```
=== index.html
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Мастер установки Сервера</title>
  <!-- Подключаем чистый современный стиль -->
    <link rel="stylesheet" href="assets/bootstrap.min.css">
    <style>
        body { background: #f4f6f9; min-height: 100vh; display: flex; align-
items: center; justify-content: center; }
        .wizard-card { background: white; border-radius: 12px; box-shadow: 0
8px 24px rgba(0,0,0,0.05); width: 650px; min-height: 450px; display: flex;
flex-direction: column; justify-content: space-between; padding: 40px; }
        .step { display: none; }
        .step.active { display: block; }
        .step-header { border-bottom: 2px solid #efefef; padding-bottom:
15px; margin-bottom: 25px; }
    </style>
</head>
<body>

<div class="wizard-card">
    <div class="wizard-content">
    <!-- 1.2.1 Окно приветствия -->
        <div id="step-1" class="step active">
            <div class="step-header"><h4>Шаг 1: Приветствие</h4></div>
            <p class="lead">Добро пожаловать в веб-инсталлятор вашего нового
сервера!</p>
            <p>Этот мастер поможет развернуть и настроить операционную систему без
подключения монитора к серверу. Все действия выполняются удаленно через этот
браузер.</p>
        </div>

    <!-- 1.2.2 Выбор языка -->
        <div id="step-2" class="step">
            <div class="step-header"><h4>Шаг 2: Язык системы</h4></div>
            <label class="form-label">Выберите основной язык для устанавливаемой
ОС:</label>
            <select id="sys_lang" class="form-select form-select-lg">
                <option value="ru_RU.UTF-8">Русский (ru_RU)</option>
                <option value="en_US.UTF-8">English (en_US)</option>
            </select>
        </div>

    <!-- 1.2.3 Выбор раскладки клавиатуры -->
        <div id="step-3" class="step">
            <div class="step-header"><h4>Шаг 3: Раскладка клавиатуры</h4></div>
            <label class="form-label">Выберите раскладку клавиатуры по
умолчанию:</label>
            <select id="sys_layout" class="form-select form-select-lg">
```

```
                <option value="ru">Русская (ru)</option>
                <option value="us">Английская (us)</option>
            </select>
        </div>

    <!-- 1.2.4 Выбор часового пояса -->
        <div id="step-4" class="step">
            <div class="step-header"><h4>Шаг 4: Часовой пояс</h4></div>
            <label class="form-label">Укажите ваш часовой пояс:</label>
            <select id="sys_timezone" class="form-select form-select-lg">
                <option value="Europe/Moscow">Москва (GMT+3)</option>
                <option value="Europe/Kaliningrad">Калининград
(GMT+2)</option>
                <option value="Asia/Yekaterinburg">Екатеринбург
(GMT+5)</option>
                <option value="UTC">UTC (Универсальное время)</option>
            </select>
        </div>

    <!-- 1.2.5 Имя компьютера -->
        <div id="step-5" class="step">
            <div class="step-header"><h4>Шаг 5: Сетевое имя
(Hostname)</h4></div>
            <label class="form-label">Введите имя вашего будущего
сервера:</label>
            <input type="text" id="sys_hostname" class="form-control form-
control-lg" value="my-cool-server" oninput="validateHostname(this)">
            <div class="form-text text-danger">Разрешена только латиница (a-
z), цифры и дефис.</div>
        </div>



        <!-- 1.2.6 Разбивка диска и RAID -->
        <div id="step-6" class="step">
            <div class="step-header"><h4>Шаг 6: Разметка накопителей</h4></div>
            <label class="form-label">Выберите конфигурацию дисковой
подсистемы:</label>
            <div class="form-check mb-3">
                <input class="form-check-input" type="radio"
name="disk_mode" id="disk_default" value="default" checked>
                <label class="form-check-label" for="disk_default">
                    <strong>Режим по умолчанию:</strong> Система + 25GB под
установку, остальное в резерв под домашнюю папку пользователя
                </label>
            </div>
            <div class="form-check mb-4">
                <input class="form-check-input" type="radio"
name="disk_mode" id="disk_raid" value="raid1">
                <label class="form-check-label" for="disk_raid">
                    <strong>Программный RAID-1 (Зеркало):</strong>
Объединить два диска для отказоустойчивости
```

```
                </label>
            </div>

      <!-- ВОТ ЭТОТ БЛОК КРИТИЧЕСКИ НЕОБХОДИМ ДЛЯ ИСПРАВЛЕНИЯ ОШИБКИ 444 -->
            <div id="disks_selection_zone" class="p-3 bg-light border
rounded mb-4" style="display: none;">
                <h6>Доступные физические диски для построения RAID-1
массивов:</h6>
                <div id="disks_list" class="mt-2">
                    <div class="text-muted small">Опрос накопителей
контроллером...</div>
                </div>
                <div class="form-text text-muted mt-2">Для активации
зеркалирования (RAID-1) выберите ровно два одинаковых накопителя.</div>
            </div>

            <div class="alert alert-danger mt-4 d-flex align-items-center">
                <div class="me-3" style="font-size: 24px;">⚠</div>
                <div>
                    <strong>ВНИМАНИЕ:</strong> Все существующие данные
(включая Windows или старые ОС) на выбранных накопителях будут <strong>полностью и
безвозвратно удалены</strong>. Разделы будут переформатированы!
                </div>
            </div>
            <div class="form-check mb-3">
                <input class="form-check-input border-danger"
type="checkbox" id="confirm_erase" onchange="toggleNextButton()">
                <label class="form-check-label text-danger fw-bold"
for="confirm_erase">
          Я понимаю риски. Подтверждаю полное уничтожение данных и форматирование
дисков.
                </label>
            </div>
        </div>

        <!-- 1.2.7 Создание пользователя -->
        <div id="step-7" class="step">
            <div class="step-header"><h4>Шаг 7: Учетная запись
администратора</h4></div>
            <div class="mb-3">
                <label class="form-label">Имя пользователя (Логин):</label>
                <input type="text" id="username" class="form-control"
value="eva" oninput="validateUsername(this)" placeholder="только маленькие
латинские буквы">
                <div class="form-text text-muted">Используйте исключительно
маленькие английские буквы (a-z).</div>
            </div>
            <div class="row mb-3">
                <div class="col">
                    <label class="form-label">Пароль:</label>
                    <input type="password" id="password" class="form-
```

```
control" oninput="this.value = this.value.replace(/[а-яА-ЯёЁ]/g, '');
checkPasswordStrength(this.value)">
                    <div class="form-text text-danger fw-bold" style="font-
size: 13px;">⚠ Ввод строго на АНГЛИЙСКОЙ раскладке! Русский язык запрещен.</div>
                    <div class="progress mt-2" style="height: 6px;">
                        <div id="pass_progress" class="progress-bar bg-
danger" style="width: 0%"></div>
                    </div>
                    <div id="pass_status" class="small text-muted
mt-1">Сложность: слишком слабый</div>
                </div>
                <div class="col">
                    <label class="form-label">Подтверждение пароля:</label>
                    <input type="password" id="password_confirm"
class="form-control" oninput="this.value = this.value.replace(/[а-яА-ЯёЁ]/g,
''); validatePasswordMatch()">
                    <div id="match_status" class="small text-danger
mt-1"></div>
                </div>
            </div>
            <div class="p-3 bg-light border rounded mb-3">
                <h6 class="mb-2" style="font-size: 14px;">Критерии надежности
пароля:</h6>
                <ul class="password-rules-list mb-0">
                    <li id="rule_len" class="text-danger">✗ Не менее 8
symbols</li>
                    <li id="rule_upper" class="text-danger">✗ Минимум одна
заглавная буква (A-Z)</li>
                    <li id="rule_number" class="text-danger">✗ Минимум одна
цифра (0-9)</li>
                    <li id="rule_symbol" class="text-danger">✗ Минимум один
спецсимвол (@, #, $, !)</li>
                </ul>
            </div>
        </div>

    <!-- НОВЫЙ ШАГ 8: Окно Двухфакторной аутентификации (2FA) -->
        <div id="step-8" class="step">
            <div class="step-header"><h4>Шаг 8: Безопасность аккаунта
(2FA)</h4></div>
            <div class="p-3 bg-light border rounded">
                <h5>Настройка Двухфакторной аутентификации</h5>
                <p class="small text-muted mb-2">Отсканируйте этот QR-код
мобильным приложением перед началом развертывания системы:</p>
                <div class="mb-3 small text-secondary">
                    <strong>Поддерживаемые приложения на телефоне:</strong>
                    <ul class="mb-2 mt-1 ps-3">
                        <li>Google Authenticator (iOS / Android)</li>
                        <li>Yandex Key / Яндекс Ключ</li>
                        <li>Microsoft Authenticator</li>
                        <li>2FAS / Aegis / Икс-Ключ</li>
```

```
                    </ul>
                    <span class="text-warning fw-bold">⚠ Внимание:</span>
Сохраните этот аккаунт в приложении на телефоне. Код потребуется при каждом входе на
сервер.
                </div>

                <div id="qrcode_container" class="text-center py-2">
                    <img id="qr-image"
                         src="about:blank"
                         onload="if(this.src=='about:blank') {
this.src='api/2fa.php?username=' +
encodeURIComponent(document.getElementById('username').value) + '&hostname='
+ encodeURIComponent(document.getElementById('sys_hostname').value); }"
                         alt="QR-код для 2FA"
                         class="img-fluid">
        <!-- Временный вывод для тестирования шага 8 -->
                <div id="qr-debug-zone" class="mt-3 p-3 bg-light border
rounded small text-muted text-start" style="font-family: monospace;">
                    <strong>Отладка перед отправкой в API:</strong><br>
          Логин пользователя: <span id="debug-qr-user" class="fw-bold text-
primary">ожидание...</span><br>
          Имя сервера (Host): <span id="debug-qr-host" class="fw-bold text-
primary">ожидание...</span>
                </div>
        <!-- конец Временный вывод для тестирования шага 8 -->
                </div>
            </div>
        </div>

    <!-- НОВЫЙ ШАГ 9: ЭКРАН КОНТРОЛЯ КОНФИГУРАЦИИ -->
        <div id="step-9" class="step">
            <h4 class="mb-3 text-primary">🔍 Шаг 9: Проверка конфигурации
сервера</h4>
            <p class="text-muted small">Внимательно проверьте параметры. После
перехода к следующему шагу диски будут принудительно отформатированы, а данные
перезаписаны!</p>
            <div class="mt-3 text-start">
                <label class="form-label fw-bold text-secondary">Содержимое
файла install_config.txt:</label>
        <!-- Текстовая область, имитирующая файл конфигурации -->
                <textarea id="config-preview-zone"
                          class="form-control bg-light border-primary"
                          rows="9"
                          readonly
                          style="font-family: monospace; font-size: 13px;
color: #2c3e50; font-weight: bold; background-color: #f8f9fa !important;
box-shadow: inset 0 1px 3px rgba(0,0,0,0.1);"></textarea>
            </div>
            <div class="alert alert-warning d-flex align-items-center mt-3
small" role="alert">
                <div>
```

```
                    ⚠ <strong>Внимание:</strong> Нажатие кнопки «Далее»
безвозвратно уничтожит текущие таблицы разделов на целевых накопителях.
                </div>
            </div>
        </div>




    <!-- ШАГ 10: ПРОЦЕСС УСТАНОВКИ С ВНУТРЕННЕЙ КОНСОЛЬЮ -->
        <div id="step-10" class="step">
            <div class="step-header"><h4>Шаг 10: Выполнение
установки</h4></div>
            <p id="install_status" class="fw-bold text-
primary">Инициализация скриптов разметки дисковых массивов...</p>
            <div class="progress mb-3" style="height: 25px;">
                <div id="install_progress" class="progress-bar progress-bar-
striped progress-bar-animated" role="progressbar" style="width:
0%;">0%</div>
            </div>
            <div class="text-muted small">Время выполнения операции: <span
id="install_timer" class="fw-bold text-dark">0 сек.</span></div>

      <!-- Консоль теперь живет строго ЗДЕСЬ и будет появляться только на этом шаге -->
            <div class="mt-4 p-3 bg-dark text-success rounded small text-
start" id="installer-console" style="font-family: monospace; min-height:
120px; font-size: 12px; line-height: 1.4; opacity: 0.95; box-shadow: inset 0
0 10px #000;">
                <div class="text-muted border-bottom border-secondary pb-1
mb-2">▶ СИСТЕМНЫЙ ЖУРНАЛ УСТАНОВКИ:</div>
                <div id="console-output-lines">Ожидание потока
данных...</div>
            </div>
        </div>




    <!-- Теперь это ШАГ 11: Завершение и перезагрузка -->
        <div id="step-11" class="step">
            <div class="step-header"><h4 class="text-success">Установка
успешно завершена!</h4></div>
            <p>Операционная система развернута. Сетевые сервисы, SSH-доступ и
конфигурация RAID подготовлены.</p>
      <!-- Сюда JS выведет надпись об обновлении страницы -->
            <div id="install_status_final" class="my-3"></div>

            <div class="alert alert-warning text-center p-4">
                <h5>Внимание! Сервер будет отправлен в перезагрузку через:</h5>
                <h1 id="reboot_timer" class="display-3 font-monospace my-3
text-danger fw-bold">5</h1>
                <p class="mb-0 small fw-bold">Извлеките загрузочную USB-
```

```
флешку из разъема, чтобы сервер загрузился с основного диска.</p>
            </div>
        </div>




    </div>

  <!-- Кнопки управления шагами -->
    <div class="buttons d-flex justify-content-between mt-4 border-top
pt-3">
        <button id="btn-prev" class="btn btn-outline-secondary px-4"
onclick="changeStep(-1)" disabled>Назад</button>
        <button id="btn-next" class="btn btn-primary px-4"
onclick="changeStep(1)">Далее</button>
    </div>
</div>

<script>
let currentStep = 1;
const totalSteps = 11;

// Флаги валидации для учетной записи
let isPasswordValid = false;
let isPasswordMatching = false;

// Контроль блокировки кнопки "Далее" на Шаге 6 (Диски)
function toggleNextButton() {
    if (currentStep === 6) {
        const confirmed = document.getElementById('confirm_erase').checked;
        const diskMode =
document.querySelector('input[name="disk_mode"]:checked').value;
        let isDiskSelectionOk = true;

        if (diskMode === 'raid1') {
            const selectedDisksCount = document.querySelectorAll('.disk-
checkbox:checked').length;
            isDiskSelectionOk = (selectedDisksCount === 2); // Строго 2 диска
для RAID-1
        }
        document.getElementById('btn-next').disabled = !(confirmed &&
isDiskSelectionOk);
    }
}

function changeStep(direction) {
  // Скрываем блок ошибок перед проверкой
    const errorBlock = document.getElementById('error-message-zone');
    if (errorBlock) errorBlock.style.display = 'none';
```

```
  // АВТО-ОЧИСТКА СЕРВЕРА: Если пользователь возвращается НАЗАД с Шага 9
    if (direction === -1 && currentStep === 9) {
    // Блокируем интерфейс на долю секунды для выполнения зачистки
        document.getElementById('btn-prev').disabled = true;
        document.getElementById('btn-next').disabled = true;

        fetch('api/cancel_install.php')
        .then(res => res.json())
        .then(result => {
      // Разблокируем кнопки после успешной очистки сервера
            document.getElementById('btn-prev').disabled = false;
            document.getElementById('btn-next').disabled = false;
        })
        .catch(err => {
            console.log("Ошибка фоновой очистки: ", err);
            document.getElementById('btn-prev').disabled = false;
            document.getElementById('btn-next').disabled = false;
        });
    }

  // Подгрузка доступных дисков при переходе с 5 на 6 шаг
    if (currentStep === 5 && direction === 1) {
        loadSystemDisks();
    }

  // ТРИГГЕР: Переход с 7 на 8 шаг (Безопасное обновление QR)
    if (direction === 1 && currentStep === 7) {
        if (!isPasswordValid || !isPasswordMatching) {
            showInlineError('Пароль не соответствует требованиям или не
совпадает!');
            return;
        }

    // Забираем данные и обновляем QR-код
        try {
            const currentHost =
document.getElementById('sys_hostname').value.trim();
            const currentUser =
document.getElementById('username').value.trim();

            const dbgUser = document.getElementById('debug-qr-user');
            const dbgHost = document.getElementById('debug-qr-host');
            if (dbgUser) dbgUser.innerText = currentUser;
            if (dbgHost) dbgHost.innerText = currentHost;

      // ИСПРАВЛЕНО: Точный поиск картинки по ID
            const qrImg = document.getElementById('qr-image');
            if (qrImg) {
                qrImg.src =
`api/2fa.php?username=${encodeURIComponent(currentUser)}&hostname=${encodeUR
IComponent(currentHost)}`;
```

```
            }
        } catch (e) {
            console.log("Ошибка обновления элементов Шага 8: ", e);
        }
    }

  // ТРИГГЕР: Нажатие "Далее" на Шаге 8 (Сборка превью install_config.txt без
отправки на сервер)
    if (direction === 1 && currentStep === 8) {
        let selectedDisks = [];
        const diskMode =
document.querySelector('input[name="disk_mode"]:checked').value;
        if (diskMode === 'raid1') {
            document.querySelectorAll('.disk-checkbox:checked').forEach(cb
=> {
                selectedDisks.push(cb.value);
            });
        } else {
            const firstDiskInput = document.querySelector('.disk-checkbox');
      // ИСПРАВЛЕНО: Убран хардкод sda. Если чекбокс найден - берем его value, иначе
оставляем пустоту для валидации
            selectedDisks.push(firstDiskInput ? firstDiskInput.value : '');
        }

    // Формируем точный текст будущего конфигурационного файла для вывода
пользователю
        let configPreview =
`SYS_LANG=${document.getElementById('sys_lang').value}\n`;
        configPreview +=
`SYS_LAYOUT=${document.getElementById('sys_layout').value}\n`;
        configPreview +=
`SYS_TIMEZONE=${document.getElementById('sys_timezone').value}\n`;
        configPreview +=
`SYS_HOSTNAME=${document.getElementById('sys_hostname').value.trim()}\n`;
        configPreview += `DISK_MODE=${diskMode}\n`;
        configPreview += `SELECTED_DISKS=${selectedDisks.join(' ')}\n`;
        configPreview +=
`SYS_USER=${document.getElementById('username').value.trim()}\n`;
        configPreview +=
`SYS_PASS=${document.getElementById('password').value.trim()}`;

    // Безопасно выводим сформированный текст в зону контроля нового Шага 9
        const previewZone = document.getElementById('config-preview-zone');
        if (previewZone) {
            previewZone.value = configPreview;
        }

    // Переключаем интерфейс со старого Шага 8 на новый Шаг 9
        document.getElementById(`step-
${currentStep}`).classList.remove('active');
        currentStep = 9;
```

```
        document.getElementById(`step-9`).classList.add('active');
    // Кнопки навигации остаются активными, так как это экран контроля, а не установка
        document.getElementById('btn-prev').disabled = false;
        document.getElementById('btn-next').disabled = false;
        return; // Полностью прерываем выполнение, чтобы не сработал линейный
переход ниже
    }

  // ТРИГГЕР: Нажатие "Далее" на НОВОМ Шаге 9 (Фактический СТАРТ установки и
форматирования)
    if (direction === 1 && currentStep === 9) {
        let selectedDisks = [];
        const diskMode =
document.querySelector('input[name="disk_mode"]:checked').value;
        if (diskMode === 'raid1') {
            document.querySelectorAll('.disk-checkbox:checked').forEach(cb
=> {
                selectedDisks.push(cb.value);
            });
        } else {
            const firstDiskInput = document.querySelector('.disk-checkbox');
            selectedDisks.push(firstDiskInput ? firstDiskInput.value :
'sda');
        }

        const payload = {
            lang: document.getElementById('sys_lang').value,
            layout: document.getElementById('sys_layout').value,
            timezone: document.getElementById('sys_timezone').value,
            hostname: document.getElementById('sys_hostname').value.trim(),
            disk_mode: diskMode,
            disks: selectedDisks,
            username: document.getElementById('username').value.trim(),
            password: document.getElementById('password').value.trim()
        };

    // Блокируем кнопку на время отправки, чтобы избежать повторных кликов
        document.getElementById('btn-next').disabled = true;

    // ПРИНУДИТЕЛЬНЫЙ ЗАПУСК ОТПРАВКИ НА СЕРВЕР
        fetch('api/start_install.php', {
            method: 'POST',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify(payload)
        })
        .then(res => res.json())
        .then(result => {
            if (result && result.success) {
        // Если сервер принял данные, гасим Шаг 9 и открываем Шаг 10 (Прогресс-бар)
document.getElementById('step-9').classList.remove('active');
                currentStep = 10;
```

```
                document.getElementById('step-10').classList.add('active');
        // Намертво скрываем нижние кнопки навигации на время установки
                document.getElementById('btn-prev').style.display = 'none';
                document.getElementById('btn-next').style.display = 'none';
        // Запускаем AJAX-опрос системного журнала установки
                startInstallationSimulation();
            } else {
                document.getElementById('btn-next').disabled = false;
                showInlineError('Ошибка сервера: ' + (result &&
result.message ? result.message : 'Неизвестный ответ'));
            }
        })
        .catch(err => {
            document.getElementById('btn-next').disabled = false;
            showInlineError('Ошибка сети: сервер установки недоступен.');
        });
        return; // Блокируем стандартный линейный переход, ждем ответа от сервера
    }


  // Линейный переход для остальных шагов
    document.getElementById(`step-
${currentStep}`).classList.remove('active');
    currentStep += direction;
    document.getElementById(`step-${currentStep}`).classList.add('active');

    document.getElementById('btn-prev').disabled = (currentStep === 1 ||
currentStep >= 10);
    document.getElementById('btn-next').disabled = (currentStep >= 10);

    if (currentStep === 6) {
        toggleNextButton();
    }
}




// Валидаторы ввода с ограничением до 16 символов
function validateHostname(input) {
    input.value = input.value.toLowerCase().replace(/[^a-z0-9-]/g,
'').substring(0, 16);
}

// Изменено: разрешаем латиницу и цифры для соответствия бэкенду
function validateUsername(input) {
    input.value = input.value.toLowerCase().replace(/[^a-z0-9]/g,
'').substring(0, 16);
}
```

```
// Запрет пробелов и ограничение длины в самом поле пароля
const passwordInputField = document.getElementById('password');
if (passwordInputField) {
    passwordInputField.addEventListener('input', function() {
        this.value = this.value.replace(/\s+/g, '').substring(0, 16);
    });
  // Беззвучный запрет вставки
    ['paste', 'drop'].forEach(e => {
        passwordInputField.addEventListener(e, (ev) => ev.preventDefault());
    });
}

// Проверка силы пароля
function checkPasswordStrength(password) {
    const rules = {
        len: password.length >= 8,
        upper: /[A-Z]/.test(password),
        number: /[0-9]/.test(password),
        symbol: /[@$!%*?&#]/.test(password)
    };

    updateRuleVisual('rule_len', rules.len, "Не менее 8 символов");
    updateRuleVisual('rule_upper', rules.upper, "Минимум одна заглавная буква
(A-Z)");
    updateRuleVisual('rule_number', rules.number, "Минимум одна цифра (0-9)");
    updateRuleVisual('rule_symbol', rules.symbol, "Минимум один спецсимвол (@,
#, $, !)");

    let score = Object.values(rules).filter(Boolean).length;
    const progress = document.getElementById('pass_progress');
    const status = document.getElementById('pass_status');
    if (progress) {
        progress.className = "progress-bar ";
        if (score <= 1) {
            progress.style.width = "25%"; progress.classList.add("bg-
danger");
            if (status) status.innerText = "Сложность: слишком слабый";
isPasswordValid = false;
        } else if (score === 2) {
            progress.style.width = "50%"; progress.classList.add("bg-
warning");
            if (status) status.innerText = "Сложность: средний";
isPasswordValid = false;
        } else if (score === 3) {
            progress.style.width = "75%"; progress.classList.add("bg-info");
            if (status) status.innerText = "Сложность: хороший";
isPasswordValid = false;
        } else if (score === 4) {
            progress.style.width = "100%"; progress.classList.add("bg-
success");
            if (status) status.innerText = "Сложность: отличный (безопасный)";
```

```
isPasswordValid = true;
        }
    }
    validatePasswordMatch();
}

function updateRuleVisual(elementId, isValid, text) {
    const el = document.getElementById(elementId);
    if (el) {
        el.innerText = (isValid ? "□ " : "□ ") + text;
        el.className = isValid ? "text-success fw-bold" : "text-danger";
    }
}

function validatePasswordMatch() {
    const passInput = document.getElementById('password');
    const confirmInput = document.getElementById('password_confirm');
    const matchStatus = document.getElementById('match_status');

    if (!passInput || !confirmInput || !matchStatus) return;

    const pass = passInput.value;
    const confirm = confirmInput.value;

    if (!confirm) { matchStatus.innerText = ""; isPasswordMatching = false;
return; }

    if (pass === confirm) {
        matchStatus.innerText = "□ Пароли совпадают";
        matchStatus.className = "small text-success fw-bold";
        isPasswordMatching = true;
    } else {
        matchStatus.innerText = "□ Пароли не совпадают";
        matchStatus.className = "small text-danger";
        isPasswordMatching = false;
    }
}

// Мониторинг лога бэкенда и управление Шагом 10
function startInstallationSimulation() {
    const progressBar = document.getElementById('install_progress');
    const statusText = document.getElementById('install_status');
    const timerText = document.getElementById('install_timer');
    let elapsedSeconds = 0;

  // Счётчик времени выполнения операции
    const durationTimer = setInterval(() => {
        elapsedSeconds++;
        if (timerText) timerText.innerText = elapsedSeconds + ' сек.';
    }, 1000);
```

```
  // Ежесекундный  AJAX-опрос системного журнала установки
    const logWatcher = setInterval(() => {
        fetch('api/get_log.php')
        .then(res => res.json())
        .then(data => {
            if (data && data.success) {
        // Обновляем визуальный прогресс-бар
                if (progressBar) {
                    progressBar.style.width = data.progress + '%';
                    progressBar.innerText = data.progress + '%';
                }
        // Обновляем текущий статус над шкалой
                if (statusText) statusText.innerText = data.status;

        // ВЫВОД ПОЛНОЙ ИСТОРИИ ЛОГА В ИНТЕРАКТИВНУЮ ВЕБ-КОНСОЛЬ:
                const consoleZone = document.getElementById('console-output-
lines');
                const consoleWrapper = document.getElementById('installer-
console');
                if (consoleZone && data.console && data.console.length > 0)
{
          // Объединяем полученные строки через перенос строки
                    consoleZone.innerHTML = data.console.join('<br>');
          // Безопасная автоматическая прокрутка терминала вниз
                    if (consoleWrapper) {
                        consoleWrapper.scrollTop =
consoleWrapper.scrollHeight;
                    }
                }

        // Переход с Шага 10 на Финальный Шаг 11 при достижении 90% (С ЗАДЕРЖКОЙ
ДЛЯ ИНСПЕКЦИИ)
                if (data.progress >= 90) {
          // 1. Мгновенно останавливаем опрос логов, чтобы зафиксировать финальный
текст на экране
                    clearInterval(logWatcher);
                    clearInterval(durationTimer);
          // Устанавливаем прогресс-бар на ровные 100% для красоты финала
                    const progressBar =
document.getElementById('install_progress_bar');
                    if (progressBar) {
                        progressBar.style.width = '100%';
                        progressBar.setAttribute('aria-valuenow', 100);
                        progressBar.innerText = '100%';
                    }

                    console.log("Установка завершена. Запуск 5-секундной паузы для
чтения логов перед ребутом...");

          // 2. Включаем задержку на 5000 мс (5 секунд) перед переключением экранов
                    setTimeout(function() {
```

```
document.getElementById('step-10').classList.remove('active');
                        currentStep = 11;
document.getElementById('step-11').classList.add('active');
            // Запускаем финальный обратный отсчет самой перезагрузки
                        startRebootCountdown();
          }, 5000); // 5000 миллисекунд = 5 секунд
                }

            }
        })
        .catch(err => {
            console.log("Внутренняя заминка опроса лога: ", err);
        });
    }, 1500);
}



// Вспомогательная функция вывода ошибок на экран (Привязка к контейнеру .buttons)
function showInlineError(message) {
    let errorZone = document.getElementById('error-message-zone');
    if (!errorZone) {
        errorZone = document.createElement('div');
        errorZone.id = 'error-message-zone';
        errorZone.className = 'alert alert-danger my-2 text-center';
        const buttonsContainer = document.querySelector('.buttons');
        const prevBtn = document.getElementById('btn-prev');
        if (buttonsContainer && prevBtn) {
            buttonsContainer.insertBefore(errorZone, prevBtn);
        } else {
            const card = document.querySelector('.wizard-card') ||
document.body;
            card.appendChild(errorZone);
        }
    }
    errorZone.innerText = message;
    errorZone.style.display = 'block';
}

function loadSystemDisks() {
    const disksList = document.getElementById('disks_list');
    if (!disksList) return;
    disksList.innerHTML = '<div class="spinner-border spinner-border-sm
text-primary"></div> Опрашиваю дисковую подсистему...';
    fetch('api/get_disks.php')
    .then(response => response.json())
    .then(data => {
        if (data && data.success && data.disks.length > 0) {
            disksList.innerHTML = '';
            data.disks.forEach(disk => {
                const diskDiv = document.createElement('div');
```

```
                diskDiv.className = 'form-check mb-2';
        // Обновленная верстка: выводим имя, объем и МОДЕЛЬ накопителя
                diskDiv.innerHTML = `
                    <input class="form-check-input disk-checkbox"
type="checkbox" value="${disk.name}" id="disk_${disk.name}">
                    <label class="form-check-label" for="disk_${disk.name}">
                        💽 <strong class="text-
dark">/dev/${disk.name}</strong> —
            Размер: <span class="badge bg-secondary">${disk.size}</span> —
            Модель: <span class="fw-bold text-primary">${disk.model}</span>
                    </label>
                `;
                disksList.appendChild(diskDiv);
            });
        } else {
            disksList.innerHTML = '<div class="text-danger">⚠ Накопители для
установки не найдены!</div>';
        }
    })
    .catch(error => {
        disksList.innerHTML = '<div class="text-danger">⚠ Ошибка бэкенда API
при опросе дисков!</div>';
    });
}


// Логика переключения режимов разметки дисков
document.addEventListener('change', function(e) {
    if (e.target && e.target.name === 'disk_mode') {
        const zone = document.getElementById('disks_selection_zone');
        if (zone) zone.style.display = (e.target.value === 'raid1') ?
'block' : 'none';
    }
});

document.addEventListener('click', function(e) {
    if (e.target && e.target.classList.contains('disk-checkbox')) {
        toggleNextButton();
    }
});

// Защита от возврата кнопкой Back браузера
history.pushState(null, null, location.href);
window.addEventListener('popstate', function () {
    history.pushState(null, null, location.href);
    showInlineError('Для навигации используйте кнопки мастера установки.');
});
</script>


</body>
```

```
</html>
```

```
=== 2fa.php ===
<?php
// 1. Полностью отключаем дисковое кэширование библиотеки QR
define('QR_CACHEABLE', false);

ob_start();

// Жестко привязываем абсолютный путь к qrlib.php через __DIR__
$lib_path = __DIR__ . '/qrlib.php';

if (!file_exists($lib_path)) {
    ob_end_clean();
    header('HTTP/1.1 500 Internal Server Error');
    header('Content-Type: application/json');
    echo json_encode(['error' => 'Файл qrlib.php не найден в каталоге api/']);
    exit;
}

// Подключаем библиотеку
include_once $lib_path;

// 2. ПРИНИМАЕМ ПАРАМЕТРЫ ИЗ ФОРМЫ (Если пусто — ставим заводские значения)
$company = isset($_GET['hostname']) && !empty($_GET['hostname']) ?
trim($_GET['hostname']) : "Arch-Server";
$user    = isset($_GET['username']) && !empty($_GET['username']) ?
trim($_GET['username']) : "eva";
$secret  = 'SECRETKEY1234567'; // Ваш мастер-ключ

// Формируем ссылку по международному стандарту TOTP с динамическими данными
$otpauth_url = "otpauth://totp/" . rawurlencode($company) . ":" .
rawurlencode($user) . "?secret=" . $secret . "&issuer=" .
rawurlencode($company);

// Стираем любые предупреждения или случайные пробелы до этого момента
if (ob_get_length()) ob_end_clean();

// Отдаем чистый заголовок картинки
header('Content-Type: image/png');
header('Cache-Control: no-store, no-cache, must-revalidate, max-age=0');
header('Pragma: no-cache');

// 3. ГЕНЕРИРУЕМ QR-КОД НАПРЯМУЮ В ПАМЯТЬ (БЕЗ КЭША НА ДИСКЕ)
// Пятый параметр (размер пикселя) = 5, шестой параметр (размер рамки) = 2, седьмой (кэш)
= false
QRcode::png($otpauth_url, false, QR_ECLEVEL_L, 5, 2, false);
exit;
```

```
=== cancel_install.php ===
```

```
<?php
// Скрипт принудительной очистки при возврате назад (api/cancel_install.php)
header('Content-Type: application/json');
ini_set('display_errors', 0);

$config_file = __DIR__ . '/install_config.txt';
$log_file = '/tmp/install.log';

// 1. Физически удаляем файл конфигурации, если он успел записаться
if (file_exists($config_file)) {
    unlink($config_file);
}

// 2. Стираем лог, чтобы get_log.php обнулился
if (file_exists($log_file)) {
    unlink($log_file);
}

// 3. Мягко завершаем фоновые процессы, если они были инициализированы
exec('sudo killall -9 disk_prepare.sh system_install.sh rsync tar
2>/dev/null');

// 4. Размонтируем диски, если они остались заблокированы в системе
exec('sudo umount -R /mnt 2>/dev/null');

echo json_encode(['success' => true, 'message' => 'Стенд успешно очищен при
возврате назад']);
exit;
```

```

=== chroot_configure.sh ===
#!/bin/bash
# Итоговый эталонный скрипт настройки под UEFI (api/chroot_configure.sh)

# Импортируем переданные переменные из временного файла конфигурации
if [ -f /etc/installer_env.conf ]; then
    source /etc/installer_env.conf
fi

# 1. Сетевое имя хоста
if [ -z "$SYS_HOSTNAME" ]; then SYS_HOSTNAME="arch-server-btrfs"; fi
echo "$SYS_HOSTNAME" > /etc/hostname

cat <<EOF > /etc/hosts
127.0.0.1   localhost
::1         localhost
127.1.1.1   $SYS_HOSTNAME.localdomain $SYS_HOSTNAME
EOF

# 2. Часовой пояс
/usr/bin/ln -sf /usr/share/zoneinfo/Europe/Moscow /etc/localtime
```

```
/usr/bin/hwclock --systohc

# ========================================================================
# 3. Настройка учетной записи суперпользователя root (ЗАЩИЩЕННАЯ)
# ========================================================================
if [ -z "$SYS_PASS" ]; then SYS_PASS="12345678"; fi

# Используем стандартный потоковый passwd, который никогда не роняет скрипт
echo -e "$SYS_PASS\n$SYS_PASS" | /usr/bin/passwd root >/dev/null 2>&1

# ========================================================================
# 4. Пользователь и права администратора (ЗАЩИЩЕННАЯ)
# ========================================================================
if [ -z "$SYS_USER" ]; then SYS_USER="eva"; fi
if ! id "$SYS_USER" &>/dev/null; then
    /usr/bin/useradd -m -G wheel,storage,power,http -s /bin/bash "$SYS_USER"
fi

# Назначаем пароль пользователю через надежный механизм
echo -e "$SYS_PASS\n$SYS_PASS" | /usr/bin/passwd "$SYS_USER" >/dev/null 2>&1

echo "%wheel ALL=(ALL:ALL) NOPASSWD: ALL" > /etc/sudoers.d/10_wheel
chmod 440 /etc/sudoers.d/10_wheel
```

```
=== disk_prepare.sh ===
#!/bin/bash
# Финальный эталонный скрипт разметки UEFI (api/disk_prepare.sh)
cd "$(dirname "$0")"
CONFIG_FILE="install_config.txt"
LOG_FILE="/tmp/install.log"

# Безопасная функция записи, обходящая конфликты прав http/root
log_msg() {
    echo "$1" | /usr/bin/tee -a "$LOG_FILE" > /dev/null
    echo "$1"
}

# ========================================================================
# 1. Принудительное удаление логов и сброс монтирований (Авто-очистка буфера)
# ========================================================================

# полностью освобождая виртуальный dev-буфер для нового прогона
sudo /usr/bin/umount -l /mnt/dev/pts 2>/dev/null
sudo /usr/bin/umount -l /mnt/dev 2>/dev/null
sudo /usr/bin/umount -l /mnt/proc 2>/dev/null
sudo /usr/bin/umount -l /mnt/sys 2>/dev/null
sudo /usr/bin/umount -l /mnt/run 2>/dev/null

# чтобы mkfs.vfat никогда не спотыкался о занятые дескрипторы
sudo /usr/bin/umount -R /mnt/boot/efi 2>/dev/null
```

```
# предотвращая аппаратную блокировку "Device or resource busy"
sudo /usr/bin/umount -R /mnt 2>/dev/null
sudo /usr/bin/umount -R /tmp/recovery_pool 2>/dev/null

# полностью освобождая диски sdb и sdc от фоновой блокировки ядра
sudo /usr/bin/btrfs device scan --forget 2>/dev/null

# Если файл лога существовал, мгновенно стираем его, обходя любые конфликты прав
sudo /usr/bin/rm -f "$LOG_FILE"

# Создаем абсолютно новый, девственно чистый файл лога
echo "[PROGRESS] 0" > "$LOG_FILE"

# Сразу даем ему права 666, чтобы в него могли дописывать строки и http, и root (через
sudo)
/usr/bin/chmod 666 "$LOG_FILE"

# Пишем первый синий маркер старта
echo "[INFO] Инициализация разметки дисковых массивов Btrfs..." >> "$LOG_FILE"
# ====================================================================



if [ ! -f "$CONFIG_FILE" ]; then
    log_msg "[ERROR] Конфигурация install_config.txt не найдена!"
    exit 1
fi

# ====================================================================
# 2. Чтение конфигурации с ЖЕСТКОЙ зачисткой от символов \r (Windows)
# ====================================================================
set -a
# Команда tr -d '\r' полностью вычищает скрытые переносы строк Windows
source <(tr -d '\r' < "$CONFIG_FILE")
set +a

# 3. Безопасное уничтожение файла конфигурации на диске
# Сохраняем бэкап конфига для system_install.sh перед удалением оригинала
cp "$CONFIG_FILE" "install_config.txt.bak" 2>/dev/null
/usr/bin/shred -u "$CONFIG_FILE"
log_msg "[INFO] Конфигурация импортирована, install_config.txt безвозвратно
уничтожен."

# ====================================================================
# 4. Сбор дисков из памяти
# ====================================================================
log_msg "[PROGRESS] 10"
if [ -z "$SELECTED_DISKS" ]; then
    SELECTED_DISKS="sdb sdc"
fi
IFS=' ' read -r -a DISKS_ARRAY <<< "$SELECTED_DISKS"
```

```
# ========================================================================
# НОВОЕ: Динамический сброс таблиц разделов ядра для выбранных дисков
# ========================================================================
# Этот цикл заставит ядро Linux принудительно перечитать структуру именно
# тех накопителей, которые выбрал пользователь, полностью исключая "Device or
resource busy"
for raw_disk in "${DISKS_ARRAY[@]}"; do
    disk=$(echo "$raw_disk" | tr -d '\r')
    sudo /usr/bin/blockdev --rereadpt "/dev/$disk" 2>/dev/null
done

# ========================================================================
# 5. Принудительная зачистка (Wipe Out) сигнатур прошлых ФС через sudo
# ========================================================================
log_msg "[INFO] Очистка старых таблиц разделов и сигнатур..."
for disk in "${DISKS_ARRAY[@]}"; do
    sudo /usr/bin/wipefs -a -f "/dev/$disk" | /usr/bin/tee -a "$LOG_FILE" >
/dev/null 2>&1
done

log_msg "[PROGRESS] 20"
log_msg "[INFO] Создание новой разметки GPT и UEFI-разделов через fdisk..."

# ========================================================================
# 6. Автоматическая нарезка таблиц GPT через fdisk (Чистый синтаксис)
# ========================================================================
log_msg "[PROGRESS] 30"
for raw_disk in "${DISKS_ARRAY[@]}"; do
  # Намертво вычищаем фантомные переносы строк из имени каждого диска
    disk=$(echo "$raw_disk" | tr -d '\r')
    sudo /usr/bin/fdisk "/dev/$disk" <<EOF >> "$LOG_FILE" 2>&1
g
n
1

+512M
t
1
n
2



w
EOF
done

# ИСПРАВЛЕНО: Явное извлечение и очистка имени первого диска для форматирования EFI
CLEAN_FIRST_DISK=$(echo "${DISKS_ARRAY[0]}" | tr -d '\r')

# Форматируем созданный первый раздел в FAT32 (UEFI) через стабильный mkfs.vfat
log_msg "[INFO] Форматируем созданный первый раздел в FAT32 (UEFI) ..."
```

```
sudo /usr/bin/mkfs.vfat -F 32 "/dev/${CLEAN_FIRST_DISK}1" >> "$LOG_FILE"
2>&1

# ============================================================================
# 7. Форматирование ВТОРЫХ РАЗДЕЛОВ в пул Btrfs RAID-1 (БЕЗОПАСНО)
# ============================================================================
log_msg "[PROGRESS] 40"
# Используем разделы (sdb2, sdc2), а не диски целиком (sdb, sdc)
TARGET_PARTITIONS=$(printf " /dev/%s2" "${DISKS_ARRAY[@]}")

if [ "$DISK_MODE" == "raid1" ]; then
    sudo /usr/bin/mkfs.btrfs -f -d raid1 -m raid1 $TARGET_PARTITIONS |
/usr/bin/tee -a "$LOG_FILE" > /dev/null 2>&1
else
    sudo /usr/bin/mkfs.btrfs -f $TARGET_PARTITIONS | /usr/bin/tee -a
"$LOG_FILE" > /dev/null 2>&1
fi

# 8. Передача эстафеты следующему этапу
log_msg "[INFO] Дисковая структура готова. Переход к клонированию системы..."
echo "[PROGRESS] 50" | /usr/bin/tee -a "$LOG_FILE" > /dev/null

# ============================================================================
# □ ИНСПЕКЦИЯ ЭТАПА РАЗМЕТКИ И ФОРМАТИРОВАНИЯ (ФИЗИЧЕСКИЙ КОНТРОЛЬ)
# ============================================================================
log_msg "[DIAGNOSTIC] === СТАРТ ФИЗИЧЕСКОЙ ПРОВЕРКИ ТАБЛИЦЫ РАЗДЕЛОВ ==="

# 1. Выводим реальный статус физических разделов sdb и sdc из ядра Linux
sudo /usr/bin/lsblk -o NAME,FSTYPE,SIZE,UUID /dev/sdb >> "$LOG_FILE" 2>&1
sudo /usr/bin/lsblk -o NAME,FSTYPE,SIZE,UUID /dev/sdc >> "$LOG_FILE" 2>&1

# 2. Временно монтируем чистый корень sdb2 наружу для проверки структуры
sudo /usr/bin/mkdir -p /tmp/diag_btrfs
sudo /usr/bin/mount /dev/sdb2 /tmp/diag_btrfs 2>/dev/null

log_msg "[DIAGNOSTIC] Проверка созданных подтомов Btrfs на реальном диске sdb2:"
sudo /usr/bin/btrfs subvolume list /tmp/diag_btrfs >> "$LOG_FILE" 2>&1

sudo /usr/bin/umount /tmp/diag_btrfs 2>/dev/null
sudo /usr/bin/rmdir /tmp/diag_btrfs

log_msg "[DIAGNOSTIC] === КОНЕЦ ФИЗИЧЕСКОЙ ПРОВЕРКИ РАЗМЕТКИ ==="
# ============================================================================


sudo -E /srv/http/installer/api/system_install.sh
```

```
=== get_disks.php ===
<?php
```

```
// Автоматический опрос дисков с выводом модели устройства (api/get_disks.php)
header('Content-Type: application/json');
ini_set('display_errors', 0);

// 1. Автоматически определяем имя установочной флешки
$usb_drive = '';
$boot_mount = shell_exec("findmnt -n -o SOURCE /run/archiso/bootmnt
2>/dev/null");
if (empty($boot_mount)) {
    $boot_mount = shell_exec("df / | tail -n 1 | awk '{print $1}'");
}
if (!empty($boot_mount)) {
    $usb_drive = preg_replace('/[0-9]+/', '', basename(trim($boot_mount)));
}

// 2. Автоматически определяем диск текущей запущенной ОС (Ubuntu / Донор)
$current_os_drive = '';
$root_mount = shell_exec("findmnt -n -o SOURCE / 2>/dev/null");
if (!empty($root_mount)) {
    $current_os_drive = preg_replace('/[0-9]+/', '',
basename(trim($root_mount)));
}

// 3. Получаем список физических дисков
$sys_blocks = glob('/sys/block/sd*');
if (empty($sys_blocks)) {
    $sys_blocks = glob('/sys/block/nvme*');
}

$disks = [];

// [Этот блок находится внутри api/get_disks.php]
foreach ($sys_blocks as $block) {
    $disk_name = basename($block);
  // СТРОГИЙ ФИЛЬТР: Скрываем ТОЛЬКО и ИСКЛЮЧИТЕЛЬНО установочную флешку
  // Все остальные диски (даже со старыми Linux/Windows) обязаны отображаться!
    if ($disk_name === $usb_drive) {
        continue;
    }
  // Пропускаем виртуальные loop-устройства и CD-ROM
    if (strpos($disk_name, 'loop') === 0 || strpos($disk_name, 'sr') === 0)
{
        continue;
    }

  // Читаем размер диска
    $size_sectors = (float)trim(file_get_contents("$block/size"));
    $size_gb = round(($size_sectors * 512) / (1024 * 1024 * 1024), 1);
  // Игнорируем слишком маленькие накопители (меньше 2 ГБ)
    if ($size_gb < 2) {
        continue;
```

```
    }

  // Читаем модель диска напрямую из sysfs ядра Linux
    $model_path = "$block/device/model";
    $disk_model = "Unknown Storage Device";
    if (file_exists($model_path)) {
        $disk_model = trim(file_get_contents($model_path));
        $disk_model = preg_replace('/\s+/', ' ', $disk_model);
    }

    $disks[] = [
        'name'  => $disk_name,
        'size'  => $size_gb . ' GB',
        'model' => $disk_model
    ];
}


echo json_encode([
    'success' => true,
    'disks' => $disks
], JSON_UNESCAPED_UNICODE);
exit;
```

```
=== get_log.php ===
<?php
// Полный сбор истории логов (api/get_log.php)
header('Content-Type: application/json');
ini_set('display_errors', 0);

$log_path = '/tmp/install.log';

if (!file_exists($log_path)) {
    echo json_encode([
        'success' => true,
        'progress' => 0,
        'status' => 'Запуск фонового процесса разметки дисков...',
        'console' => ['Ожидание инициализации логирования...']
    ], JSON_UNESCAPED_UNICODE);
    exit;
}

$lines = file($log_path, FILE_IGNORE_NEW_LINES | FILE_SKIP_EMPTY_LINES);
$progress = 0;
$status = 'Развертывание системы...';
$clean_lines = [];

// Парсим лог целиком
foreach ($lines as $line) {
    if (strpos($line, '[PROGRESS]') === 0) {
```

```
        $progress = (int)trim(str_replace('[PROGRESS]', '', $line));
    } elseif (strpos($line, '[INFO]') === 0) {
        $status = trim(str_replace('[INFO]', '', $line));
        $clean_lines[] = "□ " . $status;
    } elseif (strpos($line, '[ERROR]') === 0) {
        $status = trim(str_replace('[ERROR]', '', $line));
        $clean_lines[] = "□ " . $status;
    } else {
    // Все остальные строки (вывод rsync, mkfs, wipefs) добавляем как есть
        $clean_lines[] = $line;
    }
}

echo json_encode([
    'success' => true,
    'progress' => $progress,
    'status' => $status,
    'console' => $clean_lines // Отдаем ВСЮ историю строк лога
], JSON_UNESCAPED_UNICODE);
exit;
```

```
=== start_install.php ===
<?php
header('Content-Type: application/json');
ini_set('display_errors', 0);

$inputData = file_get_contents('php://input');
$data = json_decode($inputData, true);

if (!$data) {
    echo json_encode(['success' => false, 'message' => 'Пустой запрос']);
    exit;
}

// Жесткая очистка строк
$clean_hostname = preg_replace('/[^a-zA-Z0-9_\-\.]/', '',
substr(trim($data['hostname']), 0, 16));
$clean_username = preg_replace('/[^a-zA-Z0-9_\-\.]/', '',
substr(trim($data['username']), 0, 16));
$clean_password = preg_replace('/[^a-zA-Z0-9_\-\.]/', '',
substr(trim($data['password']), 0, 16));

// Пишем конфиг с нуля (всегда новые данные)
// [Этот блок находится внутри api/start_install.php]
// Формируем чистые строки конфигурации для Bash без скрытых символов \r
$configContent = "SYS_LANG=" . trim($data['lang']) . "\n";
$configContent .= "SYS_LAYOUT=" . trim($data['layout']) . "\n";
$configContent .= "SYS_TIMEZONE=" . trim($data['timezone']) . "\n";
$configContent .= "SYS_HOSTNAME=" . $clean_hostname . "\n";
$configContent .= "DISK_MODE=" . trim($data['disk_mode']) . "\n";
```

```
// Жестко очищаем массив дисков от любых пробелов и переносов перед склейкой
$clean_disks = array_map('trim', $data['disks']);
$configContent .= "SELECTED_DISKS=" . implode(' ', $clean_disks) . "\n";

$configContent .= "SYS_USER=" . $clean_username . "\n";
$configContent .= "SYS_PASS=" . $clean_password . "\n";


$target_file = __DIR__ . '/install_config.txt';

// Перезаписываем файл актуальными данными
if (file_put_contents($target_file, $configContent) !== false) {
    chmod($target_file, 0600);
  // Передаем управление ОДНОМУ управляющему скрипту
    exec('sudo /srv/http/installer/api/disk_prepare.sh >> /tmp/install.log
2>&1 &');
    echo json_encode(['success' => true, 'message' => 'Установка успешно
запущена.']);
} else {
    echo json_encode(['success' => false, 'message' => 'Ошибка записи
конфигурации']);
}
exit;
```

```
=== system_install.sh ===

#!/bin/bash
# Скрипт точного клонирования и создания Recovery (api/system_install.sh)
cd "$(dirname "$0")"
LOG_FILE="/tmp/install.log"

log_msg() {
    echo "$1" | /usr/bin/tee -a "$LOG_FILE" > /dev/null
    echo "$1"
}

# ====================================================================
# 1. Чтение конфигурации и ЖЕСТКОЕ МОНТИРОВАНИЕ ФИЗИЧЕСКИХ ДИСКОВ
# ====================================================================
# Импортируем переменные напрямую (на случай, если PHP потерял их в памяти)
if [ -f "install_config.txt.bak" ]; then
    source <(tr -d '\r' < "install_config.txt.bak")
elif [ -f "$CONFIG_FILE" ]; then
    source <(tr -d '\r' < "$CONFIG_FILE")
fi

if [ -z "$SELECTED_DISKS" ]; then
    SELECTED_DISKS="sdb sdc"
fi
```

```
IFS=' ' read -r -a DISKS_ARRAY <<< "$SELECTED_DISKS"

# Вычисляем чистые имена разделов без скрытых символов Windows
FIRST_BTRFS_DISK="/dev/$(echo "${DISKS_ARRAY[0]}" | tr -d '\r')2"
FIRST_EFI_DISK="/dev/$(echo "${DISKS_ARRAY[0]}" | tr -d '\r')1"

# Принудительно монтируем реальные физические разделы жестких дисков через sudo
# чтобы перепримонтировать диски строго по Btrfs-полочкам
sudo /usr/bin/umount -R /mnt 2>/dev/null

    log_msg "[INFO] Монтирование физического раздела $FIRST_BTRFS_DISK в
/mnt..."
    sudo /usr/bin/mount -o noatime,compress=zstd,subvol=@
"$FIRST_BTRFS_DISK" /mnt >> "$LOG_FILE" 2>&1
    sudo /usr/bin/mkdir -p /mnt/home
    sudo /usr/bin/mount -o noatime,compress=zstd,subvol=@home
"$FIRST_BTRFS_DISK" /mnt/home >> "$LOG_FILE" 2>&1
    sudo /usr/bin/mkdir -p /mnt/boot/efi
    sudo /usr/bin/mount "$FIRST_EFI_DISK" /mnt/boot/efi >> "$LOG_FILE" 2>&1



log_msg "[PROGRESS] 55"
log_msg "[INFO] Клонирование системных директорий сервера-донора через rsync..."

log_msg "[DIAGNOSTIC] === КАНТРОЛЬ МОНТИРОВАНИЯ ПЕРЕД ЗАПИСЬЮ ==="
# Проверяем, что /mnt — это реальный диск, а не RAM-папка веб-сервера
sudo /usr/bin/df -h /mnt >> "$LOG_FILE" 2>&1
sudo /usr/bin/df -h /mnt/boot/efi >> "$LOG_FILE" 2>&1


# 2. ЗАПУСК КЛОНИРОВАНИЯ (RSYNC): Пишем через tee -a, обходя ошибки прав
/usr/bin/rsync -aAXv --
exclude={/dev/*,/proc/*,/sys/*,/tmp/*,/run/*,/mnt/*,/media/*,/lost+found,/tm
p/install.log} / /mnt/ | /usr/bin/tee -a "$LOG_FILE" > /dev/null

if [ ${PIPESTATUS[0]} -ne 0 ]; then
    log_msg "[ERROR] Ошибка клонирования системных директорий!"
    exit 1
fi

# ======================================================================
# □ ИНСПЕКЦИЯ РЕЗУЛЬТАТОВ ЗАПИСИ (ФИЗИЧЕСКИЙ КОНТРОЛЬ КЛОНИРОВАНИЯ И GRUB)
# ======================================================================
log_msg "[DIAGNOSTIC] === СТАРТ ПРОВЕРКИ ФИЗИЧЕСКОЙ ЗАПИСИ ДАННЫХ ==="

log_msg "[DIAGNOSTIC] Физический объем данных, записанных rsync на Btrfs-
зеркало:"
sudo /usr/bin/du -sh /mnt/ --exclude=/mnt/boot/efi >> "$LOG_FILE" 2>&1

log_msg "[DIAGNOSTIC] Проверка структуры папок корня новой ОС на диске:"
```

```
sudo /usr/bin/ls -la /mnt/ >> "$LOG_FILE" 2>&1

log_msg "[DIAGNOSTIC] Физический объем загрузчика, записанного на EFI FAT32:"
sudo /usr/bin/du -sh /mnt/boot/efi/ >> "$LOG_FILE" 2>&1

log_msg "[DIAGNOSTIC] Контроль наличия монолитных файлов GRUB и модулей (.mod)
на FAT32:"
sudo /usr/bin/ls -R /mnt/boot/efi/ >> "$LOG_FILE" 2>&1

log_msg "[DIAGNOSTIC] Содержимое сгенерированного файла fstab новой системы:"
sudo /usr/bin/cat /mnt/etc/fstab >> "$LOG_FILE" 2>&1

log_msg "[DIAGNOSTIC] === КОНЕЦ ТОТАЛЬНОЙ ИНСПЕКЦИИ ЗАПИСИ ==="
# ====================================================================


# 3. ЗАПИСЬ "ЗОЛОТОГО ОБРАЗА" В ОБЛАСТЬ RECOVERY DISK
log_msg "[PROGRESS] 70"
log_msg "[INFO] Создание эталонного резервного архива в области Recovery Disk..."

/usr/bin/mkdir -p /tmp/recovery_pool
/usr/bin/mount -o noatime,compress=zstd,subvol=@security "$FIRST_DISK"
/tmp/recovery_pool | /usr/bin/tee -a "$LOG_FILE" > /dev/null 2>&1

# Архивируем чистую систему напрямую в подтом @security через tee
/usr/bin/tar -cpzf /tmp/recovery_pool/golden_image.tar.gz -C /mnt/ . |
/usr/bin/tee -a "$LOG_FILE" > /dev/null 2>&1

if [ ${PIPESTATUS[0]} -ne 0 ]; then
    log_msg "[ERROR] Не удалось собрать архив восстановления в @security!"
    /usr/bin/umount /tmp/recovery_pool
    exit 1
fi

/usr/bin/umount /tmp/recovery_pool
/usr/bin/rmdir /tmp/recovery_pool
log_msg "[INFO] Золотой образ успешно сохранен на встроенном Recovery Disk."

log_msg "[PROGRESS] 80"
log_msg "[INFO] Generation новой таблицы разделов fstab..."

# 4. Генерируем чистый fstab для новых Btrfs-разделов
# ИСПРАВЛЕНО: Генерируем fstab напрямую в файл новой системы, перезаписывая старый
мусор донора
sudo /usr/bin/genfstab -U /mnt | sudo /usr/bin/tee /mnt/etc/fstab >
/dev/null

log_msg "[PROGRESS] 85"
log_msg "[INFO] Базовая система и Recovery-раздел успешно подготовлены."

# ====================================================================
```

```
# 5. ХОСТ-СБОРКА GRUB UEFI И СТАНДАРТНЫЙ CHROOT НАСТРОЕК (БЕЗОШИБОЧНЫЙ)
# ======================================================================
log_msg "[PROGRESS] 85"
log_msg "[INFO] Сборка и установка монолитного загрузчика GRUB UEFI..."

# 1. Принудительно заставляем хост собрать GRUB для целевой системы, указывая ключ --
boot-directory=/mnt/boot
# Хост имеет 100% доступ к дискам, поэтому grub-install выполнится БЕЗ ошибок!
sudo /usr/bin/grub-install --target=x86_64-efi --efi-directory=/mnt/boot/efi
--boot-directory=/mnt/boot --bootloader-id=GRUB --modules="btrfs part_gpt
normal fat ext2 configfile echo" --removable --no-nvram --recheck >>
"$LOG_FILE" 2>&1

# 2. АВТОНОМНЫЙ ФИКС: Копируем модули GRUB напрямую на FAT32 раздел
sudo /usr/bin/mkdir -p /mnt/boot/efi/EFI/BOOT/x86_64-efi
sudo /usr/bin/cp -r /usr/lib/grub/x86_64-efi/*
/mnt/boot/efi/EFI/BOOT/x86_64-efi/

# 3. не сканировал диски хоста и гарантированно НЕ ронял PHP-FPM в 502 ошибку!
export GRUB_DISABLE_OS_PROBER=true
# Генерируем конфигурацию GRUB прямо из контекста хоста внутрь рейда
sudo /usr/bin/grub-mkconfig -o /mnt/boot/grub/grub.cfg >> "$LOG_FILE" 2>&1
sudo /usr/bin/grub-mkconfig -o /mnt/boot/efi/EFI/BOOT/grub.cfg >>
"$LOG_FILE" 2>&1

log_msg "[INFO] Запуск chroot для настройки сетевого имени и учетных записей..."

# 4. Монтируем только базовые папки для chroot настроек пользователей
sudo /usr/bin/mount --bind /dev /mnt/dev
sudo /usr/bin/mount --bind /proc /mnt/proc
sudo /usr/bin/mount --bind /sys /mnt/sys

# 5. Передаем переменные во временный файл конфигурации окружения
cat <<EOF | sudo /usr/bin/tee /mnt/etc/installer_env.conf > /dev/null
SYS_HOSTNAME=$(echo "$SYS_HOSTNAME" | tr -d '\r')
SYS_USER=$(echo "$SYS_USER" | tr -d '\r')
SYS_PASS=$(echo "$SYS_PASS" | tr -d '\r')
SYS_TIMEZONE=$(echo "$SYS_TIMEZONE" | tr -d '\r')
EOF

# 6. Фильтрует скрытые символы \r и запускает chroot_configure.sh ТЕПЕРЬ ТОЛЬКО
ДЛЯ ПАРОЛЕЙ И ЮЗЕРОВ
sudo /usr/bin/tr -d '\r' < /srv/http/installer/api/chroot_configure.sh |
sudo /usr/bin/tee /mnt/root/chroot_exec.sh > /dev/null
sudo /usr/bin/chmod +x /mnt/root/chroot_exec.sh
sudo /usr/bin/chroot /mnt /root/chroot_exec.sh >> "$LOG_FILE" 2>&1

# 7. Ленивое безопасное размонтирование и зачистка
sudo /usr/bin/rm -f /mnt/root/chroot_exec.sh
sudo /usr/bin/rm -f /mnt/etc/installer_env.conf
sudo /usr/bin/umount -l /mnt/dev 2>/dev/null
```

```
sudo /usr/bin/umount -l /mnt/proc 2>/dev/null
sudo /usr/bin/umount -l /mnt/sys 2>/dev/null

log_msg "[PROGRESS] 90"
log_msg "[INFO] Финальная настройка успешно завершена. Сервер готов к работе!"
```

```
===validate_input.php===

<?php
// validate_input.php
// Функция полной очистки строк
function sanitize_and_validate($data, $max_length = 16) {
  // 1. Запрещаем пробелы, табы, переносы строк
    $data = preg_replace('/\s+/', '', $data);
  // 2. Обрезаем строго до 16 символов (п. 7)
    $data = mb_substr($data, 0, $max_length, 'UTF-8');
  // 3. Вырезаем любые HTML-теги, скрипты, конструкции типа <?php (п. 4, 5, 6, 8)
    $data = strip_tags($data);
  // 4. Оставляем ТОЛЬКО буквы (латиница), цифры и знаки подчеркивания/дефиса/точки
  // Это полностью уничтожает любые попытки SQL-инъекций (типа OR 1=1) и XSS-атак
    $data = preg_replace('/[^a-zA-Z0-9_\-\.]/', '', $data);
    return $data;
}

// Черные списки запрещенных слов (п. 4.1 и 4.2)
$forbidden_usernames = ['admin', 'administrator', 'test', 'root', 'user',
'guest'];
$forbidden_passwords = ['qwerty', 'abcdef', '123456', 'iamnumber1',
'password'];

// Пример проверки пришедших POST-данных (встраивается в api/start_install.php)
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $input_user = isset($_POST['username']) ? $_POST['username'] : '';
    $input_pass = isset($_POST['password']) ? $_POST['password'] : '';

  // Чистим данные
    $clean_user = sanitize_and_validate($input_user, 16);
    $clean_pass = sanitize_and_validate($input_pass, 16);

  // Проверяем по массивам запретов (вхождение подстроки)
    foreach ($forbidden_usernames as $bad_user) {
        if (strpos(strtolower($clean_user), $bad_user) !== false ||
empty($clean_user)) {
            echo json_encode(['error' => 'Недопустимое имя пользователя!']);
            exit;
        }
    }

    foreach ($forbidden_passwords as $bad_pass) {
        if (strpos(strtolower($clean_pass), $bad_pass) !== false ||
empty($clean_pass)) {
```

```
            echo json_encode(['error' => 'Слишком простой или запрещенный
пароль!']);
            exit;
        }
    }
}
```