

2 Разработка информационной системы поддержки психологического состояния

2.1 Проектирование схемы

Для удобства разработки информационной системы была спроектирована общая схема сайта (рисунок 1).

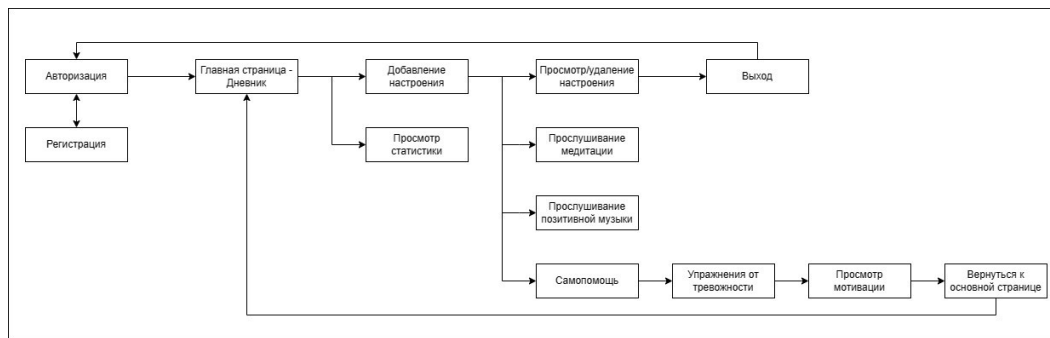


Рисунок 1 - Общая схема

В ней отображена логическая структура ссылок, согласно которым происходит перемещение пользователя между функционалом. На разрабатываемые страницы были созданы макеты, начальная изображена на рисунке 2.

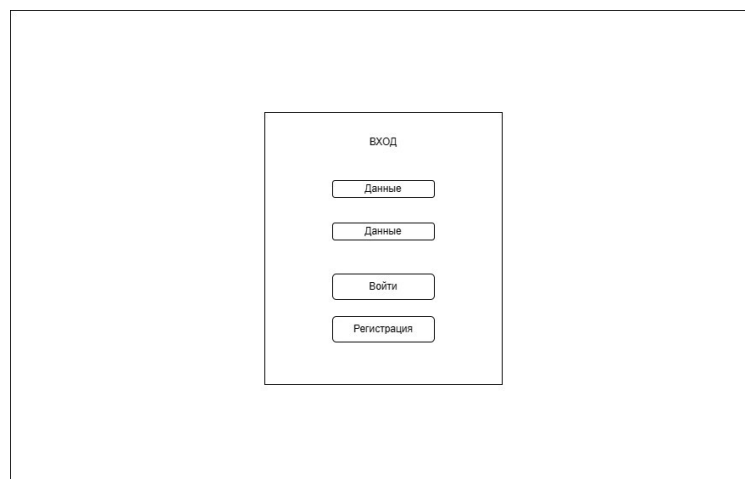


Рисунок 2 - Макет страницы авторизации

Заходя на сайт, пользователя будет встречать первая страница – авторизации. Имея логин и пароль, тот сможет перейти к главной странице. Тут же имеется и возможность обратиться к регистрации, при отсутствии таковой.

На рисунке 3 предоставлен прототип системы регистрации нового пользователя в систему.

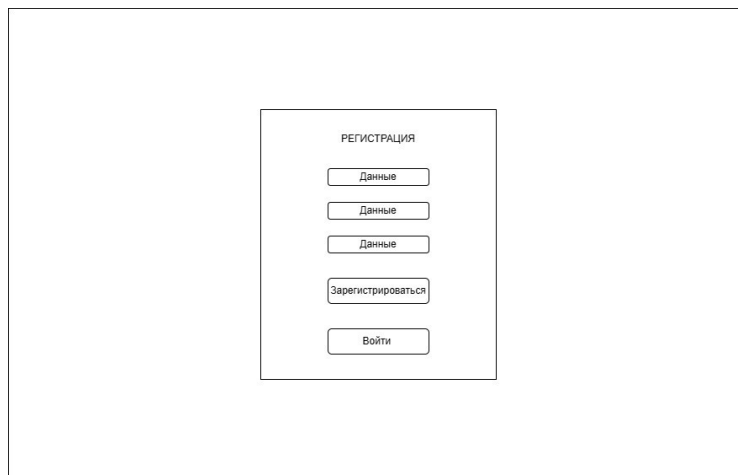


Рисунок 3 представляет собой прототип страницы регистрации. В центре экрана находится прямоугольная форма с заголовком "РЕГИСТРАЦИЯ". Внутрь этой формы встроены пять кнопок, расположенных вертикально: три кнопки с текстом "Данные", одна с текстом "Зарегистрироваться" и одна с текстом "Войти".

Рисунок 3 - Макет страницы регистрации

После успешной регистрации, пользователь авторизуется и автоматически переходит на основную страницу сайта.

Макет главной страницы предоставлен на рисунке 4.

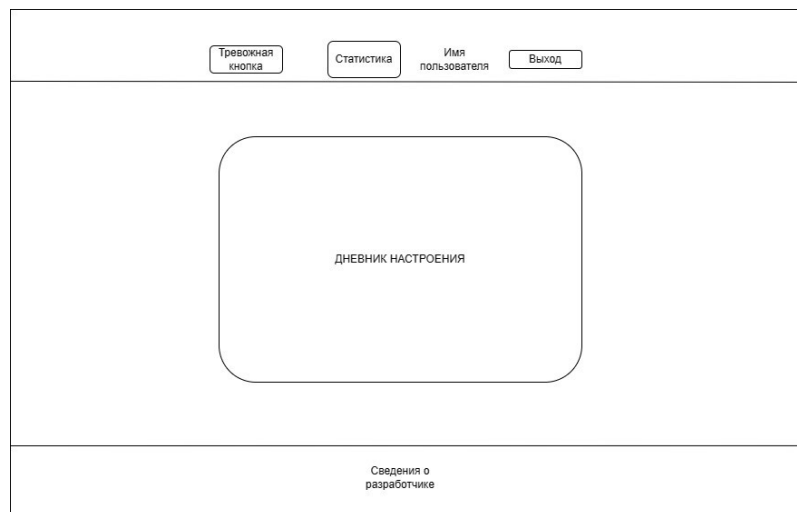


Рисунок 4 представляет собой макет главной страницы. В верхней части (шапке) расположены четыре кнопки: "Тревожная кнопка", "Статистика", "Имя пользователя" и "Выход". В центре экрана находится большая кнопка с закругленными углами и текстом "ДНЕВНИК НАСТРОЕНИЯ". В нижней части (подвале) сайта размещен текст "Сведения о разработчике".

Рисунок 4 - Макет главной страницы

Когда пользователь попадает на главную страницу сайта, он получает доступ к дневнику настроения. В шапке которого находится имя пользователя и кнопка выхода из системы. В подвале сайта будет указана информация об разработчике системы.

2.2 Разработка информационной системы

2.2.1 Разработка системы регистрации и авторизации

Перед входом на основную страницу сайта пользователю необходимо пройти систему аутентификации и/или регистрации. Без этого ему не будет доступен дневник.

Для того, чтобы реализовать данную систему необходимо начать с создания базы данных, в которую будут записываться, храниться и считываться данные пользователей.

На рисунке 5 расположена структура таблицы.

Сервер: 127.0.0.1:3306 / База данных: registerUser / Таблица: users									
Обзор Структура SQL Поиск Вставить Экспорт Импорт Привилегии Операции Триггеры									
Структура таблицы Связи									
#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
<input type="checkbox"/> 1	id	int(11)			Нет	Нем		AUTO_INCREMENT	Ещё
<input type="checkbox"/> 2	username	varchar(50)	utf8_general_ci		Нет	Нем			Ещё
<input type="checkbox"/> 3	password	varchar(50)	utf8_general_ci		Нет	Нем			Ещё
<input type="checkbox"/> 4	email	varchar(50)	utf8_general_ci		Нет	Нем			Ещё
<input type="checkbox"/> 5	create_datetime	datetime			Нет	Нем			Ещё

↑ ☐ Отметить все С отмеченными:

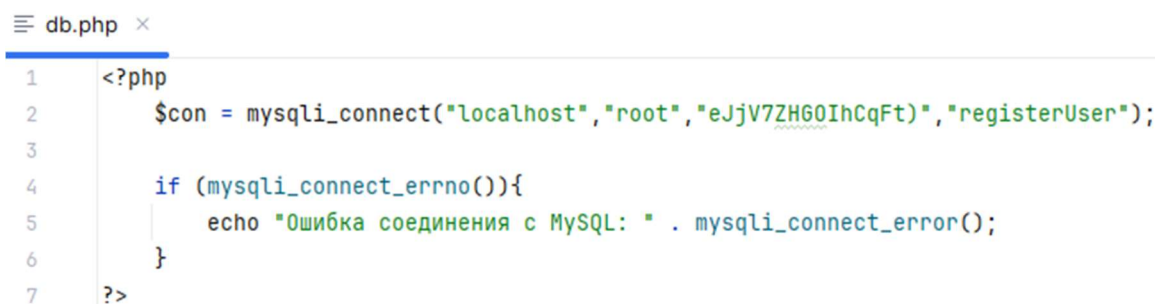
Рисунок 5 - Структура таблицы пользователей

Таблица «users» показана из web-приложения phpMyAdmin. В неё добавлены следующие атрибуты:

- id – идентификатор, являющийся первичным ключом;
- username – поле, в котором хранятся данные об имени пользователя;
- password – атрибут для хранения зашифрованного пароля пользователя;
- email – электронная почта пользователя;
- create_datetime – дата и время регистрации.

После создания базы данных, необходимо установить соединение с сервером.

На рисунке 6 произведено подключение к базе данных.

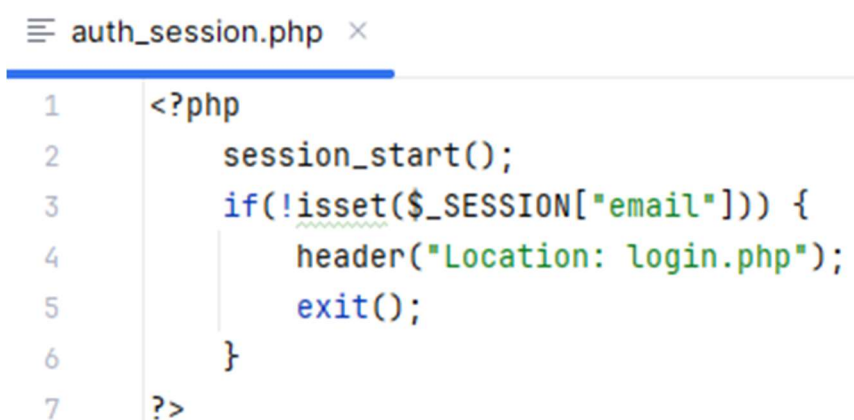


```
db.php x
1  <?php
2      $con = mysqli_connect("localhost","root","eJjV7ZH60IhCqFt"),"registerUser");
3
4      if (mysqli_connect_errno()){
5          echo "Ошибка соединения с MySQL: " . mysqli_connect_error();
6      }
7  ?>
```

Рисунок 6 - Соединение с базой данных

В файле db.php происходит соединение с сервером. В качестве сервера выступает localhost, имени пользователя – root, далее указан пароль и имя базы данных, в которой находится таблица.

Так же необходимо проверять авторизован ли пользователь в системе или нет. Код проверки на рисунке 7.



```
auth_session.php x
1  <?php
2      session_start();
3      if(!isset($_SESSION["email"])) {
4          header("Location: login.php");
5          exit();
6      }
7  ?>
```

Рисунок 7 - Проверка на авторизацию

Если пользователь не прошёл аутентификацию, то система перенаправит его на страницу входа.

Следом реализована система авторизации пользователя. Она предоставлена на рисунке 8.

```

38     <div class="form">
39         <div class="form-panel one">
40             <div class="form-header">
41                 <h1>Вход</h1>
42             </div>
43             <form method="post">
44                 <div class="form-group">
45                     <label for="password">Почта</label>
46                     <input type="email" class="login-input" name="email" placeholder="Email" >
47                 </div>
48                 <div class="form-group">
49                     <label for="password">Пароль</label>
50                     <input type="password" class="login-input" name="password" placeholder="Пароль"/>
51                 </div>
52                 <div class="form-group">
53                     <button type="submit">Вход</button>
54                 </div>
55                 <p class="link"><a href="registration.php">Зарегистрироваться</a></p>
56             </form>
57         </div>
58     </div>

```

Рисунок 8 - Авторизация пользователя в систему

Данный фрагмент кода создает окна для ввода почты и пароля, по которым в дальнейшем происходит проверка наличия пользователя в системе.

На рисунке 9 находится код, проверяющий наличие пользователя.

```

6         <link rel="stylesheet" href="css/style.css"/>
7     </head>
8     <body>
9     <?php
10         require('db.php');
11         session_start();
12         // После отправки формы - проверяет и создает сеанс
13         if (isset($_POST['email'])) {
14             $email = stripslashes($_REQUEST['email']);
15             $email = mysqli_real_escape_string($con, $email);
16             $password = stripslashes($_REQUEST['password']);
17             $password = mysqli_real_escape_string($con, $password);
18             // Проверяет наличие пользователя в БД
19             $query = "SELECT * FROM `users` WHERE email='$email'
20                     AND password= '' . md5($password) . ''";
21             $result = mysqli_query($con, $query) or die(mysql_error());
22             $rows = mysqli_num_rows($result);
23             if ($rows == 1) {
24                 $users = mysqli_fetch_assoc($result);
25                 $_SESSION['email'] = $email;
26                 $_SESSION['username'] = $users['username'];
27                 // Перенаправляет на основную страницу
28                 header("Location: calendar.php");
29             } else {
30                 echo "<div class='form'>
31                     <div class='error'>
32                         <h2>Неверная Почта и/или Пароль</h2><br/>
33                         <p class='link'><a href='login.php'>Войти</a> снова.</p>
34                     </div></div>";
35             }

```

Рисунок 9 - Проверка наличия пользователя

Если в результате проверки в базе данных находится заходящий пользователь – его перенаправит на главную страницу сайта, в ином случае будет выводиться ошибка о том, что данные введены некорректно.

В папке css находится файл style.css., который подключен в строке 6, это позволяет отобразить на странице примененные к ней стили.

На рисунке 10 предоставлена часть используемых стилей. Они визуализируют внешний вид сайта, формы авторизации и ввода логина с паролем.

```
6 body {
7     background: -webkit-linear-gradient(45deg, rgba(66, 183, 245, 0.8) 0%, rgba(66, 245, 189, 0.4) 100%);
8     background: linear-gradient(45deg, rgba(66, 183, 245, 0.8) 0%, rgba(66, 245, 189, 0.4) 100%);
9     color: rgba(0, 0, 0, 0.6);
10    font-family: sans-serif;
11    font-size: 14px;
12    line-height: 1.6em;
13    -webkit-font-smoothing: antialiased;
14    -moz-osx-font-smoothing: grayscale;
15 }
16
17 .form {
18     z-index: 15;
19     position: relative;
20     background: #FFFFFF;
21     width: 600px;
22     border-radius: 4px;
23     box-shadow: 0 0 30px rgba(0, 0, 0, 0.1);
24     box-sizing: border-box;
25     margin: 100px auto 10px;
26     overflow: hidden;
27 }
28
29 .form-group {
30     display: flex;
31     flex-wrap: wrap;
32     justify-content: space-between;
33     margin: 0 0 20px;
34 }
```

Рисунок 10 - Стили страницы авторизации

Селектор body, при помощи background задал фон сайта – градиент цвета, значения font-family, font-size определяют шрифт и его размер, а line-height - величину межстрочного интервала. Свойство -webkit-font-smoothing: antialiased сглаживает шрифт. Для того, чтобы это применялось на всех браузерах использовано -moz-osx-font-smoothing: grayscale.

В классе form описан стиль формы для авторизации/регистрации, был задан порядок наложения элементов (z-index), нейтральный цвет фона, расположение, сглаживание углов, а также создана тень за элементом.

В form-group задано расположение ячеек для полей ввода данных.

Страница регистрации создана подобным образом. На рисунке 11 показан её код.

```
74 <div class="form">
75 <div class="form-panel one">
76 <div class="form-header">
77 <h1>Регистрация</h1>
78 </div>
79 <form method="post">
80 <div class="form-group">
81 <label for="username">Имя пользователя</label>
82 <input type="text" class="login-input" name="username" placeholder="Имя пользователя" required />
83 </div>
84 <div class="form-group">
85 <label for="password">Почта</label>
86 <input type="email" class="login-input" name="email" placeholder="Email" />
87 </div>
88 <div class="form-group">
89 <label for="password">Пароль</label>
90 <input type="password" class="login-input" name="password" placeholder="Пароль" />
91 </div>
92 <div class="form-group">
93 <button type="submit">Зарегистрироваться</button>
94 </div>
95 <p class="link"><a href="login.php">Нажмите сюда, чтобы войти</a></p>
96 </form>
97 </div>
98 </div>
```

Рисунок 11 - Страница регистрации

Аналогично предыдущей странице здесь реализована визуальная часть страницы— поля имени пользователя, почта и пароль. Так же имеется переход на страницу авторизации. На этой странице необходимо несколько проверок.

Проверка на заполнение пользователем обязательного поля — почты, а также на наличие такого пользователя в системе на рисунке 12.

```
22 if (empty($_POST['email']) or empty($_POST['password'])) {
23     echo "<div class='form'>
24         <div class='error'>
25             <h3>Регистрация не удалась. Не заполнена почта или пароль.</h3><br/>
26             <p class='link'><a href='registration.php'>Зарегистрироваться</a> заново.</p>
27         </div></div>";
28 } else {
29     $query = "SELECT * FROM `users` WHERE email='$_POST[email]'";
30     $result = mysqli_query($con, $query);
31     $rows = mysqli_num_rows($result);
32
33     //Проверка на наличие такого пользователя
34     if ($rows == 1) {
35         echo "<div class='form'>
36             <div class='error'>
37                 <h3>Такой пользователь уже есть</h3><br/>
38                 <p class='link'><a href='registration.php'>Зарегистрироваться</a> заново.</p>
39             </div></div>";
40     }
```

Рисунок 12 - Проверка почты

Так как поле электронного адреса является обязательным, то пользователю будет выведена ошибка, в случае его не заполнения. Во избежание повторной регистрации на одну почту, система выведет соответствующую ошибку.

На рисунке 13 реализована проверка на действительность почты.

```
50 //Проверка на действительность почты
51 elseif (preg_match("/^([a-z0-9]+(?:[-_]?[a-z0-9]+)?@[a-z0-9_-]+(?:\.[a-z0-9]+)?\.[a-z]{2,5})$/i", $email))
52 {
53     $query = "INSERT into `users` (username, password, email, create_datetime)
54             VALUES ('$username', '" . md5($password) . "', '$email', '$create_datetime)";
55     $result = mysqli_query($con, $query);
56     echo "<div class='form'>
57         <div class='regist'>
58             <h2>Вы успешно зарегистрированы!</h2><br/>
59             <p class='link'>Перейти к <a href='login.php'>Входу</a></p>
60         </div></div>";
61 }
62 else {
63     echo "<div class='form'>
64         <div class='error'>
65             <h3>Регистрация не удалась. Недействительный адрес почты</h3><br/>
66             <p class='link'><a href='registration.php'>Зарегистрироваться</a> заново.</p>
67         </div></div>";
68 }
69 }
```

Рисунок 13 - Действительность почты

Для того, чтобы символ «@» не был пропущен – в коде, предоставленном выше использовался атрибут «name="email"». Он выведет ошибку о том, что адрес электронной почты должен содержать данный символ. Чтобы пользователь ввел существующую почту, проходит проверка на наличие недопустимых для неё символов.

На рисунке 13 показана проверка на пароль.

```
//Минимальная длина пароля - 6
elseif (strlen($_POST["password"]) < 6)
{
    echo "<div class='form'>
        <div class='error'>
            <h3>Регистрация не удалась. Пароль должен содержать не менее 6 символов.</h3><br/>
            <p class='link'><a href='registration.php'>Зарегистрироваться</a> заново.</p>
        </div></div>";
}
```

Рисунок 14 - Проверка пароля

Необходимо, чтобы длина пароля пользователя была не менее 6 введенных символов. Если же пользователь вводит меньшее – выведется ошибка о том, что пароль не прошёл проверку.

При успешной регистрации данные будут занесены в базу данных, также будет добавлена дата и время регистрации. Пароль пользователя будет зашифрован, для этого использован способ хеширование строки md5.

Для данной страницы также используются стили с предыдущей.

Стили, применяемые к выводу ошибок и успешному прохождению регистрации продемонстрированы на рисунке 15.

```
113     .error {
114         padding: 60px 60px 60px 60px;
115         box-sizing: border-box;
116         text-align: center;
117         color: #f00;
118     }
119
120     .regist {
121         padding: 60px 60px 60px 60px;
122         box-sizing: border-box;
123         text-align: center;
124         color: #351abd;
125     }
```

Рисунок 15 - Стил ь ошибок

Вывод совершается в окнах с внутренними отступами в 60 пикселей, текст центрируется. В случае вывода ошибки текст будет иметь красный цвет, в ином – синий.

Для того, чтобы гиперссылки имели стабильный цвет, был применен стиль на рисунке 16.

```
102     .link {
103         color: #666;
104         font-size: 16px;
105         text-align: center;
106         margin-bottom: 0px;
107     }
108
109     .link a {
110         color: #666;
111     }
```

Рисунок 16 - Стили ссылок

При переходе по ссылке, он сохраняет цвет. А также задает стиль для всех ссылок на сайте. Они имеют серый цвет, размер в 16 пикселей и центрируются.

На рисунке 17 предоставлен выход.

```
logout.php x
1  <?php
2      session_start();
3      //Закончить сессию
4      if(session_destroy()) {
5          //Возвращает на вход
6          header("Location: login.php");
7      }
8  ?>
```

Рисунок 17 - Выход из системы

Данный фрагмент кода завершает сессию и выводит пользователя на страницу авторизации.

После того как все фрагменты кода были реализованы возможно посмотреть разработанный вход и регистрацию.

Разработанный вход продемонстрирован на рисунке 18.

Рисунок 18 - Страница входа

Данный вид соответствует предоставленному макету входа. В самом верху окна расположена надпись «Вход», ниже предоставлены окна ввода данных с размещенными в них подсказками, после расположена кнопка входа, а также гиперссылка, ведущая на страницу регистрации.

На рисунке 19 предоставлена страница регистрации.

Рисунок 19 - Страница регистрации

Все поля и кнопки расположены на соответствующих позициях, потому полученный результат удовлетворяет спроектированному прототипу.

Далее была произведена разработка главной страницы сайта.

2.2.2 Разработка главной части информационной системы

На основной странице необходимо реализовать дневник настроения пользователя.

Шапка сайта реализована на рисунке 20.

```

15 <header>
16   <div class="header">
17     <button id="btnPhone" class="buttonHelp">ТРЕВОЖНАЯ КНОПКА</button>
18     <button id="colorStatsButton">Посмотреть статистику</button>
19     <p class="username">Здравствуйте, <?<= htmlspecialchars($_SESSION['username']) ?>!</p>
20     <button class="button link"> <a href="logout.php"> Выйти</button></a>
21   </div>
22 </header>

```

Рисунок 20 - Шапка страницы

Данный код выводит в шапку сайта имя пользователя и кнопки тревоги, статистики и выхода. Так же в строке 12 подключены стили, применяемые к данной странице.

Сам дневник выполнен в виде календаря с использованием модальных окон.

На рисунке 21 показан код шаблона календаря.

```
21 <div class="container">
22   <div class="hea">
23     <div id="month"></div>
24     <div>
25       <button id="btnBack">Назад</button>
26       <button id="btnNext">Вперед</button>
27     </div>
28   </div>
29   <div class="weekdays">
30     <div>ПН</div>
31     <div>Вт</div>
32     <div>Ср</div>
33     <div>Чт</div>
34     <div>Пт</div>
35     <div>Сб</div>
36     <div>Вс</div>
37   </div>
38   <div id="calendar"></div>
39 </div>
```

Рисунок 21 - Шаблон дневника

Вначале выводится текущий месяц и кнопки продвижения по месяцам, следом дни недели и ниже формируется уже сам календарь. Для того, чтобы у пользователя была возможность ввести данные на каждое число месяца, были использованы модальные окна.

На рисунке 22 предоставлен код модального окна.

```
40 <div id="calendar"></div>
41 </div>
42 <div id="modal"></div>
43 <div id="addEvent">
44   <h2>События</h2>
45   <input type="text" id="txtTitle" placeholder="Событие" /> <br>
46   <h2>Выберите подходящее настроение:</h2>
47   <select id="eventColor">
48     <option value="#3399ff">Счастливое</option>
49     <option value="#a1d98b">Хорошее</option>
50     <option value="#f7941d">Так себе</option>
51     <option value="#f7941d">Плохое</option>
52     <option value="#f7941d">Ужасное</option>
53   </select> <br>
54   <button id="btnSave">Сохранить</button>
55   <button class="btnClose">Закрыть</button>
56 </div>
57
58 <div id="viewEvent">
59   <h2>События</h2>
60   <p id="eventText">События</p>
61   <h2>Прослушайте медитацию:</h2>
62   <audio controls src="/media/meditation.mp3"></audio>
63   <audio controls src="/media/meditation2.mp3"></audio>
64   <br>
65   <h2>Прослушайте весёлую музыку:</h2>
66   <audio controls src="/media/funny.mp3"></audio>
67   <br>
68   <h2>Тренировка самопомощи:</h2>
69   <p class="linkhelp"><a href="/help.html" rel="nofollow" target="_blank">Помощь</a></p>
70   <button id="btnDelete">Удалить</button>
71   <button class="btnClose">Закрыть</button>
72 </div>
73
74 <div id="Phone">
75   <h2>Позвоните по номеру:</h2>
76   <p><a href="tel:78002000122" class="linkhelp">8-800-2000-122</a></p>
77   <button class="btnClose">Закрыть</button>
78 </div>
79 <script src="script.js"></script>
```

Рисунок 22 - Модальное окно

В форме имеется два функционала: поле для ввода текста и выпадающий список с выбором настроения.

Сохраненные события отображаются в календаре, для их просмотра достаточно нажать на них. В случае необходимости пользователь может удалить какое-либо из настроений.

Для генерации календаря и всех функций был использован язык javascript, скрипт был подключен на 79 строке.

Внизу сайта находится подвал (рисунок 23).

```
74 <footer>
75     <span>Выполнено студентом: Громовой Ксенией Андреевной ИБ-913</span><br>
76     <span>СПбГУТ им. проф. Бонч-Бруевича</span>
77 </footer>
```

Рисунок 23 - Подвал сайта

В нем выводится информация о разработчике сайта.

Когда внешний вид настроен, необходимо организовать генерацию календаря.

На рисунке 24 произведена генерация месяцев календаря.

```
10 function loadCalendar() : void {
11     const dt : Date = new Date();
12
13     if (navigation != 0) {
14         dt.setMonth( month: new Date().getMonth() + navigation);
15     }
16     const day : number = dt.getDate();
17     const month : number = dt.getMonth();
18     const year : number = dt.getFullYear();
19     monthBanner.innerText = `${russianMonths[month]} ${year}`;
20     calendar.innerHTML = "";
21     const dayInMonth : number = new Date(year, monthIndex: month + 1, date: 0).getDate();
22     const firstDayOfMonth : Date = new Date(year, month, date: 1);
23     const dateText : string = firstDayOfMonth.toLocaleDateString( locales: "en-GB", options: {
24         weekday: "long",
25         year: "numeric",
26         month: "numeric",
27         day: "numeric",
28     });
```

Рисунок 24 - Генерация месяцев

Создается новый объект с текущей датой и временем. Если значения навигации не равно нулю, то в качестве месяца устанавливается текущий и к нему прибавляется значение навигации. Это позволяет календарю переходить к другому месяцу. Затем извлекается день, месяц и год и задаются на русском языке.

Далее реализованы дни в календаре (рисунок 25).

```

30     const dayString : string = dateText.split( separator: ", ")[0];
31     const emptyDays : number = weekdays.indexOf(dayString);
32
33     for (let i : number = 1; i <= dayInMonth + emptyDays; i++) {
34         const dayBox : HTMLDivElement = document.createElement( tagName: "div");
35         dayBox.classList.add("day");
36         const monthVal : string | number = month + 1 < 10 ? "0" + (month + 1) : month + 1;
37         const dateVal : string | number = i - emptyDays < 10 ? "0" + (i - emptyDays) : i - emptyDays;
38         const dateText : string = `${dateVal}-${monthVal}-${year}`;
39         if (i > emptyDays) {
40             dayBox.innerText = i - emptyDays;
41             //Настройка
42             const eventOfTheDay = events.find((e) : boolean => e.date == dateText);
43
44             if (i - emptyDays === day && navigation == 0) {
45                 dayBox.id = "currentDay";
46             }
47
48             if (eventOfTheDay) {
49                 const eventDiv : HTMLDivElement = document.createElement( tagName: "div");
50                 eventDiv.classList.add("event");
51                 eventDiv.innerText = eventOfTheDay.title;
52                 eventDiv.style.backgroundColor = eventOfTheDay.color; // добавляет цвет
53                 dayBox.appendChild(eventDiv);
54             }
55
56             dayBox.addEventListener( type: "click", listener: () : void => {
57                 showModal(dateText);
58             });
59         } else {
60             dayBox.classList.add("plain");
61         }
62         calendar.append(dayBox);
63     }
64 }

```

Рисунок 25 – Фиксация дней в календаре

Фрагмент кода создает тело календаря. Сначала вычисляется количество дней в текущем месяце, затем первый день месяца и день недели. Так же вычисляется количество пустых дней до первого дня месяца, то есть количество дней в предыдущем месяце, которое необходимо пропустить.

Каждая итерация в цикле создает новый элемент с классом div day, если итерация больше, чем количество пустых дней, то задается внутренний текст на номер дня (1-31), в ином случае добавляется класс для обозначения пустого дня.

Затем вычисляется, является ли день текущим и добавляется идентификатор true к if currentDay. Проверяется наличие введенного настроя, связанное с текущим днем (если таковое имеется) и отображается с текстом и цветом.

В конце добавляется обработчик событий к дню, для вывода модального окна при щелчке.

На рисунке 26 предоставлен код функционала кнопок.

```
76 btnBack.addEventListener( type: "click", listener: () : void => {
77     navigation--;
78     loadCalendar();
79 });
80 btnNext.addEventListener( type: "click", listener: () : void => {
81     navigation++;
82     loadCalendar();
83 });
84 modal.addEventListener( type: "click", closeModal);
85 closeButton.forEach( callbackfn: (btn : Element) : void => {
86     btn.addEventListener( type: "click", closeModal);
87 });
88 btnDelete.addEventListener( type: "click", listener: function () : void {
89     events = events.filter((e) : boolean => e.date !== clicked);
90     localStorage.setItem("events", JSON.stringify(events));
91     closeModal();
92 });
93
94 btnSave.addEventListener( type: "click", listener: function () : void {
95     if (txtTitle.value) {
96         const eventColor = document.querySelector( selectors: "#eventColor").value;
97         events.push({
98             date: clicked,
99             title: txtTitle.value.trim(),
100             color: eventColor, // сохраняет выбранный цвет
101         });
102         txtTitle.value = "";
103         localStorage.setItem("events", JSON.stringify(events));
104         closeModal();
105     } else {
106         txtTitle.classList.add("error");
107     }
108 });
109
110 PhoneButtons.onclick = function () : void {
111     showPhoneModal();
112 }
113 }
```

Рисунок 26 - Функционал кнопок

btnBack и btnNext – кнопки, при нажатии на которые уменьшается и увеличивается переменная навигации. Это позволяет перелистывать месяцы.

Далее настроены кнопки модальных окон:

— clodeButtons – закрывает модальное окно;

- btnDelete – удаляет событие из календаря;
- btnSave – передает значения даты, текста и цвета в событие;
- PhoneButtons – запускает функцию showPhoneModal, которая отображает окно с выводом телефона экстренной помощи.

Функционал модального окна создан на рисунке 27.

```

121 function showModal(dateText) : void {
122     clicked = dateText;
123     const eventOfTheDay = events.find((e) : boolean => e.date == dateText);
124     if (eventOfTheDay) {
125         //Событие уже есть
126         document.querySelector(selectors: "#eventText").innerText = eventOfTheDay.title;
127         viewEventForm.style.display = "block";
128     } else {
129         //Добавить новое событие
130         addEventForm.style.display = "block";
131     }
132     modal.style.display = "block";
133 }

```

Рисунок 27 - Функции модального окна

Функция showModal ищет событие в массиве. Если событие найдено, оно выводит соответствующий текст элемента. Если события нет – отображает форму добавления события на день.

Для того, чтобы было возможно отследить изменения в состоянии была создана функция просмотра статистики (рисунок 28).

```

165 function displayColorStats() : void {
166     const newWindow : WindowProxy = window.open('url: **', target: "ColorStats", features: "width=400,height=300");
167     newWindow.document.write(`
168     <html>
169     <head>
170     <title>Статистика настроения</title>
171     </head>
172     <body>
173     <h1>Статистика настроения</h1>
174     <div style="display: flex; flex-direction: column; align-items: center;">
175     ${Object.entries(colorStats) .map((string, number) => `
176     .map(
177     ([description : string , count : number ]) : string => `
178     <div style="display: flex; align-items: center; margin-bottom: 10px;">
179     <div style="width: ${(count / events.length) * 100}%; height: 20px; background-color: ${getColor(description)}; margin-right: 10px;"></div>
180     <div style="font-size: 24px;">${description} - ${count} (${Math.round( x: (count / events.length) * 100)}%)</div>
181     </div>`
182     ) string[]
183     .join("**")}
184     </div>
185     </body>
186     </html>
187     `);
188 }

```

Рисунок 28 - Функция статистики

displayColorStats, при нажатии на кнопку «Посмотреть статистику» выводит в отдельном окне собранные данные по самочувствию пользователя.

В экстренный случай клиент может воспользоваться тревожной кнопкой (рисунок 29).


```

211 function showPhoneModal() : void {
212     PhoneForm.style.display = "block";
213     modal.style.display = "block";
214 }
215
216 //Закреть модальное окно
1+ usages
217 function closeModal() : void {
218     viewEventForm.style.display = "none";
219     PhoneForm.style.display = "none";
220     addEventForm.style.display = "none";
221     modal.style.display = "none";
222     clicked = null;
223     loadCalendar();
224 }

```

Рисунок 29 - Функция тревожной кнопки и закрытия окон

Операция showPhoneModal открывает модальное окно с выводом и вызовом телефона помощи. Чтобы все из окон закрывались использована closeModal, которая позволяет, при нажатии соответствующей кнопки, убирать с экрана отображение.

Часть стилей для дневника предоставлена на рисунке 30.

Класс calendar задает размеры и расположение календаря. Day формирует окошки для каждого дня. Чтобы пользователю было удобно ориентироваться, имеется стиль: day: hover, который выделяет день, на который наведен курсор. currentDay окрашивает другим цветом текущий день. event – задает размеры и вид отображения событий в календаре. По умолчанию цвет событий – серый, однако он сменяется в зависимости от выбора пользователя.

```

113 #calendar {
114     width: 100%;
115     margin: auto;
116     display: flex;
117     flex-wrap: wrap;
118 }
119
120 .day {
121     width: 100px;
122     height: 100px;
123     padding: 10px;
124     cursor: pointer;
125     margin: 5px;
126     box-sizing: border-box;
127     box-shadow: 0px 0px 3px #cbd4c2;
128     color: #7f8fa6;
129     font-weight: 500;
130     display: flex;
131     flex-direction: column;
132     justify-content: space-between;
133 }
134 .day:hover {
135     background-color: rgba(112, 111, 211, 0.1);
136     color: #706fd3;
137 }
138
139 #currentDay {
140     background-color: #706fd3;
141     color: #fff;
142 }
143
144 .event {
145     font-size: 11px;
146     padding: 3px;
147     background-color: #3d3d3d;
148     color: #141313;
149     border-radius: 5px;
150     max-height: 50px;
151     overflow: hidden;
152 }

```

Рисунок 30 - Стили календаря

После того как все фрагменты кода были реализованы возможно посмотреть разработанную страницу дневника.

Разработанный дневник предоставлен на рисунке 31.

Полученный результат соответствует созданному макету главной страницы. В шапке сайта выведены тревожная и статистическая кнопки, имя пользователя, справа от него выход. Ниже расположен сам дневник, реализована возможность листать месяцы, визуально отображается текущий день. Все события имеют свой вывод, который зависит от выбранного в форме настроения.

Каждое число календаря — это ячейка, при нажатии на которую вызывается модальное окно с возможностью ввода информации и выбора самочувствия.

В подвале сайта выведены данные о разработчике сайта.

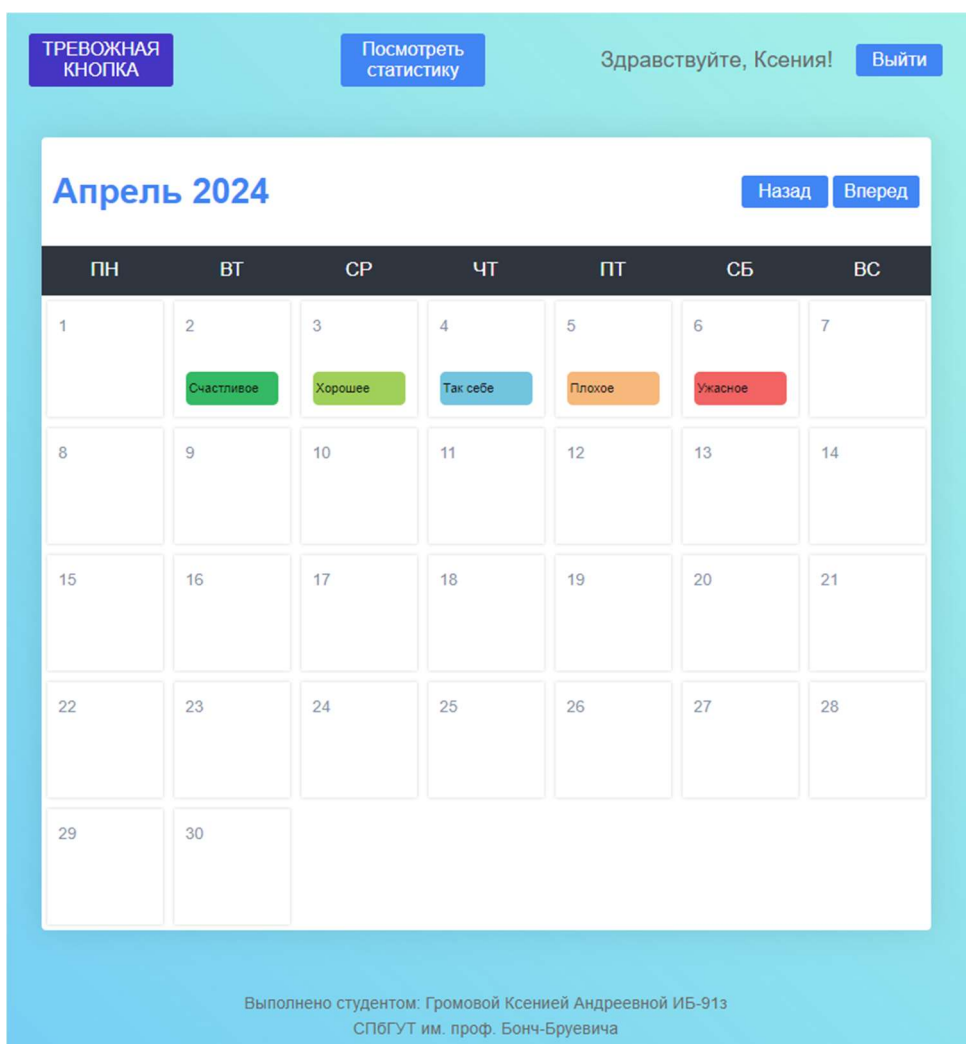


Рисунок 31 – Главная страница

В поле события пользователь может записать любые свои данные, будь то дела за день или то, что стало источником его состояния. В числе данные будут выводиться сокращенно, однако на них можно нажать, чтобы вывести и прочитать полноценно.

Если у человека плохое самочувствие или он желает отдохнуть от насыщенных мероприятий, ему предложены различные способы поддержки состояния и избавления от тревоги (рисунок 32).

На свой выбор он может прослушать медитативную или же позитивную мелодию, что позволит расслабиться и отвлечься от событий на день. В случае, если этого оказалось недостаточно, имеется возможность перейти на страницу самопомощи.

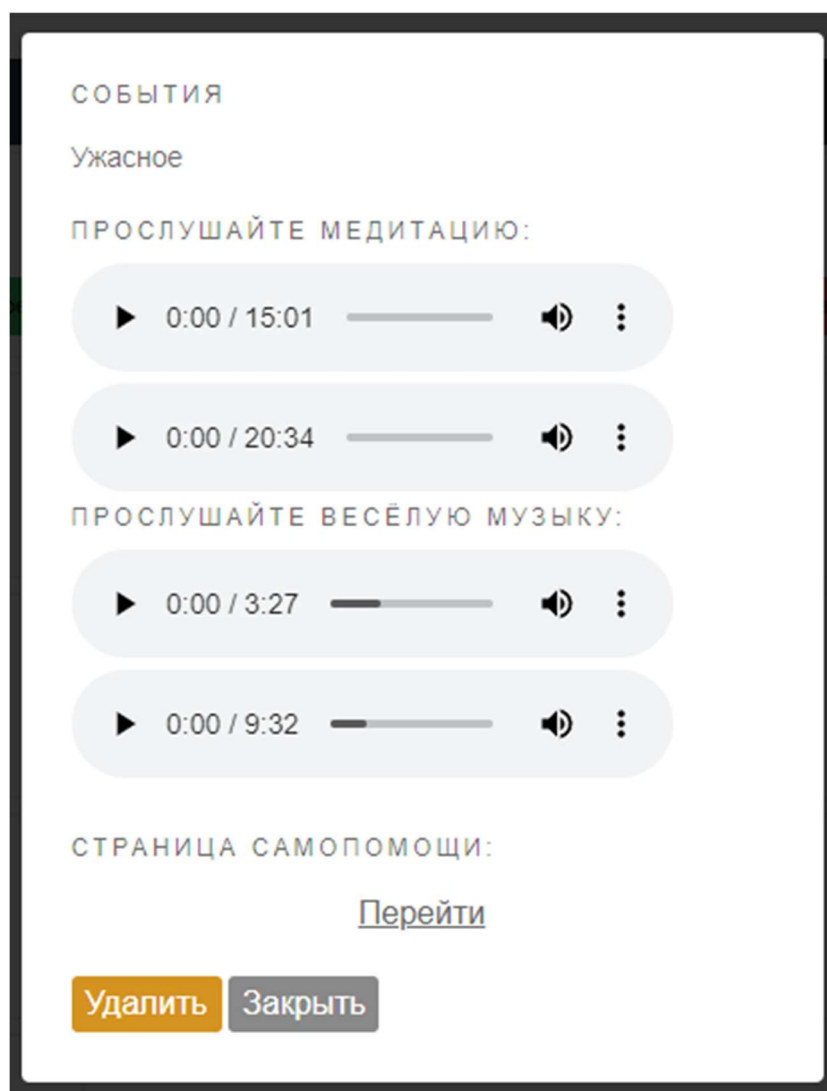


Рисунок 32 - Окно поддержки

Когда человек нажимает на ссылку «Перейти», он оказывается на следующей странице сайта, что расположена помочь ему справиться со своими эмоциями и состоянием.

Помимо описания техник, которые направлены на уменьшение тревоги, в нем расположен слайдер с мотивирующими картинками.

За возможность листать картинки, а также их отображение отвечает скрипт (рисунок 33).

Функция `showSlide` задает фотографии стиль, в зависимости от того, должна ли она сейчас быть отображена пользователю или нет. Если картинка не выбрана как активная, применяется стиль – `inactive`, накладывающий на неё эффект размытия.

```

54     const slider = document.querySelector('.slider');
55     const slides = slider.querySelectorAll('.slide');
56     const prev = slider.querySelector('.prev');
57     const next = slider.querySelector('.next');
58
59     let currentSlide = 0;
60
61     1+ usages
62     function showSlide(n) {
63         slides.forEach((slide) => {
64             slide.classList.remove('active');
65             slide.classList.add('inactive');
66         });
67         slides[n].classList.remove('inactive');
68         slides[n].classList.add('active');
69         currentSlide = n;
70     }
71
72     1+ usages
73     function nextSlide() {
74         showSlide((currentSlide + 1) % slides.length);
75     }
76
77     1+ usages
78     function prevSlide() {
79         showSlide(currentSlide === 0 ? slides.length - 1 : currentSlide - 1);
80     }
81
82     showSlide(currentSlide);
83
84     prev.addEventListener('click', prevSlide);
85     next.addEventListener('click', nextSlide);

```

Рисунок 33 – Слайдер

На рисунке 34 описаны применяемые стили.

Для плавного изменения состояния элемента используется transition: transform. Помимо этого, slide и slide img задают размеры для самого слайдера и картинок.

Чтобы скрытые изображения имели мутный эффект к ним применяется slide.inactive, в котором filter: blur(5px) задает тот самый результат. К отображаемой картинке применяется стиль slide.active, отменяющий наложенный фильтр.

```

136 .slide {
137     width: 25%;
138     height: 100%;
139     position: relative;
140     transition: transform 0.5s ease;
141 }
142
143 .slide img {
144     width: auto;
145     height: 302px;
146 }
147
148 .slide.active {
149     opacity: 1;
150     filter: none;
151 }
152
153 .slide.inactive {
154     opacity: 1;
155     filter: blur(5px);
156 }

```

Рисунок 34 - Стили слайдера

Реализованная страница поддержки расположена на рисунке 35.

Вначале предложены два способа избавления от беспокойства и негативных мыслей, ниже отображены изображения, расположенные на поднятие духа и состояния. Также добавлена ссылка возвращения к дневнику.

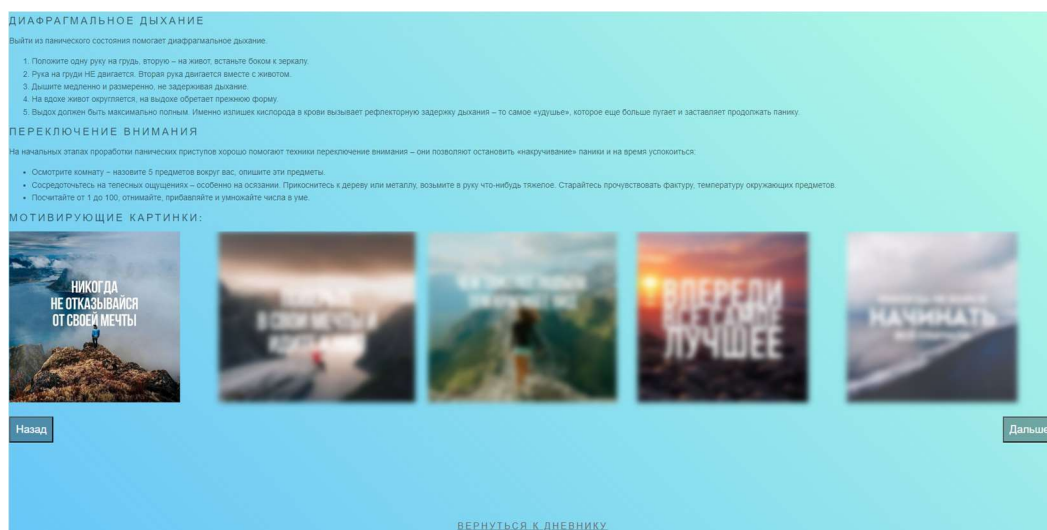


Рисунок 35 - Страница самопомощи