МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Отделение информационных технологий и вычислительной техники

Курсовая работа по дисциплине «Технология разработки ПО»

тема: Разработка информационной системы "Больница"

Выполнил: Леонов А.О.
Проверил: Сафаров Р.А.

Новосибирск, 2021 г.

Оглавление

ВВЕДЕНИЕ

В современном обществе каждое предприятие сталкивается с такими трудностями, как обработка, поиск и хранение необходимой информации. С целью оптимизации данных процессов – они автоматизируются, т.е. создаются информационные системы. Информационные системы дают возможность хранить большие объемы данных, экономить время и выдавать нужную информацию в кратчайшие сроки в удобном для пользователя виде.

Больница – это такая организация, которая работает с очень большим объемом информации, как о сотрудниках, так и о пациентах. Врачам нужно всегда следить ща данными о своих пациентах, о курсе лечения больных. А руководству и бухгалтерии необходимо быть в курсе событий о своих сотрудниках. Для этого нужна общая база данных, включающая всю необходимую информацию. Программа является очень актуальной на сегодняшний день, она автоматизирует работу с базой данных и предоставляет пользователю понятный и дружественный интерфейс.

Мощность базы данных обусловлена возможностью ее постоянного пополнения новыми данными, причем в неограниченном количестве информации. Это является очень удобным для пользователя. Таким образом, создание базы данных, обладающей такими свойствами, задача достаточно актуальная и полезная.

Целью данной работы является создание базы данных больницы. Она предназначена для хранения информации о врачах, пациентах клиники, вспомогательной информации о распорядке работы больницы и приеме больных, с возможностью внесения данных, выборки и изменения данных, вывода информации в необходимом формате.

ГЛАВА 1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. История развития баз данных

Историю развития баз данных можно разделить на четыре периода.

1. Период становления – начало 60-х - начало 70-х гг. В этот период появляется сам термин «база данных» и создается несколько первоначальных систем. Основой появления баз данных явилось предложение конца 50-х годов использовать файлы для хранения исходных данных. Основное требование к таким файловым системам – быть совместно используемым хранилищем данных. В последующем стало очевидным, что совместно используемые данные, должны обладать специфическими свойствами, в частности: независимость данных, отсутствие дублирования и противоречивости, контроль прав доступа к данным, эффективная техника доступа к данным, а также многие другие.

Осознание этих фактов, а также появление больших компьютеров с магнитными дисками в качестве носителей данных привело к появлению в середине 60-х гг. первых систем управления базами данных, из которых наиболее развитой оказалась система IMS фирмы IMB, которая поддерживала иерархическую структуру данных. Бахман в 1963 г. разработал первую промышленную систему баз данных IDS. Система IDS поддерживала сетевую организацию данных на магнитных носителях.

Ассоциация CODASYL, являющаяся органом, разработавшим язык программирования Кобол, в 1967 г. организовала рабочую группу по базам данных. Эта группа обобщила языковые спецификации систем баз данных и в 1969 и 1971 гг. издала соответствующие отчеты, которые по наименованию рабочей группы (DataBaseTaskGroup) получили название DBTG69, DBTG71. Основой избранного Рабочей группой подхода послужила сетевая структура данных и способы навигации по ней, разработанные в системе IDS, однако

сетевая модель данных в отчетах DBTG получила существенное развитие и обоснование.

Типичным представителем системы, поддерживающей предложения DBTGCODASYL явилась Integrated Database Management System (IDMS) компании Cullinet Software Inc., предназначенная для использования на машинах основного класса фирмы IBM под управлением большинства операционных систем.

В этот же период четко выкристаллизовались два подхода относительно проблемы замкнутости систем баз данных. Системы замкнутого типа характеризуются тем, что они не содержат в своем составе традиционных языков программирования, а имеют непроцедурные языки запросов. Основной целью в данном случае является создание системы, с применением которой мог бы справиться не специалист по программированию. К таким системам относились TDMS и UL/1.

Системы с включаемыми языками помимо собственно языков манипулирования базами данных предоставляют также языковые и инструментальные средства разработки приложений, с использованием существующих языков программирования. Этот принцип, в частности, исповедовался DBTG.

В конце данного периода появился термин *информационно-управляющая система* (MIS). В то время под MIS понималась система баз данных, ориентированная на поиск данных и обеспечивающая возможность работы с удаленного терминала.

Период развития – 70-е годы. Концепция баз данных широко распространяется благодаря повышению характеристик аппаратного обеспечения компьютеров. Идет успешное внедрение систем, поддерживающих иерархическую и сетевую структуры данных.

Весь этот период продолжалась работа DBTG CODASYL. Была специфицирована система языков для баз данных CODASYL, которая включила следующие группы языковых спецификаций:

Язык описания данных ЯОД (Data Definition Language - DDL). Представляет собой описание концептуальной схемы в терминах сетевой структуры данных.

Средства базы данных языка Кобол. Представляет собой средства обеспечения интерфейса языка Кобол с базой данных, описанной в DDL. В Кобол включены средства языка манипулирования данными.

Средства базы данных языка Фортран. Представляет собой средства обеспечения интерфейса языка Фортран с базой данных, описанной в DDL. В Кобол включены средства языка манипулирования данными.

Средства конечного пользователя. Определяет интерфейс пользователя случае, когда такой пользователь управляет базой данных, описанной в DDL.

Язык описания хранения данных ЯОХД (Data Stored Definition Language - DSDL). Представляет собой язык, который отображает концептуальную схему, описанную в DDL, во внутреннюю схему.

В 1975 г. появился отчет рабочей группы ANSI/X3/SPARC Американского Национального Института Стандартов, который явился значительной вехой в развитии проблематики баз данных. Перед группой была поставлена задача исследовать, в какой мере целесообразно ставить вопрос о стандартизации баз данных СУБД и что именно может быть подвержено стандартизации. Группа пришла к выводу, что если ставить вопрос о стандартизации, то только относительно интерфейсов, которые могут существовать между различными компонентами СУБД, сами программные компоненты ни в коем случае подвергаться стандартизации не могут. В

связи с этим они направили свои последующие усилия на выявление таких интерфейсов и, в конце концов, пришли к формулировке трехуровневой архитектуре баз данных, которая стала классической и не потеряла свою актуальность до сих пор.

Однако этот период в большей мере характеризуется появлением реляционной модели данных, предложенной в 1970 г. сотрудником института фирмы ИБМ в Сан-Хосе Э.Ф. Коддом, всесторонними исследованиями теоретических и прикладных вопросов этой модели, разработкой экспериментальных реляционных СУБД. Теоретические исследования привели, в конце концов, к созданию формальной теории баз данных, которая до этого носила описательный характер. На протяжении многих лет многие ведущие формы проводили экспериментальные исследования по созданию прототипов реляционных СУБД, повышению их эффективности и функциональности. В конце 70-х гг. появляются первые промышленные реляционные СУБД.

Период зрелости – 80-е годы. Реляционная модель получила полное теоретическое обоснование. Разработаны крупные реляционные СУБД Oracle, Informix, и другие. Промышленные реляционные системы получают широкое распространение во всех сферах человеческой деятельности. Реляционные системы практически вытеснили с мирового рынка ранние СУБД иерархического и сетевого типа.

Дальнейшее развитие реляционных СУБД шло в следующих направлениях:

Удобство применения. Появление персональных компьютеров сделал принципиальным вопрос удобства использования программ, что также относилось и к СУБД. На протяжении всего этого периода интенсивно развивается внешний интерфейс взаимодействия пользователей с базами данных.

Многоплановость. Изначально базы данных разрабатывались для хранения и обработки символьной информации и традиционно использовались в таких сферах, как обработка экономической информации, статистика, банковское дело, системы резервирования, информационные системы различного направления. Появление спроса к базам данных в нетрадиционных сферах их применения, системы автоматизации проектирования, издательское дело и другие, потребовали хранения в базах данных и обработки изображений, звуков, полнотекстовой информации.

Этот период также характеризуется теоретическими и экспериментальными исследованиями в области баз знаний. Разрабатываются многочисленные экспертные системы, использующие базы знаний. В подавляющем большинстве случаев базы знаний разрабатываются на основе реляционных СУБД.

Пост реляционный период – с начало 90-х гг. В этот период начались проводиться интенсивные исследования по дедуктивным и объектно-ориентированным базам данных, а также разработка исследовательских прототипов таких систем.

Особое место в развитии проблематики объектно-ориентированных СУБД занимает деятельность группы по управлению объектными базами данных ODMG (Object Data Management Group), - неприбыльным консорциумом производителей объектных баз данных и других организаций, заинтересованных в выработке стандартов по хранению объектов в базах данных. ODMG была создана в 1991 г. В 1993 г. группа выпустила свой первый стандарт – ODMG-93. В 1995 г. был опубликован усовершенствованный вариант этого стандарта.

В связи с развитием Интернет-технологий прикладываются большие усилия по внедрению баз данных в Интернет. Возникают различные подходы по включению СУБД с их базами данных во всемирную паутину, начиная от

простейших «публикаций» баз данных в Интернет и заканчивая разработкой web-серверов баз данных, которые в состоянии предоставлять весь спектр услуг пользователям Интернета по использованию баз данных на сервере.

Наконец, интенсивно развиваются исследования и разработки по представлению и манипулированию структурами данных в Интернет.

1.2. Техническое задание

Цель работы: создание автоматизированной информационной системы для поликлиники.

Объект исследования: информационная система поликлиники.

Предмет исследования: использование программных продуктов для создания информационной системы поликлиники.

1.2.1 Информация о целевой аудитории, цели и задачи сайта

Целевой аудиторией являются лечащий персонал, сотрудники регистратуры и пациенты.

Цель сайта – обеспечить пользователей простым и понятным интерфейсом. А также облегчить работу лечащему персоналу.

1.2.2. Технические требования к работе информационной системы

Основные возможности информационной системы:

- хранение и выдача информации о пользователях в системе
- регистрация пользователей
- просмотр информации о врачах и больных
- внесение номеров телефона в базу
- добавление и обновление новостей медицины

Веб-сайт должен иметь простой понятный интерфейс доступный для просмотра на любом устройстве.

1.2.3. Список требований к хостингу

В качестве веб-хостинга была выбрана облачная платформа НГТУ: CLOUD.NSTU.RU.

Основная задача облачной платформы – обеспечить аудиторные и внеаудиторные занятия виртуальными лабораториями для работы со специализированным программным обеспечением.

Доступные ресурсы:

- Виртуальный сервер на Windows
- Процессоров 4vCPU
- Оперативная память 2ГБ
- Хранилище 100ГБ

1.3. Инструментальные средства разработки

1.3.1. Visual Studio Code

Visual Studio Code – это редактор исходного кода. Он имеет многоязычный интерфейс пользователя и поддерживает ряд языков программирования, подсветку синтаксиса, IntelliSense, рефакторинг, отладку, навигацию по коду, поддержку Git и другие возможности. Многие возможности Visual Studio Code недоступны через графический интерфейс, зачастую они используются через палитру команд или JSON-файлы. Палитра команд представляет собой подобие командной строки, которая вызывается сочетанием клавиш.

VS Code также позволяет заменять кодовую страницу при сохранении документа, символы перевода строки и язык программирования текущего документа.

С 2018 года появилось расширение Python для Visual Studio Code с открытым исходным кодом. Оно предоставляет разработчикам широкие возможности для редактирования, отладки и тестирования кода.

Также VS Code поддерживает редактирование и выполнение файлов типа «Блокнот Jupyter» напрямую «из коробки» без установки внешнего модуля в режиме визуального редактирования и в режиме редактирования исходного кода.

На март 2019 года посредством встроенного в продукт пользовательского интерфейса можно загрузить и установить несколько тысяч расширений только в категории «programming languages» (языки программирования).

Также расширения позволяют получить более удобный доступ к программам, таким как Docker, Git и другие. В расширениях можно найти линтеры кода, темы для редактора и поддержку синтаксиса отдельных языков.

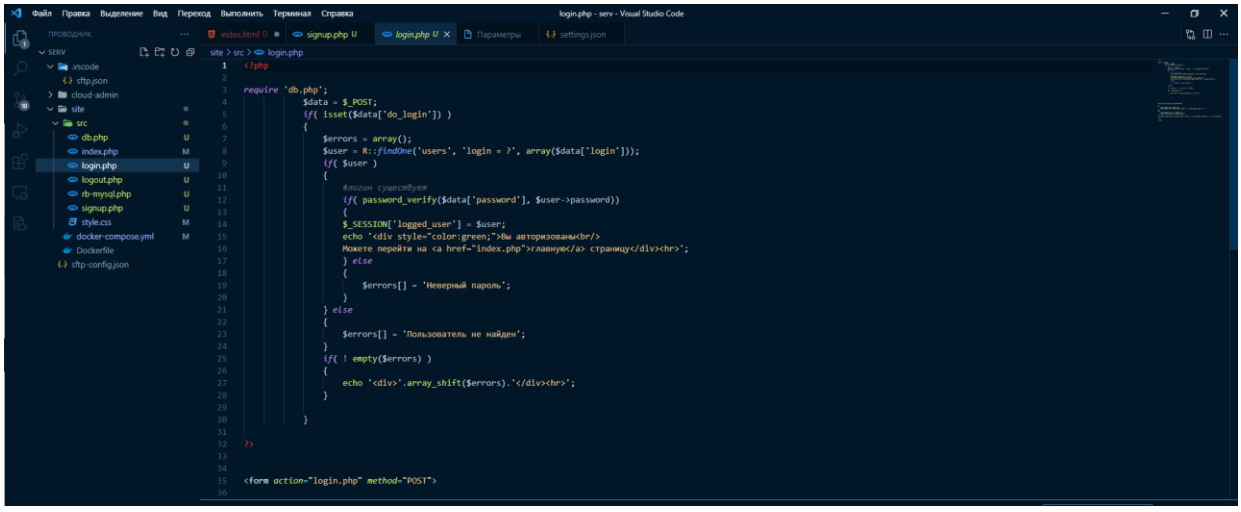

Рис. 1. Visual Studio Code

1.3.2. Sublime Text

Sublime Text является бесплатным редактором текстовых файлов и поддерживает синтаксис огромного количества языков программирования (более пятидесяти) в среде Windows.

Программа потребляет достаточно мало системных ресурсов и, соответственно, совсем не требовательна к железу компьютера, при этом она имеет огромные возможности - как для текстового редактора, и неплохую скорость работы.

Sublime Text сделан для того, чтобы облегчить работу программисту (причем достаточно широкого профиля), предлагая универсальный редактор кода.

Sublime Text поддерживает большое количество языков программирования и имеет возможность подсветки синтаксиса для C, C++, C#, CSS, D, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, MATLAB, Ocaml, Perl, PHP, Python, Rust, SQL и XML.

В дополнение к тем языкам программирования, которые включены по умолчанию, пользователи имеют возможность загружать плагины для поддержки других языков.

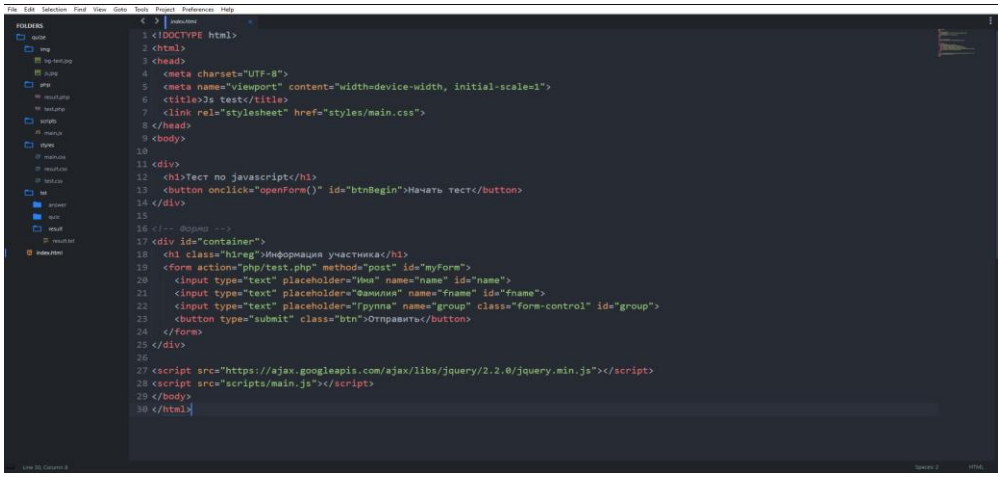

Рис. 2. Sublime Text

1.3.3. Notepad++

Notepad++ - свободный текстовый редактор с открытым исходным кодом для Windows с подсветкой синтаксиса, разметки, а также языков описания аппаратуры VHDL и Verilog.

Базируется на компоненте Scintilla, написан на C++ с использованием STL, а также Windows API, и распространяется под лицензией GNU General Public License. Базовая функциональность программы может быть расширена за счёт плагинов. Поддерживает открытие более 100 форматов.

Возможности:

- Подсветка синтаксиса
- Сворачивание кода
- Поддержка и конвертирование кодировок ANSI, UTF-8 и UCS-2
- FTP-менеджер (плагин NppFTP)
- Автосохранение

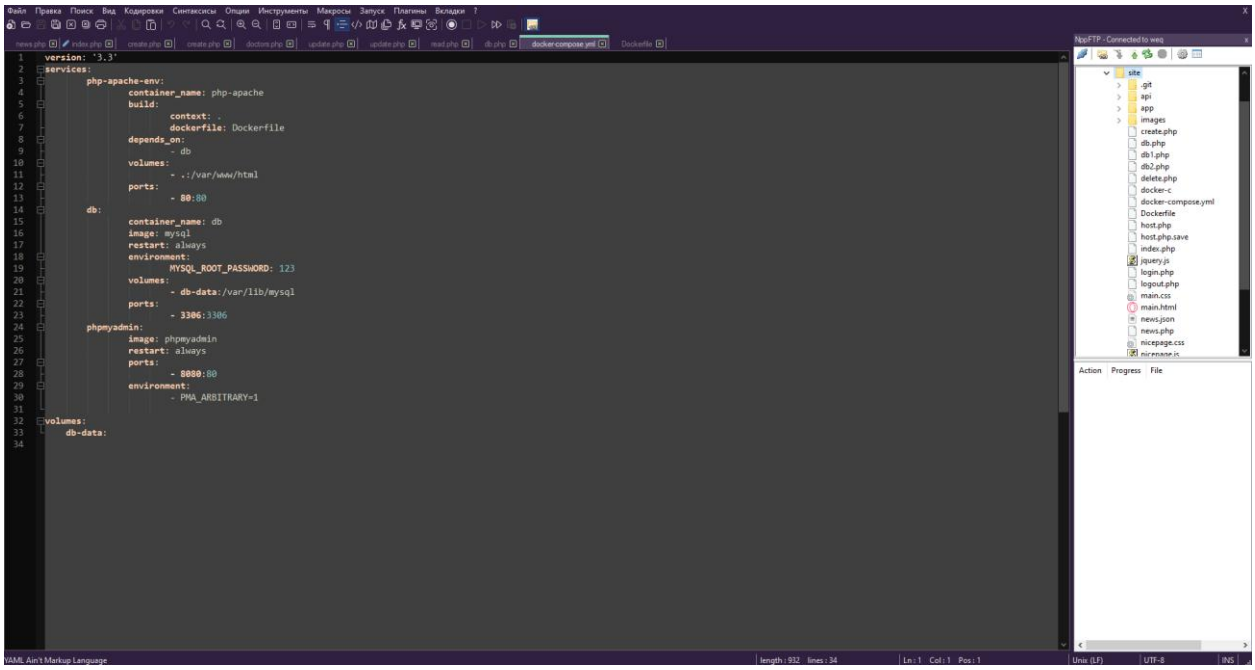

Рис. 3. Notepad++

1.3.4. PhpMyAdmin

PhpMyAdmin – веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL. PhpMyAdmin позволяет через браузер осуществлять администрирование сервера MySQL, запускать команды SQL и просматривать содержимое таблиц и баз данных. Приложение пользуется большой популярностью у веб-разработчиков, так как позволяет управлять СУБД MySQL без непосредственного ввода SQL команд.

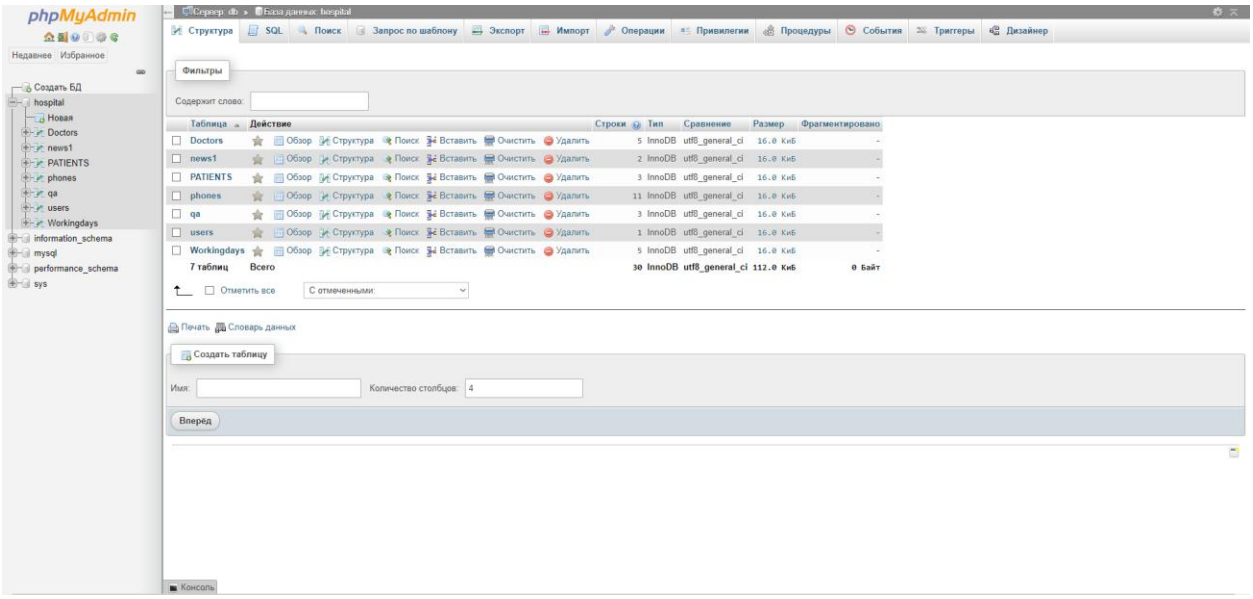

Рис. 4. PhpMyAdmin

1.3.5. Docker

Docker – программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой и управления приложениями в средах с поддержкой контейнеризации, контейнеризатор приложений. Позволяет «упаковать» приложение со всем его окружением и зависимости в контейнер, который может быть развернут на любой Linux-системе с поддержкой cgroups в ядре, а также предоставляет набор команд для управления этими контейнерами.

ГЛАВА 2. РАЗРАБОТКА СИСТЕМЫ

2.1. Структура проекта

Проект состоит из четырёх основных скриптов: чтение, запись, удаление и обновление. Также был создан интерфейс сайта, к которому подключены эти скрипты.

Скрипт «Чтение» отвечает за вывод данных с БД на сайт. Его задача: считать данные с БД «News» и вывести эти данные на сайт в поле «Новости».

```php
include_once "db2.php";
include_once "news.php";

$database = new Database();

$db = $database->getConnection();

$news = new News($db);

$stmt =$news->read();
$num = $stmt->rowCount();

if ($num > 0) {
    $News_arr = array();
    $News_arr['records'] = array();

    while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
        extract($row);
        $news_item = array(
            "NEWS" => $NEWS,
        );

        array_push($News_arr['records'], $news_item);
    }

    http_response_code(200);

    $json_data = json_encode($News_arr);
    file_put_contents('news.json', $json_data);
}
else {
    http_response_code(404);

    echo json_encode(array("message" => "Записи не найдены"), JSON_UNESCAPED_UNICODE);
}
```

Рис. 5. Чтение записей

Скрипт «Запись» отвечает за внесение данных с сайта в БД. Его задача: записать данные, введённые на сайте в специальном поле в БД «News».

```php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
header("Access-Control-Allow-Methods: POST");

include_once "db2.php";
include_once "news.php";

$database = new Database();
$db = $database->getConnection();


$news = new News($db);

$data = json_decode(file_get_contents("php://input"));

if (!empty($data->NEWS)) {
    $news->NEWS = $data->NEWS;


    if ($news->create()) {
        http_response_code(201);

        echo json_encode(array("message" => "Запись была создана"), JSON_UNESCAPED_UNICODE);
    }
    else {
        http_response_code(503);

        echo json_encode(array("message" => "Невозможно создать запись"), JSON_UNESCAPED_UNICODE);
    }
}
else {
    http_response_code(400);

    echo json_encode(array("message" => "Невозможно создать запись. Данные неполные"), JSON_UNESCAPED_UNICODE);
}
```

Рис. 6. Создание записи

Скрипт «Удаление» отвечает за удаление данных с сайта и БД. Его задача: удалить данные одной из строк, внесённых в БД «News».

```php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
header("Access-Control-Allow-Methods: DELETE");
header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

include_once "db2.php";
include_once "news.php";

$database = new Database();
$db = $database->getConnection();

$news = new News($db);

$data = json_decode(file_get_contents("php://input"));

$news->id = $data->id;

if ($news->delete()){

    http_response_code(200);

    echo json_encode(array("message" => "Запись была удалёна."), JSON_UNESCAPED_UNICODE);
}
else {

    http_response_code(503);

    echo json_encode(array("message" => "Не удалось удалить запись"), JSON_UNESCAPED_UNICODE);
}
```

Рис. 7. Удаление записи

Скрипт «Обновление» отвечает за обновление данных, как на сайте, так и в БД. Его задача обновить данные одной из строк в БД «News».

```php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
header("Access-Control-Allow-Methods: PUT");

include_once "db2.php";
include_once "news.php";

$database = new Database();
$db = $database->getConnection();


$news = new News($db);

$data = json_decode(file_get_contents("php://input"));

//$news->id = $data->id;

$news->NEWS = $data->NEWS;

if ($news->update()){

    http_response_code(200);

    echo json_encode(array("message" => "Заспись была изменена"), JSON_UNESCAPED_UNICODE);
}
else {

    http_response_code(503);

    echo json_encode(array("message" => "Невозможно обновить запись"), JSON_UNESCAPED_UNICODE);
}
```

Рис. 8. Обновление записи

2.2. Структура базы данных

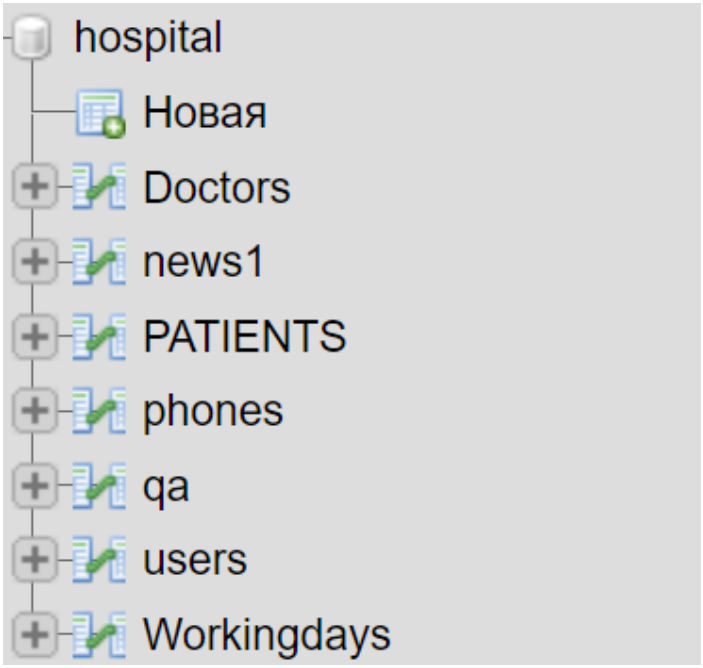

Рис. 9. Структура базы данных

БД состоит из четырех таблиц, представленных на рисунке 6.

Таблица Doctors содержит данные о докторах. Чтобы доктор был добавлен в таблицу, он должен пройти регистрацию, по специальной ссылке. Структура представлена в таблице 1.

Таблица 1

Лечащие врачи

Название	**Тип**	**Комментарий**
id	integer	Уникальный идентификатор доктора
POSITION	varchar(20)	Должность доктора
NAME	varchar(20)	Имя доктора
SURNAME	varchar(20)	Фамилия доктора
MIDDLENAME	varchar(20)	Отчество доктора
ROOMNUM	integer	Номер кабинета

В таблице news1 находятся новости медицины. Структура представлена в таблице 2.

Таблица 2

Новости медицины

Название	**Тип**	**Комментарий**
id	integer	Уникальный идентификатор новости
NEWS	text	Новости медицины

Таблица PATIENTS содержит в себе информацию о пациентах и их заболеваниях. Чтобы пациент был добавлен в таблицу, он должен записаться на прием к любому врачу. Так же лечащий врач может поставить диагноз, тем самым обновить данные о пациенте в таблице. Структура представлена в таблице 3.

Таблица 3

Пациенты

Название	**Тип**	**Комментарий**
id	integer	Уникальный идентификатор пациента
NAME	varchar(20)	Имя пациента
SURNAME	varchar(20)	Фамилия пациента
MIDDLENAME	varchar(20)	Отчество пациента
BIRTHDAY	date	Дата рождения пациента
DISEASES	text	Информация о заболеваниях (карточка) пациента

Таблица phones содержит в себе номера телефонов, оставленные пациентами, записавшихся на платный прием. Чтобы данные появились в таблице, пользователю надо нажать на кнопку «Записаться на платный прием», которая находится в шапке сайта. Далее в появившемся окне вписать

номер телефона и нажать кнопку «Отправить номер». Структура представлена в таблице 4.

Таблица 4

Номера телефонов

Название	Тип	Комментарий
id	integer	Уникальный идентификатор номера телефона
number	varchar(12)	Номер телефона оставленный пациентом

В таблице qa находятся часто задаваемые вопросы. Структура представлена в таблице 5.

Таблица 5

Вопросы и ответы

Название	Тип	Комментарий
id	integer	Уникальный идентификатор вопросов
QA	text	Вопрос и ответ на него

В таблице users находится информация о зарегистрированных пользователях. А точнее их: E-mail, логин и захешированный пароль. Структура представлена в таблице 6.

Таблица 6

Пользователи

Название	Тип	Комментарий
id	integer	Уникальный идентификатор пользователя
login	varchar(191)	Логин пользователя
email	varchar(191)	E-mail пользователя
password	varchar(191)	Пароль пользователя

Таблица Workingdays содержит в себе информацию о графике работы каждого доктора. Вносятся изменения в эту таблицу администратором. Структура представлена в таблице 7.

Таблица 7

Рабочие дни

Название	Тип	Комментарий
IDDOCTOR	integer	Уникальный идентификатор доктора
MONDAY	time	Рабочее время в понедельник
TUESDAY	time	Рабочее время во вторник
WEDNESDAY	time	Рабочее время в среду
THURSDAY	time	Рабочее время в четверг
FRIDAY	time	Рабочее время в пятницу
SATURDAY	time	Рабочее время в субботу
SUNDAY	time	Рабочее время в воскресенье

2.3 Развертывание системы

Система развернута на виртуальном сервере в облачной платформе НГТУ. В первую очередь необходимо установить Docker. Процесс установки запускается командой apt-get install docker.io. Далее нужно открыть порты для MYSQL. Выполняется это следующими командами: *docker pull mysql* и *docker run --name my_mysql -e MYSQL_ROOT_PASSWORD=123 -d -p 3306:3306 mysql*. Далее нужно создать папку для сайта и создать в ней два файла: Dockerfile (рис. 7.) и docker-compose.yml (рис. 8.), после чего можно начать разрабатывать ИС.

```
1  FROM php:8.0-apache
2  RUN docker-php-ext-install mysqli pdo pdo_mysql && docker-php-ext-enable mysqli pdo pdo_mysql
3  RUN apt-get update && apt-get upgrade -y
4
```

Рис. 10. Dockerfile

```yaml
version: '3.3'
services:
        php-apache-env:
                container_name: php-apache
                build:
                        context: .
                        dockerfile: Dockerfile
                depends_on:
                        - db
                volumes:
                        - .:/var/www/html
                ports:
                        - 80:80
        db:
                container_name: db
                image: mysql
                restart: always
                environment:
                        MYSQL_ROOT_PASSWORD: 123
                volumes:
                        - db-data:/var/lib/mysql
                ports:
                        - 3306:3306
        phpmyadmin:
                image: phpmyadmin
                restart: always
                ports:
                        - 8080:80
                environment:
                        - PMA_ARBITRARY=1

volumes:
    db-data:

```

Рис. 11. Docker-compose.yml

ЗАКЛЮЧЕНИЕ

В результате выполнения работы были сформированы требования к системе, изучены информация о данной предметной области, технологиях, с помощью который спроектирована требуемая система. Были спроектированы база данных и визуал сайта.

Созданная информационная система содержит структуру и информацию предметной области «Больница». В состав базы данных входят таблицы, а также при помощи api были сделаны скрипты: ввода, вывода, удаления и обновления данных.

Визуал сайта позволяет облегчить труд врачей, персонала в регистратуре, предоставляя быструю информацию о расписании врачей, а также облегчает запись пациента на прием.

Таким образом, информационная система может быть использована для увеличения эффективности работы больницы путем ускорения таких операций, как запись пациентов на прием, ведение первичной финансовой отчетности, создание и хранение архивной информации о пациентах в удобной для поиска и обработки форме.

Ссылка на ИС: http://217.71.129.139:4660/index.php

Ссылка на репозиторий: https://github.com/ICheekiBreekiII/kursovaya1-

СПИСОК ЛИТЕРАТУРЫ

1. Бураков П.В. Введение в системы баз данных: Учебное пособие / П.В. Бураков, В.Ю. Петров. – Санкт-Петербург: СПбГУ ИМТО, 2010. – 128 с.

2. Роб П. Системы баз данных: проектирование, реализация и управление / П. Роб, К. Коронел. – Санкт-Петербург: Пер. с англ. - СПб.: БХВ-Петербург, 2004. – 1040 с.

3. JavaScript [Электронный ресурс]. – 2011. - Режим доступа: http://ru.wikipedia.org/wiki/JavaScript - Дата доступа: 13.12.2021

4. Средства создания сайтов [Электронный ресурс]. – 2011. – Режим доступа: http://www.in-internet.narod.ru/teor/sreda.html – Дата доступа: 13.12.2021

5. Html5book – Справочник по HTML, CSS, PHP, JavaScript и Jquery. [Электронный ресурс]. URL: https://html5book.ru/ - Дата доступа: 13.12.2021

6. Прохорский Г.В. Как сделать свою Web-страничку или сайт: учеб. – Москва, 2005 г. – 245с.

7. Шапошников И.В.Web-сайт своими руками: [практ. рук.] / И. В. Шапошников. – СПб.: БХВ - Санкт-Петербург, 2000. – 224с.

ПРИЛОЖЕНИЕ

Исходный код основных скриптов

create.php

```
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
header("Access-Control-Allow-Methods: POST");
include_once "../config/db.php";
include_once "../objects/doctors.php";
$database = new Database();
$db = $database->getConnection();
$doctors = new Doctors($db);
$data = json_decode(file_get_contents("php://input"));
if (!empty($data->POSITION) && !empty($data->NAME) && !empty($data->SURNAME)
&& !empty($data->SALARY) && !empty($data->MIDDLENAME) && !empty($data-
>ROOMNUM)) {
        $doctors->POSITION = $data->POSITION;
        $doctors->NAME = $data->NAME;
        $doctors->SURNAME = $data->SURNAME;

$doctors->MIDDLENAME = $data->MIDDLENAME;
        $doctors->ROOMNUM = $data->ROOMNUM;
        $doctors->SALARY = $data->SALARY;
        if ($doctors->create()) {
                http_response_code(201);
                echo json_encode(array("message" => "Запись была создана"),
JSON_UNESCAPED_UNICODE);
        }
        else {
                http_response_code(503);
                echo json_encode(array("message" => "Невозможно создать запись"),
JSON_UNESCAPED_UNICODE);
        }
}
else {
        http_response_code(400);
        echo json_encode(array("message" => "Невозможно создать запись. Данные
неполные"), JSON_UNESCAPED_UNICODE); } ?>
```

delete.php

```
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
header("Access-Control-Allow-Methods: DELETE");
header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers,
Authorization, X-Requested-With");
```

```
include_once "../config/db.php";
include_once "../objects/doctors.php";
$database = new Database();
$db = $database->getConnection();
$doctors = new Doctors($db);
$data = json_decode(file_get_contents("php://input"));
$doctors->id = $data->id;
if ($doctors->delete()){
        http_response_code(200);
        echo json_encode(array("message" => "Запись была удалёна."),
JSON_UNESCAPED_UNICODE); }
else {
        http_response_code(503);
        echo json_encode(array("message" => "Не удалось удалить запись"),
JSON_UNESCAPED_UNICODE); } ?>
```

read.php

```
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset-UTF-8");
include_once "../config/db.php";
include_once "../objects/doctors.php";
$database = new Database();
$db = $database->getConnection();
$doctors = new Doctors($db);
$stmt =$doctors->read();
$num = $stmt->rowCount();
if ($num > 0) {
        $Doctors_arr = array();
        $Doctors_arr['records'] = array();
        while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
                extract($row);
                $doctors_item = array(
                        "POSITION" => $POSITION,
                        "NAME" => $NAME,
                        "SURNAME" => $SURNAME,
                        "MIDDLENAME" => $MIDDLENAME,
                        "ROOMNUM" => $ROOMNUM,
                        "SALARY" => $SALARY,
                );
                array_push($Doctors_arr['records'], $doctors_item);
        }
        http_response_code(200);
        $json_data = json_encode($Doctors_arr);
        file_put_contents('doctors.json', $json_data);
}
else {
        http_response_code(404);
```

```
        echo json_encode(array("message" => "Записи не найдены"),
JSON_UNESCAPED_UNICODE); }
{
  header ('Location: test.html');
  exit();    } ?>
```

update.php

```
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
header("Access-Control-Allow-Methods: PUT");
include_once "../config/db.php";
include_once "../objects/doctors.php";
$database = new Database();
$db = $database->getConnection();
$doctors = new Doctors($db);
$data = json_decode(file_get_contents("php://input"));
$doctors->POSITION = $data->POSITION;
$doctors->NAME = $data->NAME;
$doctors->SURNAME = $data->SURNAME;
$doctors->MIDDLENAME = $data->MIDDLENAME;
$doctors->ROOMNUM = $data->ROOMNUM;
$doctors->SALARY = $data->SALARY;
if ($doctors->update()){
        http_response_code(200);
        echo json_encode(array("message" => "Заспись была изменена"),
JSON_UNESCAPED_UNICODE);
}
else {
        http_response_code(503);
        echo json_encode(array("message" => "Невозможно обновить запись"),
JSON_UNESCAPED_UNICODE); } ?>
```