

## Описание созданного стенда и снимки экрана веб-приложения, работающего на стенде

### 1. Описание созданного стенда

При выполнении тестового задания был разработан пакет сборки и развертывания веб-приложения, а с его помощью создан тестовый стенд на базе OS Ubuntu 16.04 (Xenial). Поскольку, в тестовом задании необходимо было продемонстрировать конфигурирование окружения, сборку и развертывание приложения, то наиболее близко этому соответствовал формат build/stage окружения, предполагающий как сборку, так и деплой. Поэтому, после конфигурирования стенда исходные модули приложения копируются на стенд, после чего разворачиваются и собираются сборщиком Maven, затем собранное приложение разворачивается в Tomcat. Для автоматизации задач конфигурирования, сборки и развертывания приложения была выбрана система управления конфигурациям Ansible, поскольку дает возможность автоматизировать все этапы конфигурирования стенда, при этом не требует от целевого хоста наличия дополнительного программного обеспечения либо агентов, достаточно SSH-доступа (Рис. 1).

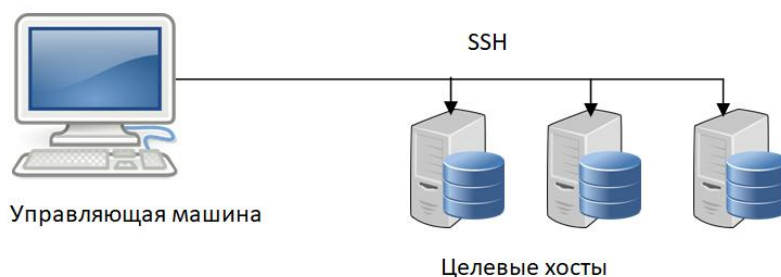


Рис. 1. Структурная схема развертывания конфигурации

Данный подход к автоматизации позволяет гибко изменять процесс развертывания как по составу используемого программного обеспечения, так и по количеству и составу целевых машин. После запуска и выполнения пакета инсталляции, на хостах назначения, указанных в инвентори-файле, разворачивается все инфраструктурное ПО, развертывается и собирается проект, после чего собранное приложение устанавливается и запускается в сервере Tomcat. Сразу после выполнения установки веб-приложение доступно по адресу <адрес целевого хоста>:8080/absence-control, создаются журналы логов установщика, сервера Tomcat и самого приложения. Скриншоты работающего веб-приложения приводятся в Приложении 1, тестовый стенд доступен по адресу <http://31.211.65.203:8080/absence-control>

### 2. Особенности реализации

2.1. Для подготовки стенда использовалась сборка Ubuntu 16.04, поэтому все скрипты пакета гарантированно работают для стандартной инсталляции данной ОС и оснащены методами проверки версии платформы назначения. В других сборках работа скриптов не гарантируется, для использования пакета установки на других платформах следует модифицировать установочный пакет с учетом особенностей реализации данных платформ.

2.2. Часть необходимого инфраструктурного ПО требуемых версий отсутствует в официальных репозиториях, поэтому в файлах конфигурации указывается URL на tar.gz-архив с официального сайта (например Tomcat 9)

2.3. Сборка осуществляется непосредственно на целевом хосте назначения. Поскольку, некоторые параметры захардкожены в `rom.xml` (например порт подключения к БД и `dbUser`), то эти параметры должны быть определены в соответствии с параметрами хоста, где разворачивается приложение. Так как они задаются в переменных соответствующих ролей Ansible, то во избежание ошибок, файл `rom.xml` генерируется по j2-шаблону и приложение собирается по месту, после чего разворачивается. Также реализован вариант простого развертывания собранного заранее web-архива, для этого следует указать тэг `--fastdeploy` при выполнении плейбука (тогда сборка осуществляться не будет, а будет развернут `war`-файл из роли `deploy/files`).

2.4. Сборка, конфигурирование и разворачивание приложения осуществляется с управляющей машины администратора с установленным пакетом Ansible. На целевой машине (машинах) должен быть лишь настроен доступ по SSH. При необходимости, роль управляющей может исполнять сама целевая машина, для этого необходимо добавить в инвентори-файле (`./inventory/hosts`) строку `127.0.0.1`

2.5. В папке проекта был расположен `sql`-скрипт, добавляющий набор тестовых данных в базу. Поскольку, загрузка тестовых данных была предусмотрена в самом приложении при первом запуске, то заполнение базы тестовыми данными при разворачивании не требуется.

2.6. Скрипт **"check\_app.sh"** проверки правильности развертывания находится в разделе **utils** пакета. В качестве параметра задается адрес хоста назначения, где было развернуто приложение

2.7. Журналирование работы приложения было включено путем создания в проекте схемы описания логгера для `log4j2`. Вывод журнала производится в `absence-control.log`

### 3. Пути дальнейшего развития пакета развертывания приложения **absence-control**

3.1. Автоматизация строилась исходя из предположения что на платформе назначения установлена сборка Ubuntu 16.04. В целях обеспечения платформонезависимости скрипт может быть модифицирован для создания виртуальной машины Vagrant и конфигурирования в ней окружения для последующего развертывания приложения.

3.2. Разделение операций деплоя и сборки таким образом, чтобы после успешной сборки приложение упаковывалось в `deb`-package и сохранялось в локальном репозитории, а при деплое разворачивалось так же как инфраструктурное ПО. Вариант не очень хорош для билд-окружения, однако хорошо впишется при развертывании стабильных версий в продакшн.

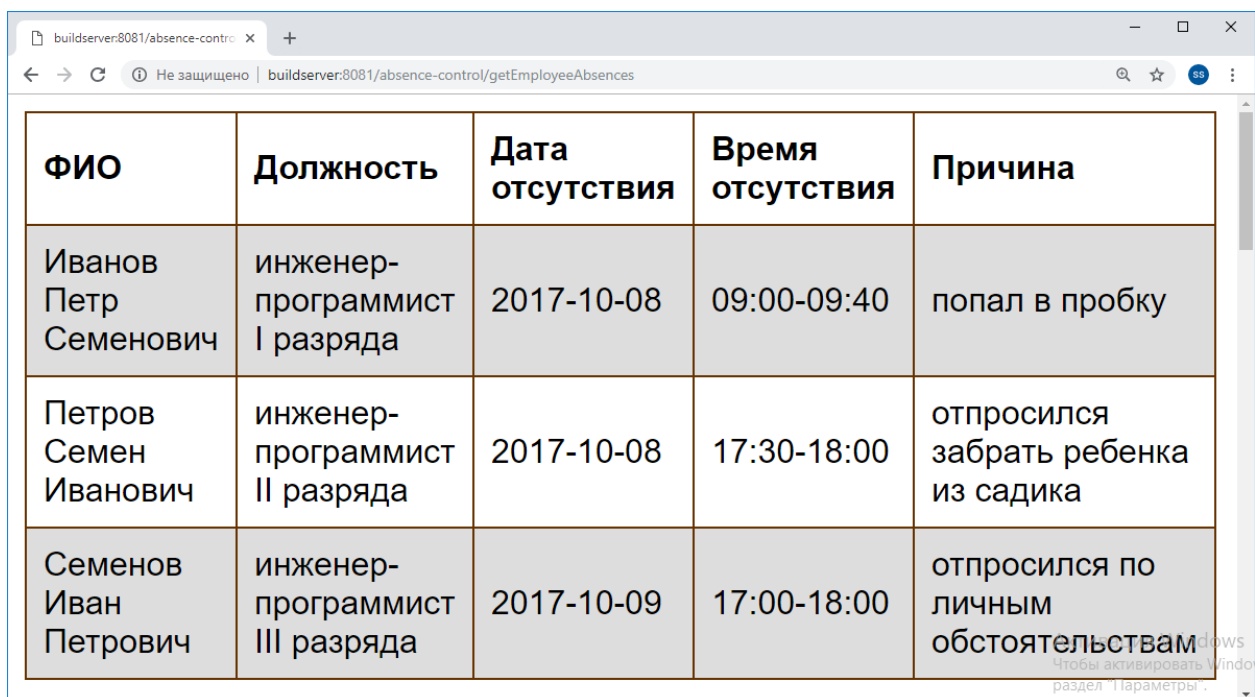
3.3. Сейчас требуемая версия Tomcat 9 скачивается и устанавливается из `tar.gz` архива. Возможна его установка в виде сервиса, что даст возможность управлять состоянием сервера через `service tomcat`

3.4. Скрипт проверки корректности установки и доступности приложения следует реализовать в отдельной роли, а не в виде внешнего скрипта, при этом скрипт должен использовать переменные окружения, заданные для пакета

3.5. Следует реализовать роли для удаления и повторной установки ПО

3.6. Оформить проект на Github, создать Readme в MD-нотации, подробно описать в readme и инструкции назначение всех переменных ролей, которые могут быть использованы для настройки, а не только инвентори-файла. (Выходит за рамки тестового задания, подробно оформлять не стал)

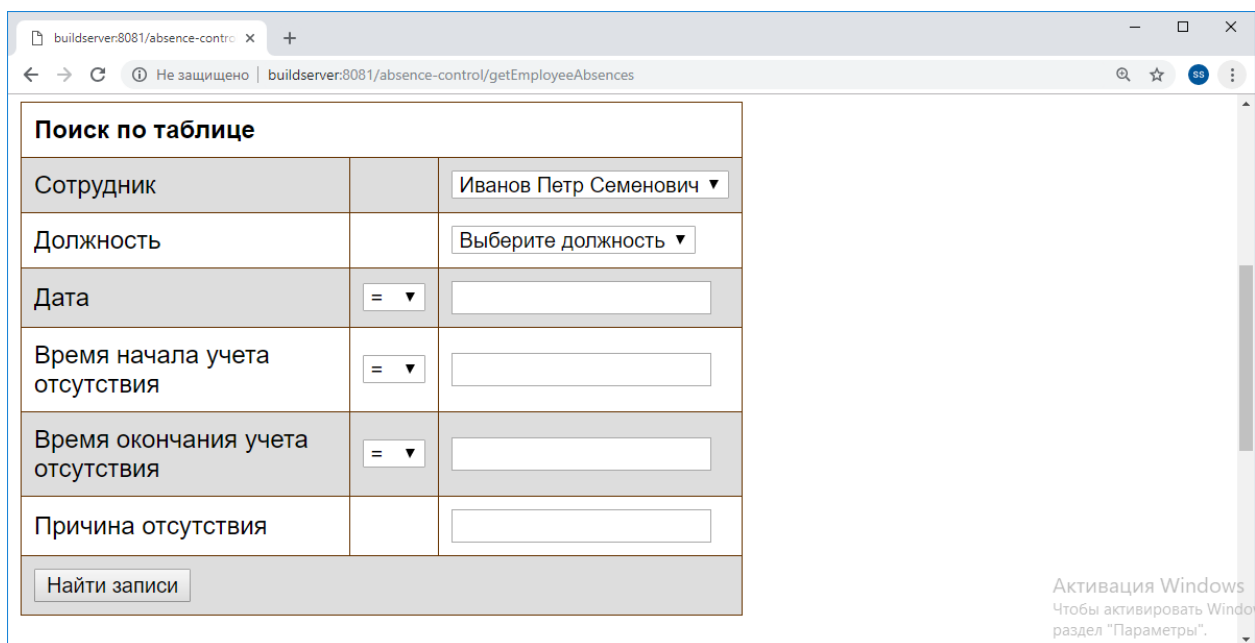
## Приложение 1. Снимки экрана, веб-приложения



The screenshot shows a web browser window with the address bar displaying 'buildserver:8081/absence-control/getEmployeeAbsences'. The main content is a table with five columns: ФИО, Должность, Дата отсутствия, Время отсутствия, and Причина. The table contains three rows of data.

ФИО	Должность	Дата отсутствия	Время отсутствия	Причина
Иванов Петр Семенович	инженер-программист I разряда	2017-10-08	09:00-09:40	попал в пробку
Петров Семен Иванович	инженер-программист II разряда	2017-10-08	17:30-18:00	отпросился забрать ребенка из садика
Семенов Иван Петрович	инженер-программист III разряда	2017-10-09	17:00-18:00	отпросился по личным обстоятельствам

Рис. 1. Скриншот главной страницы приложения после инициализации базы тестовыми данными



The screenshot shows a web browser window with the address bar displaying 'buildserver:8081/absence-control/getEmployeeAbsences'. The main content is a search form titled 'Поиск по таблице'. The form has several input fields and dropdown menus for filtering data.

Поиск по таблице		
Сотрудник		Иванов Петр Семенович ▼
Должность		Выберите должность ▼
Дата	= ▼	<input type="text"/>
Время начала учета отсутствия	= ▼	<input type="text"/>
Время окончания учета отсутствия	= ▼	<input type="text"/>
Причина отсутствия		<input type="text"/>
<input type="button" value="Найти записи"/>		

Рис. 2. Скриншот формы поиска по таблице

buildserver:8081/absence-control x +

← → ↻ ⓘ Не защищено | buildserver:8081/absence-control/getEmployeeAbsences

**Добавление информации об отсутствии**

Сотрудник	Выберите сотрудника ▼
Должность	Выберите должность ▼
Дата	<input type="text"/>
Время начала учета отсутствия	<input type="text"/>
Время окончания учета отсутствия	<input type="text"/>
Причина отсутствия	<input type="text"/>
<input type="button" value="Добавить запись"/>	

Активация Windows  
Чтобы активировать Windows, перейдите в меню "Параметры".

Рис. 3. Скриншот формы добавления информации об отсутствии