Х. Марманис, Д. Бабенко. – Символ-Плюс, 2011. – 480 с.

5. Сегаран. Т. Программируем коллективный разум / Т. Сегаран. – Символ-Плюс, 2008. – 368 с.

6. Изучаем Spark. Молниеносный анализ данных / Х. Карау, Э. Конвински, П. Венделл, М. Захария. – ДМК Пресс, 2015. – 304 с.

7. Пассиг Катрин. Программирование без дураков / Катрин Пассиг, Йоханнес Яндер. – Питер, 2017. – 416 с.

8. Нейгард Майкл. Release it! Проектирование и дизайн ПО для тех, кому не все равно / Майкл Нейгард. – Питер, 2016. – 320 с.

9. Майкл С. Миковски. Разработка одностраничных веб-приложений / Майкл С. Миковски, Джош К. Пауэлл. – ДМК Пресс, 2014. – 512 с.

10. Прохоренок Н. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера / Н. Прохоренок, В. Дронов. – БХВ-Петербург, 2015. – 768 с.

11. Котеров Д. В. PHP 7 в подлиннике / Д. В. Котеров, И. В. Симдянов. – БХВ-Петербург, 2016. – 1088 с.

12. Дунаев В. В. Web-программирование для всех / В. В. Дунаев. – БХВ-Петербург, 2008. – 560 с.

13. Маклафлин Б. PHP и MySQL. Исчерпывающее руководство / Б. Маклафлин. – Питер, 2013. – 512 с.

14. Зольников Д.С. PHP 5. Как самостоятельно создать сайт любой сложности / Д.С. Зольников. – НТ Пресс, 2007. – 272 с.

15. Бокс Джордж. Анализ временных рядов. Прогноз и управление. Выпуск 2 / Джордж Бокс, Г. Дженкинс. – Мир, 1974.

16. Орлов Ю. Нестационарные временные ряды. Методы прогнозирования с примерами анализа финансовых и сырьевых рынков. Ю. Орлов, К. Осминин. – Либроком, 2011.

ПРИЛОЖЕНИЕ

Header.php

```
<head>
    <meta charset="utf-8">
    <title></title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">


    <style id="custom-css">#jumbo {
  background-color: #333;
  color: #eee;
}
#jumbo p {
  font-size: 16px;
}
#try-header {
  margin: 30px 0px;
}
#try-more {
  margin: 30px 0px;
  font-style: italic;
}</style>

    <script>
      parent.FrameWindow = window;
      parent.FrameDocument = document;
    </script>
    <script
src="//ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js"></script>

    <script id="custom-js"></script>
  <link type="text/css" rel="stylesheet"
href="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/css/bootstrap.min.css">
<link type="text/css" rel="stylesheet"
href="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/css/bootstrap-
glyphicons.css"><script
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js"></s
cript></head>
  <body class="design"><div id="jumbo" class="jumbotron">
```

Footer.php

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
style="display: none;">
</script>
<script
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js"
style="display: none;">
</script><div class="form-group"></div><div class="form-group">
</div>
</div>
</body>
```

DataGrid.php

```
<div class="container">
  <!-- Example row of columns -->
  <h2 id="try-header">Данные потребления электроэнергии
    <br></h2>
  <div class="row">
    <div class="col-md-3">
      <div class="form-group">
        <label>Подразделение:
        </label>
        <select class="form-control">
          <option value="Губкин">Губкин (район)
          </option>
          <option value="salad">Salad
          </option>
          <option value="pizzasalad">Pizza and Salad
          </option>
        </select>
      </div>
    </div>
    <div class="col-md-3">
      <div class="form-group">
        <label>Год
        </label>
        <select class="form-control">
          <option value="2014">2014
          </option>
          <option value="salad">Salad
          </option>
          <option value="pizzasalad">Pizza and Salad
          </option>
        </select>
      </div>
    </div>
    <div class="col-md-3">
      <div class="form-group">
        <label class="">Вид:
        </label>
        <select class="form-control">
          <option value="Таблица">Таблица
          </option>
          <option value="salad">Salad
          </option>
          <option value="pizzasalad">Pizza and Salad
          </option>
        </select>
      </div>
    </div>
    <div class="col-md-3">
      <button type="submit" class="btn pull-right btn-success">Печать
      </button>
    </div>
  </div>
  <disabledscript
src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
style="display: none;">
  </disabledscript>
  <disabledscript
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js"
style="display: none;">
  </disabledscript>
  <div class="form-group">
```

```
  </div>
  <div class="form-group">
  </div>
  <table class="table table-striped">
    <tbody>


    </tbody>
    <thead>
      <tr>
        <th>#
          <br>
        </th>
        <th>Месяц
        </th>
        <th>Год
          <br>
        </th>
        <th>
          Прогнозируемое значение, МВтч
        </th>
        <th>Реальное значение, МВтч
          <br>
        </th>
        <th>
          <br>
        </th>
      </tr>
    </thead>
  </table>
  <ul class="pagination">
    <li>
    <a href="#">&lt;</a>
    </li>
    <li>
    <a href="#">1</a>
    </li>
    <li>
    <a href="#">2</a>
    </li>
    <li>
    <a href="#">3</a>
    </li>
    <li>
    <a href="#">&gt;</a>
    </li>
  </ul>
</div>
```

DataGridGraph.php

```
  <nav class="navbar navbar-default" role="navigation">
    <div class="navbar-header">
      <a class="navbar-brand" href="#">ООО "Белгородэнергосбыт" </a>
    </div>
    <div class="collapse navbar-collapse navbar-ex1-collapse">
      <ul class="nav navbar-nav">
        <li class="active">
          <a href="#">Данные</a>
        </li>
        <li>
        <a href="#">Импорт данных</a>
```

```
      </li>
      <li>
      <a href="#">Построение прогноза</a>
      </li>
      <li>
      </li>
      <li>
      </li>
    </ul>
  </div>
</nav>
<div class="container">
  <!-- Example row of columns -->
  <h2 id="try-header">Данные потребления электроэнергии
    <br></h2>
  <div class="row">
    <div class="col-md-3">
      <div class="form-group">
        <label>Подразделение:
        </label>
        <select class="form-control">
          <option value="Губкин">Губкин (район)
          </option>
          <option value="salad">Salad
          </option>
          <option value="pizzasalad">Pizza and Salad
          </option>
        </select>
      </div>
    </div>
    <div class="col-md-3">
      <div class="form-group">
        <label>Год
        </label>
        <select class="form-control">
          <option value="2014">2014
          </option>
          <option value="salad">Salad
          </option>
          <option value="pizzasalad">Pizza and Salad
          </option>
        </select>
      </div>
    </div>
    <div class="col-md-3">
      <div class="form-group">
        <label class="">Вид:
        </label>
        <select class="form-control">
          <option value="Таблица">График
          </option>
          <option value="salad">Salad
          </option>
          <option value="pizzasalad">Pizza and Salad
          </option>
        </select>
      </div>
    </div>
    <div class="col-md-3">
    <br />
      <button type="submit" class="btn pull-right btn-success">Печать
      </button>
    </div>
```

```
    </div>
    <disabledscript
src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
style="display: none;">
    </disabledscript>
    <disabledscript
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js"
style="display: none;">
    </disabledscript>
    <div class="form-group">
    </div>
    <div class="form-group">
    </div>


     <script type="text/javascript"
src="https://www.gstatic.com/charts/loader.js"></script>
  <div id="chart_div"></div>
  <script type = "text/javascript">
          google.charts.load('current', {packages: ['corechart', 'line']});
google.charts.setOnLoadCallback(drawCurveTypes);

function drawCurveTypes() {
      var data = new google.visualization.DataTable();
      data.addColumn('number', 'X');
      data.addColumn('number', 'Прогноз');
      data.addColumn('number', 'Факт.');

      data.addRows([
        [1,  26725   ,  27301   ],
        [2,  25800   ,  24778   ],
        [3,  26062   ,  24331   ],
        [4,  24379   ,  23639   ],
        [5,  22212   ,  21745   ],
        [6,  22451   ,  19338   ],
        [7,  21923  ,  21738   ],
        [8,  21926  ,  22726   ],
        [9,  22109   ,  21459   ],
        [10, 25277   , 24295   ],
        [11, 24368   ,  24899   ],
        [12,  26797   ,  24899   ]
      ]);

      var options = {
        hAxis: {
          title: 'Месяц, 2014г.'
        },
        vAxis: {
          title: 'МВтч'
        },
        series: {
       0: {curveType: 'function'} ,
          1: {curveType: 'function'}
        }


      };

      var chart = new
google.visualization.LineChart(document.getElementById('chart_div'));
      chart.draw(data, options);
    }
  </script>
```

```
  </div>
```

DataImport.php

```
<div class="container">
    <!-- Example row of columns -->
    <h2 id="try-header">Импорт данных<br></h2>
    <div class="row">
        <div class="col-md-6">
            <div class="form-group"><label>Подразделение:</label><select
class="form-control">
                <option value="Губкин">Губкин</option>
                <option value="salad">Salad</option>
                <option value="pizzasalad">Pizza and Salad</option>
            </select></div>
            <div class="form-group"><label>Период:</label><select
class="form-control">
                <option value="1 год">1 год</option>
                <option value="salad">Salad</option>
                <option value="pizzasalad">Pizza and Salad</option>
            </select></div>
            <div class="form-group"><label>Файл данных:</label><input
class="form-control" type="file"></div>
            <div class="form-group"></div>
            <button type="submit" class="btn btn-success pull-
right">Импортировать</button>
        </div>
        <div class="col-md-6"></div>
    </div>
```

Prediction.php

```
<div class="container">
    <!-- Example row of columns -->
    <h2 id="try-header" class="">Построение прогноза<br></h2>
    <div class="row">
        <div class="col-md-6">
            <div class="form-group"><label>Подразделение:</label><select
class="form-control">
                <option value="Губкин">Губкин</option>
                <option value="salad">Salad</option>
                <option value="pizzasalad">Pizza and Salad</option>
            </select></div>
            <div class="form-group"><label class="">Период:</label><select
class="form-control">
                <option value="1 год">1 год</option>
                <option value="salad">Salad</option>
                <option value="pizzasalad">Pizza and Salad</option>
            </select></div>
            <div class="form-group"></div>
            <button type="submit" class="btn btn-success pull-
right">Построить прогноз</button>
        </div>
        <div class="col-md-6"></div>
    </div>
```

AuthControlle.php

```
<?php
error_reporting(E_ALL);

class auth extends CI_Controller
{

    public function __construct()
    {
        parent::__construct();
        $this->load->model('auth_model');
        $this->load->library('cas');
        $this->load->helper('url');
    }

    public function index()
    {
        $data = array();
        $data['user'] = $this->session->userdata('user');
        $errorCode = $this->input->get('error');
        if ($errorCode == 1)
            $data['error'] = 'Вы не имеете доступа к системе, обратитесь к
администратору';
        else
            $data['error'] = 'Имя пользователя/пароль неверно';
        $this->load->view('header', $data);
        $this->load->view('authfailed', $data);
        $this->load->view('footer');
    }

    function login()
    {

        if (($this->input->post('login') == '') || $this->input-
>post('password') == '')
            header('location:/index.php/auth');

        $user = $this->auth_model->is_user($this->input->post('login'),
md5($this->input->post('password')));

        if ($user) {
            $session_data = array('logon' => 'Yes!', 'user' => $user); //
записываем в сессию признак логона
            $this->session->set_userdata($session_data);
            header('location:/index.php/panel');
        } else {
            header('location:/index.php/auth');
        }
    }

    function logout()
    {

        $this->session->sess_destroy();
        $userCAS = $this->cas->user();

        if ($userCAS) {
            $url = base_url().'index.php/panel';
            $this->cas->logout($url);
        } else {
            header('location:/index.php');
```

```
        }
    }

    function caslogin() {
        $this->cas->force_auth();
        $userCAS = $this->cas->user();
        if ($userCAS) {
            $userDB = $this->auth_model->getUserInfo($userCAS->userlogin);


            if ($userDB) {
                $session_data = array('logon' => 'Yes!', 'user' => $userDB);
// записываем в сессию признак логона
                $this->session->set_userdata($session_data);
                header('location:/index.php/panel');
            } else {
                $url = base_url().'index.php/auth/index/?error=1';
                $this->cas->logout($url);
            }

        }

    }

}
<?php

include_once('protectController.php');


/**
 * Class panel
 * @property UserListModel $userListModel
 * @property TemplateModel $templateModel
 * @property UnitModel $unitModel
 */
class templates extends protectController
{

    function __construct()
    {
        parent::__construct();
        $this->load->model('userListModel');
        $this->load->model('unitModel');
        $this->load->model('templateModel');
    }

    public function index()
    {

        $user = $this->session->userdata('user');
        //TODO добавить отлуп для неадминов
        $data['user'] = $user;

        /** @var \HistoryBsu\Template\DTO\TemplateData[] $templates */
        $templates = $this->templateModel->getList();
        $data['templates'] = $templates;

        $this->printView('templates', $data);
    }

    public function addNewTemplate()
    {
```

```
        $path_parts = pathinfo($_FILES['file']['name']);
        $ext = $path_parts['extension'];

        $templateDir = 'files/templates'; //TODO extract in config-file

        $uniqueId = uniqid('file', false);
        $fileName = sprintf('%s/%s.%s', $templateDir, $uniqueId, $ext);
        if (!move_uploaded_file($_FILES['file']['tmp_name'], $fileName)) {
            throw new RuntimeException('Cant move upload file');
        }

        $this->templateModel->addNewTemplate($fileName);
        $this->redirect('templates');
    }

    public function getTemplate($id = null)
    {
        //TODO отдавать по ID ТОЛЬКО админу
        $id = $id ? (int)$id : null;
        $template = $this->templateModel->getTemplate($id);


        $content = file_get_contents($template->fileLink);

        $ext = pathinfo($template->fileLink);
        $ext = $ext['extension'];

        header('Content-Description: File Transfer');
        header(sprintf('Content-Disposition: attachment;
filename=\'template.%s\'', $ext));
        header('Content-Transfer-Encoding: BINARY');
        header('Expires: 0');
        header('Cache-Control: must-revalidate');
        header('Pragma: public');
        header('Content-Length: ' . strlen($content));
        header('Content-Type: application/octet-stream');
        echo($content);
        exit();
    }

}
```

AuthModel.php

```
<?php

class auth_model extends CI_Model {

  public function __construct()
  {
      parent::__construct();
      // Your own constructor code
  }

   function is_user($login, $password) {
        $q = "SELECT *".
           " FROM `user` ".
            " INNER JOIN `access` ON `user`.`IdUser` = `access`.`IdUser` ".
            " INNER JOIN `role` ON `access`.`IdRole` = `role`.`IdRole` ".
           " WHERE `login` = ".$this->db->escape($login)." AND ".
           " `password` = ".$this->db->escape($password)."";
```

```
        $r = $this->db->query($q);
        if ($r->num_rows() > 0)
          {
                  return $r->row();
        }
      return 0;
   }

    function getUserInfo($login) {
        $q = "SELECT *".
            " FROM `user` ".
            " INNER JOIN `access` ON `user`.`IdUser` = `access`.`IdUser` ".
            " INNER JOIN `role` ON `access`.`IdRole` = `role`.`IdRole` ".
            " WHERE `login` = ".$this->db->escape($login).";";

        $r = $this->db->query($q);
        if ($r->num_rows() > 0)
        {
            return $r->row();
        }
        return false;

    }
}
<?php

use \HistoryBsu\Template\DTO\TemplateData;

/**
 * Class unitModel
 * @property CI_DB_query_builder $db
 */
class templateModel extends CI_Model
{

    private $table = 'template';

    public function __construct()
    {
        parent::__construct();
    }

    /**
     * @return TemplateData[]
     */
    public function getList()
    {
        $this->db->select('*');
        $this->db->from($this->table);
        $query = $this->db->get();

        $result = array();
        if ($query->num_rows() > 0) {
            $rawResult = $query->result();
            foreach ($rawResult as $rawObject) {
                $result[] = $this->createDtoFromRaw($rawObject);
            }

        }

        return $result;
    }
```

```
    /**
     * @param int|null $id Id шаблона, если не определен, то отдаст
актуальный
     * @return TemplateData
     */
    public function getTemplate($id = null)
    {
        $this->db->select('*');
        $this->db->from($this->table);
        if ($id) {
            $this->db->where('IdTemplate', $id);
        } else {
            $this->db->limit(1);
            $this->db->order_by('AddDate', 'desc');
        }
        $query = $this->db->get();

        if ($query->num_rows() != 1) {
            throw new RuntimeException(); //todo add error message
        }
        $queryResult = $query->result();
        $rawObj = reset($queryResult);
        $obj = $this->createDtoFromRaw($rawObj);

        return $obj;
    }

    public function addNewTemplate($fileName)
    {
        $data = array('FileLink' => $fileName);
        $this->db->insert($this->table, $data);
    }

    /**
     * @param object $rawObj
     * @return TemplateData
     */
    private function createDtoFromRaw($rawObj)
    {
        $obj = new TemplateData();
        $obj->id = $rawObj->IdTemplate;
        $obj->fileLink = $rawObj->FileLink;
        $dateTime = new DateTime($rawObj->AddDate);
        $obj->dateOfUpload = $dateTime->format('d.m.Y H:i');
        return $obj;
    }
}
```