# Ассистент для команд разработки

## Руководство администратора

**Всего листов: 22**

# Аннотация

Настоящий документ представляет собой руководство администратора (Руководство) по работе с Ассистентом для команд разработки (Система) и является частью эксплуатационной документации на Систему.

Документ включает в себя:

- общие сведения о функциональных возможностях Системы;
- описание порядка подготовки инфраструктуры к развертыванию программного обеспечения Системы;
- описание порядка развертывания программного обеспечения Системы;
- описание операций по администрированию Системы.

# Содержание

**Аннотация** ..... **2**

**1. Введение** ..... **4**
1.1. Область применения системы ..... 4
1.2. Краткое описание возможностей системы ..... 4
1.3. Уровень подготовки пользователя ..... 4
1.4. Перечень эксплуатационной документации для ознакомления ..... 4
1.5. Глоссарий ..... 4

**2. Общее описание системы «Ассистент для команд разработки»** ..... **9**
2.1. Назначение системы ..... 9
2.2. Условия применения системы ..... 9
2.2.1. Общие условия ..... 9
2.2.2. Требования к инфраструктуре ..... 9
2.2.3. Требования к рабочей станции для развертывания контура ..... 10
2.2.4. Требования к программному обеспечению инфраструктуры ..... 11

**3. Подготовка к работе системы «Ассистент для команд разработки»** ..... **13**
3.1. Установка серверной части ..... 13
3.1.1. Подготовка виртуальных машин и базовой среды ..... 13
3.1.2. Развертывание инфраструктуры сервисов ..... 13
3.1.3. Развертывание компонентов в Kubernetes ..... 14
3.1.4. Настройка серверов и развёртывание LLM ..... 15
3.1.5. Настройка мониторинга и проверка работоспособности ..... 13
3.2. Порядок запуска ..... 15
3.3. Порядок остановки ..... 15
3.4. Порядок установки обновлений системы ..... 16
3.5. Порядок проверки работоспособности системы ..... 16
3.6. Решения по диагностированию системы ..... 16

**4. Описание операций системы «Ассистент для команд разработки»** ..... **17**
4.1. Управление учетными записями пользователей ..... 17
4.1.1. Создание учетной записи пользователя ..... 17
4.1.2. Обновление пароля учетной записи пользователя ..... 17
4.1.3. Блокировка учетной записи пользователя ..... 17

**5. Планово-предупредительные мероприятия** ..... **19**
5.1. Документирование результатов ..... 20

**6. Аварийное восстановление системы «Ассистент для команд разработки»** ..... **21**
6.1. Критерии аварийного события ..... 21
6.2. Порядок действий при выявлении аварийного события ..... 21
6.3. Общие рекомендации по восстановлению деятельности системы «Ассистент для команд разработки» ..... 22

# 1. Введение

## 1.1. Область применения системы

Областью применения системы является генерация и документирование программного кода, а также создание автоматизированных тестов для проверки функциональности и качества программного обеспечения.

## 1.2. Краткое описание возможностей системы

Система предоставляет следующие возможности:

- Генерация программного кода по описанию на естественном языке. Поддержка генерации на следующих языках программирования: Python, C, C++, C#, Java, JavaScript, TypeScript, Kotlin, Ruby, Go, Bash, 1C;
- Генерация unit-тестов к программному коду;
- Анализ исходного кода (код-ревью) и генерация рекомендаций по его улучшению;
- Генерация DocString и комментариев к программному коду;
- Генерация комментариев к pull requests;
- Генерация тест-кейсов на основе требований к ПО.

## 1.3. Уровень подготовки пользователя

Для выполнения операций, предусмотренных настоящим Руководством, пользователю с ролью «Администратор» потребуются знания основ информационной безопасности, навыки администрирования серверов и вычислительных сетей, рекомендуется наличие навыков и опыта работы с технологиями docker, kubernetes, Keycloak, Postgresql. Необходимо обладать опытом управления системами с искусственным интеллектом (ИИ) и опытом настройки LLM , что предполагает наличие навыков создания, редактирования и оптимизации инструкций для LLM, навыков оценки качества выходных данных LLM, навыков анализа текстов для составления инструкций и примеров для LLM. Также потребуется понимание принципов обработки естественного языка, понимание синтаксиса, семантики и языковой структуры русского и английского языков.

## 1.4. Перечень эксплуатационной документации для ознакомления

Перед началом работы по установке, настройке, администрированию Системы пользователю с ролью «Администратор» необходимо ознакомиться с настоящим Руководством в полном объеме.

## 1.5. Глоссарий

В тексте настоящего документа представлены следующие сокращения (см. **Таблица 1-1**).

***Таблица 1-1*** *— Перечень сокращений*

| Сокращение/аббревиатура | Значение |
|---|---|
| **БД** | База данных - совокупность данных, организованных в соответствии с концептуальной структурой, описывающей характеристики этих данных и взаимоотношения между ними, которая поддерживает одну или более областей применения |
| **LLM** | Large Language Model - большая языковая модель |
| **ИИ** | Искусственный интеллект |
| **ИС** | Информационная система |
| **ОС** | Операционная система |
| **ПО** | Программное обеспечение |
| **Руководство** | Руководство по работе с Ассистентом для команд разработки (руководство администратора Системы) – настоящий документ |
| **Система** | Ассистент для команд разработки |
| **СУБД** | Система управления базами данных - Совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных (БД). |
| **API** | Application Programming Interface - Программный интерфейс, описание способов взаимодействия одной компьютерной программы с другими, используется при разработке приложений |
| **JSON** | JavaScript Object Notation - Текстовый формат обмена данными, основанный на языке программирования JavaScript |

| Сокращение/аббревиатура | Значение |
|---|---|
| **TCP/IP** | Transmission Control Protocol/Internet Protocol - Набор сетевых протоколов передачи данных, используемых в сетях, включая глобальную сеть Интернет |
| **URL** | Uniform Resource Locator - система унифицированных адресов электронных ресурсов, или единообразный определитель местонахождения ресурса |
| **VM** | Виртуальная машина |

В тексте настоящего документа представлены следующие термины и определения (см. **Таблица 1-2**).

***Таблица 1-2** — Перечень терминов и определений*

| Термин | Определение |
|---|---|
| **Авторизация** | Предоставление определенному лицу или группе лиц прав на выполнение определенных действий |
| **Аутентификация** | Проверка подлинности предъявленного пользователем идентификатора |
| **Заказчик** | Физическое или юридическое лицо, которое инициирует, заказывает и финансирует разработку, модификацию или поддержку программного обеспечения Системы.<br>После приемки услуг по разработке Системы Заказчик является Владельцем Системы |
| **Идентификатор пользователя** | Уникальный признак объекта, позволяющий отличать его от других |
| **Идентификация** | Процедура, в результате выполнения которой для субъекта выявляется его идентификатор, однозначно определяющий этого субъекта в Системе |

| | |
|---|---|
| **Компонент (программного обеспечения)** | Составная часть программного обеспечения, выполняющая определенную функцию |
| **Нод (Node) (kubernetes)** | Ноды или узлы, виртуальные или физические машины, на которых разворачивают и запускают контейнеры. Совокупность нод образует кластер kubernetes |
| **Под (Pod) (в kubernetes)** | Группа контейнеров с общими разделами, которые запускаются как одно приложение |
| **Пользователь** | Работник, наделенный правами доступа к Ассистенту для команд разработки |
| **Ролевая модель** | Распределение прав доступа и обязанностей между ролями. Определяет, какие действия может выполнять каждая роль в рамках работы с Системой, включая создание и редактирование правил, проведение проверок и управление |
| **Роль** | Набор полномочий, который необходим пользователю для выполнения определённых рабочих задач. Каждый сотрудник может иметь одну или несколько ролей, а каждая роль может содержать от одного до множества полномочий.<br>Роли пользователей в Системе:<br>«Пользователь»;<br>«Администратор»; |
| **Роль «Администратор»** | Главная роль Системы, предоставляющая доступ к настройкам Системы |
| **Роль «Пользователь»** | Роль для лица, использующего функциональность Системы |
| **Helm** | Пакетный менеджер для kubernetes, который используется для управления приложениями в кластере |
| **Kubectl** | Командная строка для управления Kubernetes; необходим для взаимодействия с кластером Kubernetes |

| | |
|---|---|
| **Kubernetes** | Платформа для автоматизации развёртывания, масштабирования и управления контейнеризированными приложениями. Поддерживает основные технологии контейнеризации (docker, rocket) и аппаратную виртуализацию |
| **sshpass** | Утилита для автоматизации ввода пароля при использовании SSH, что требуется для настройки доступа к серверам |
| **PostgreSQL** | Объектно-реляционная СУБД с открытым исходным кодом, которая использует и расширяет язык SQL в сочетании со многими функциями, которые позволяют безопасно хранить и масштабировать самые сложные рабочие нагрузки с данными |

# 2. Общее описание системы «Ассистент для команд разработки»

## 2.1. Назначение системы

- Система предназначена для ускорения процесса написания кода за счет подсказок и ответов на вопросы; улучшение качества кода через анализ ошибок и рекомендации по улучшению; сокращение времени разработки и вывода продуктов на рынок, сокращение материальных затрат на разработку новых сервисов;
- Система предназначена для удовлетворения задач использования безопасных решений для помощи команд разработки, соответствующих стандартам работы в корпоративных контурах, включая поддержку замкнутых сетей и исключение утечек данных;
- Система подразумевает возможность подбора размера LLM под существующую ИТ-инфраструктуру и потребности заказчика;
- Система подразумевает её использование по API другими приложениями, такими как ИИ плагины для браузера, плагины для сред разработки (IDE);
- В системе заложено использование инструментов для оценки эффективности решения и мониторинга активности команды.

## 2.2. Условия применения системы

### 2.2.1. Общие условия

Режим эксплуатации Системы: 5x8 (5 дней в неделю по 8 часов) в часы работы Заказчика.

Для возможности администрирования Системы должны выполняться условия, предусмотренные пп. 2.2.2, 2.2.3, 2.2.4 Руководства.

Для настройки журналирования событий по информационной безопасности уровень логирования рекомендуется выставлять в warning - как компромиссный уровень между объемом информации и детализацией.

### 2.2.2. Требования к инфраструктуре

Для работы Системы требуется 5 виртуальных машин, состав которых определен в **Таблице 2-1** настоящего Руководства.

***Таблица 2-1 –*** *Состав виртуальных машин*

| Компонент | Состав | Название сервера | ip | Примечание |
|---|---|---|---|---|
| Центр анализа – LLM | code LLM, nvidia GPU drivers, docker-toolkit docker engine | vdc-rk-llm-gpu | 10.10.10.10 | - |

| | | | | |
|---|---|---|---|---|
| СУБД | PostgreSQL | vdc-rk-postgres01 | 10.10.10.11 | - |
| Master node kubernetes 01 | k8s | vdc-rk-k8s01 | 10.10.10.12 | - |
| Worker node kubernetes 02 | k8s | vdc-rk-k8s02 | 10.10.10.13 | - |
| Сервер объектного хранилище S3 | minio | vdc-rk-minio01 | 10.10.10.19 | - |

### 2.2.3.Требования к рабочей станции для развертывания контура

Для выполнения операций по развертыванию контура Системы на рабочей станции должны быть выполнены следующие требования:

- операционная система Linux:
  - рабочая станция должна быть запущена на базе операционной системы (ОС) Linux;
  - ОС должна взаимодействовать с драйверами на GPU;
  - дополнительные требования к ОС определяются внутренними документами Заказчика;
- программное обеспечение:
  - helm;
  - kubectl;
  - sshpass;
- сетевые настройки: рабочая станция должна иметь доступ по SSH ко всем серверам, участвующим в развертывании контура Системы;
- доступ по SSH-ключам: для упрощения процесса развертывания рекомендуется настроить беспарольный доступ по SSH с рабочей станции на все серверы;
- права доступа: пользователь на рабочей станции должен иметь привилегированный доступ (sudo) для выполнения административных задач;
- настройки безопасности: SSH-ключи, используемые для доступа к серверам, должны быть защищены и безопасно храниться;

- дополнительные утилиты: утилиты `wget`, `curl`, `tar`, `unzip`, `telnet nc net-tools` должны быть подготовлены для скачивания и распаковки файлов;
- резервное копирование: рекомендуется регулярно создавать резервные копии конфигурационных файлов и ключей на рабочей станции;
- проверка и тестирование: на рабочей станции должны быть установлены инструменты для тестирования и проверки конфигураций, такие как ping, telnet, nslookup, net-tools, nc;
- документация: рабочая станция должна иметь доступ к актуальной документации о Системе и инструкциям по развертыванию.

### 2.2.4.Требования к программному обеспечению инфраструктуры

Для обеспечения работы Системы необходимо соблюдать следующие требования:

- наличие программного обеспечения (ПО) на виртуальных машинах (VM), составляющих инфраструктуру Системы, соответствующего требованиям, установленным в п. 2.2.4 Руководства;
- ОС: поддерживает драйверы GPU Nvidia H100 и соответствует техническим требованиям Заказчика по RHEL;
- требования к виртуальным машинам – предоставляются отдельно;
- в контуре должен быть установлен NTP-сервер;
- все машины контура должны синхронизировать время;
- на всех хостах должен быть заведён служебный пользователь с правом выполнения команд с привилегиями суперпользователя для установки и настройки компонентов Системы;
- на всех хостах должен быть установлен и настроен ssh с возможностью авторизации служебным пользователем;
- CPU – рекомендуется Xeon Gold или аналог;
- отсутствие VPN соединений между VM;
- сеть:
    - скорость не менее 1 Гбит;
    - агрегация физических линков на серверах для отказоустойчивости (не менее 2 подключений);
    - для схемы размещения в одном ЦОД все серверы должны быть в одной подсети без фильтрации трафика между ними;

- на каждой виртуальной машине должен быть установлен и настроен iptables;
- наличие Nexus: в Nexus должны быть созданы по списку репозитории для apt, helm, npm, maven, docker, pypi;
- наличие docker для запуска контейнеризованных приложений, необходимый для работы с docker-образами и контейнерами.

# 3. Подготовка к работе системы «Ассистент для команд разработки»

## 3.1. Установка серверной части

### 3.1.1. Подготовка виртуальных машин и базовой среды

Необходимо выполнить следующие шаги:

a) Создать виртуальные машины:

- PostgreSQL-сервер: одна машина для PostgreSQL для хранения данных;
- kubernetes-кластер: две виртуальные машины для kubernetes (K8s), на которых будет развёрнут кластер для компонентов обработки данных;
- S3-хранилище: один сервер для MinIO, который обеспечит функциональность S3-хранилища;
- Сервер с GPU: одна виртуальная машина с NVIDIA GPU для запуска контейнеров с поддержкой графического ускорителя, необходимых для компонента LLM;

b) Установить ОС и сетевые настройки:

1) установить совместимую с Linux ОС на всех виртуальных машинах (например, Ubuntu 20.04 или CentOS 7/8);
2) настроить сеть, чтобы обеспечить доступ между серверами, а также открыть порты для необходимых сервисов:
   - PostgreSQL: 5432;
   - kubernetes API: 6443;
   - kubernetes NodePorts: 30000, 30001, 30888, 30083, 30005, 30051, 30050;
   - MinIO: 9000;
3) установить базовые утилиты и зависимости, включая curl, wget, git, vim, docker, kubectl, и другие необходимые инструменты.

### 3.1.2. Развертывание инфраструктуры сервисов

Необходимо выполнить следующие шаги:

a) PostgreSQL:

1) установить PostgreSQL на выделенной машине;
2) настроить конфигурацию сети (разрешить доступ через localhost и IP);
3) создать необходимые базы данных (БД) и пользователей.

c) Kubernetes-кластер:
   1) установить kubernetes на двух виртуальных машинах (используя kubespray);
   2) настроить кластер и проверить его работоспособность;
   3) установить Local Path Provisioner для динамического предоставления PV (Persistent Volumes) на узлах кластера.
d) S3-хранилище (MinIO):
   1) установить и настроить MinIO на выделенном сервере;
   2) создать необходимые бакеты для хранения данных;
   3) настроить доступы для пользователей и сервисов.
e) Сервер с GPU:
   1) установить NVIDIA драйверы, nvidia-container-toolkit и docker;
   2) настроить docker для работы с GPU;
   3) проверить работу GPU с помощью nvidia-smi и запуска тестового контейнера.

### 3.1.3. Развертывание компонентов в Kubernetes

Необходимо выполнить следующие шаги:

a) Загрузить образы:
   1) подготовить образы компонентов Ассистента для команд разработки (АКР), DWH, User Check в формате .tgz;
   2) загрузить подготовленные образы в Nexus после настройки Nexus как docker registry.
b) Nexus:
   1) развернуть Nexus в kubernetes;
   2) настроить proxy и hosted-репозитории;
   3) импортировать образы docker компонентов в Nexus.
c) Установить компоненты – развернуть компоненты АКР, DWH, User Check в Kubernetes, используя манифесты YAML и предварительно загруженные образы из Nexus.
d) Проверить корректность работы всех компонентов, а также доступность компонентов через веб-интерфейсы.

### 3.1.4. Настройка серверов и развёртывание LLM

Необходимо выполнить следующие шаги:

a) Настроить docker на GPU-сервере:
   1) установить и настроить docker для работы с nvidia-container-toolkit;
   2) проверить корректность настроек с помощью тестового GPU-контейнера.

b) Запустить компоненты LLM:
   1) загрузить из Nexus образы компонентов LLM на сервер с GPU;
   2) запустить компоненты LLM в docker, используя настройки для работы с GPU.

## 3.2. Порядок запуска

Запуск в работу любого компонента Системы необходимо выполнять в зависимости от типа установки, т.к. большинство компонентов находятся в kubernetes.

Запуск компонента, имеющего файл манифеста, необходимо выполнить следующим образом:

```
kubectl apply –f <manifest file yaml>
```

Если компонент разворачивается с помощью пакетного менеджера helm, то его запуск необходимо выполнить следующим образом:

```
helm install <имя helm релиза в kubernetes> <имя helm chart / репозитория> –f values.yaml —namesapce <название kubernetes пространства имен, куда необхоодимо развернуть компонент>
```

## 3.3. Порядок остановки

Для остановки работы компонента необходимо исходить из типа установки/удаления, т.к. большинство компонентов находится в kubernetes.

Остановку компонента, имеющего файл манифеста, необходимо выполнить следующим образом:

```
kubectl delete –f < manifest file yaml>
```

Если компонент разворачивается с помощью пакетного менеджера helm, то его

остановку/удаление необходимо выполнить следующим образом:

```
helm uninstall <имя helm релиза в kubernetes> –namesapce
<название kubernetes пространства имен, куда необходимо
развернуть компонент>
```

## 3.4.Порядок установки обновлений системы

В случае предоставления новой версии кодовой базы компонентов Системы их обновление производится в соответствии с инструкциями по их установке – повторяется процедура развертывания компонентов. Предыдущие версии компонентов Системы будут заменены обновленными в связи с использованием kubernetes.

Компоненты Системы, развертываемые с помощью docker, также проходят процедуру в рамках инструкции по их развертыванию. Однако рекомендуется остановить компоненты с помощью следующей команды:

```
docker compose down –remove-orphans
```

## 3.5.Порядок проверки работоспособности системы

Способы проверки и диагностирования работы каждого компонента указаны в соответствующем пункте документа «Инструкция по развертыванию и установке программного обеспечения», в котором приведено описание развертывания компонента.

## 3.6.Решения по диагностированию системы

Способы проверки и диагностирования работы каждого компонента указаны в соответствующем пункте документа «Инструкция по развертыванию и установке программного обеспечения», в котором приведено описание развертывания компонента.

# 4. Описание операций системы «Ассистент для команд разработки»

## 4.1.Управление учетными записями пользователей

### 4.1.1.Создание учетной записи пользователя

Для создания новой учетной записи необходимо обратиться к API:

```
POST /api/v1/register
Host: 10.10.10.12
Content-Type: application/json; charset=utf-8
{
    "admin_login": "admin",
    "admin_password": "password",
    "new_user_login": "new_user",
    "new_user_password": "user_password"
}
```

### 4.1.2. Обновление пароля учетной записи пользователя

Для обновления пароля учетной записи необходимо обратиться к API:

```
POST /api/v1/update_user
Host: 10.10.10.12
Content-Type: application/json; charset=utf-8
{
    "admin_login": "admin",
    "admin_password": "password",
    "user_login": "user",
    "user_new_password": "user_password"
}
```

### 4.1.3.Блокировка учетной записи пользователя

Для блокировки учетной записи необходимо необходимо обратиться к API:

```
POST /api/v1/delete_user
Host: 10.10.10.12
Content-Type: application/json; charset=utf-8
{
    "admin_login": "admin",
```

```
    "admin_password": "password",
    "login": "user_for_delete",
}
```

# 5. Планово-предупредительные мероприятия

Представленные процедуры предназначены для регулярной проверки работоспособности системы «Ассистент для команд разработки» и локализации неработающих компонентов. Регулярное проведение данных мероприятий (см. **Таблица 5-1**) позволяет своевременно выявлять и устранять неполадки, обеспечивая стабильную работу системы.

***Таблица 51*** *— Перечень планово-предупредительных мероприятий*

| № | Содержание мероприятия | Периодичность | Ответственный |
|---|---|---|---|
| **1** | Мониторинг работоспособности различных компонентов системы | Определяется Администратором<br>(рекомендованная частота: один раз в сутки, 5 дней в неделю) | Администратор |
| **2** | Локализация неработающих компонентов системы | Определяется Администратором | Администратор |
| **3** | Проверка наличия сертификатов | Перед запуском пилотной или целевой версии системы в промышленную среду | Администратор |
| **4** | Проверка в консоли администратора Kubernetes состояния подов (все поды запущены в требуемом количестве) | После запуска пилотной или целевой версии системы в промышленную среду | Администратор |
| **5** | Проверка лог-файлов запуска компонентов продукта (микросервисов) на наличие сообщений с типом «ERROR» | После запуска пилотной или целевой версии системы в промышленную среду | Администратор |
| **6** | Проверка работоспособности микросервисов на маршрутах (/healthcheck) | После запуска пилотной или целевой версии системы в промышленную среду | Администратор |
| **7** | Проверка работоспособности микросервисов | Ежедневно | Администратор |
| **8** | Проверка работоспособности компонента БД PostgreSQL | Ежедневно | Администратор |
| **9** | Проверка статуса запуска серверов | Ежедневно | Администратор |
| **10** | Проверка статуса мониторинга использования серверов | Раз в месяц | Администратор |

| № | Содержание мероприятия | Периодичность | Ответственный |
|---|---|---|---|
| **11** | Проверка статуса мониторинга выполнения задач планировщика | Ежедневно | Администратор |
| **12** | Мониторинг использования серверов: анализ статистики по лог-файлам | Раз в месяц | Администратор |
| **13** | Резервное копирование | Один раз в сутки в период с 22.00 до 08.00 по МСК, 5 дней в неделю | Администратор |

## 5.1.Документирование результатов

Порядок документирования результатов:

- Все результаты проверки работоспособности системы и локализации неработающих компонентов должны быть задокументированы;
- Документация должна содержать информацию о дате проверки, выявленных проблемах, принятых мерах и результате их применения;
- Документация должна храниться в системе контроля версий или в специализированной системе управления инцидентами;
- Это позволит отследить историю работы платформы и легко идентифицировать причины возникновения ошибок.

# 6. Аварийное восстановление системы «Ассистент для команд разработки»

Аварийно-восстановительные работы проводятся с целью поддержки бесперебойной работы системы и восстановления работоспособности системы в максимально сжатые сроки без потери данных.

## 6.1. Критерии аварийного события

1. Недоступность приложения «Ассистент для команд разработки».
2. Отказ одной из частей ИС.
3. Массовые обращения пользователей системы в службу поддержки.
4. Выявление неисправности в результате планово-предупредительных работ со стороны службы поддержки.
5. Обращение в службу поддержки со стороны сотрудников мониторинга, администраторов СУБД и серверов приложений.
6. Не пройдены проверки работоспособности при вводе системы в эксплуатацию.

К потенциальным аварийным событиям также можно отнести:

1. Недоступность серверов приложений.
2. Недоступность баз данных.
3. Некорректная работа системы.

## 6.2. Порядок действий при выявлении аварийного события

Общий порядок действий при возникновении аварийного события:

1. Диагностика возникшей ситуации, первичный сбор информации.
2. Оповещение координатора, ответственного за выполнение аварийного восстановления, если самостоятельное устранение проблемы невозможно.
3. Определение специфики проблемы, привлечение специалистов.
4. Выполнение действий согласно сценарию восстановлению системы .
5. Рассылка оповещений об аварийной ситуации:
   - всем заинтересованным лицам;
   - каналы оповещения выбираются на усмотрение службы поддержки в зависимости от критичности ситуации (например, электронная почта, SMS-сообщение, звонок);
   - в оповещении указывается следующая информация: данные о зарегистрированном инциденте (описание аварийного события, периметр

события, возможные причины возникновения, плановые сроки устранения, влияние на смежные ИС и возможные последствия).

К процессам поддержки и восстановления работоспособности системы должны быть привлечены следующие сотрудники:

- Системные администраторы;
- Администраторы серверов приложений (в части сопровождения и эксплуатации систем поддержки бизнеса).

## 6.3.Общие рекомендации по восстановлению деятельности системы «Ассистент для команд разработки»

1. Проверить статус работоспособности приложения, при необходимости, перезапустить.
2. Закончилось место на диске - необходимо освободить место при помощи архивирования и удаления данных с дисков.
3. Истек срок действия пароля для ТУЗ - необходимо актуализировать пароль для ТУЗ.
4. Сломался сервер с виртуализацией - необходимо проанализировать ситуацию и устранить проблему.