ОПИСАНИЕ ПРОЦЕССОВ, ОБЕСПЕЧИВАЮЩИХ ПОДДЕРЖАНИЕ ЖИЗНЕННОГО ЦИКЛА

СИСТЕМЫ BERGEN INTELLIGENT PROCESS AUTOMATION

19 Листов

2021

Оглавление

Термины и сокращения 3

1 Жизненный цикл процессов реализации (разработки) Системы..... 4

1.1 Ландшафт ведения разработки..... 4

1.2 Требования к оформлению программы 4

1.3 Регламент ведения разработки и тестирования 5

1.4 Работа с Git..... 6

1.5 Сборка ПО 8

1.6 Порядок выпуска релизов..... 9

2 Поддержание жизненного цикла Системы 10

2.1 Данные о персонале, задействованном в процессе разработки и тестирования 11

2.2 Информация о процессе сопровождения и поддержки 13

3 Устранение неисправностей, выявленных в ходе эксплуатации Системы..... 13

4 Совершенствование Системы..... 15

5 Техническая поддержка Системы 16

5.1 Данные о персонале, задействованном в процессе техподдержки **..... 16**

Термины и сокращения

Сокращение	Полное наименование
ЦОД	Центр обработки данных
DEV	Сервер разработки
QA	Сервер тестирования
PROD	Продуктивный сервер
RM	RedMine (Система управления задачами проекта)
РП	Руководитель проекта
АДМ	Системный администратор проекта
АРП	Архитектор проекта
БД	База данных
ИБ	Информационная безопасность
Заказчик	Физическое или юридическое лицо, использующее Систему на договорной основе
ПО	Программное обеспечение
ПП	Программный продукт
Разработчик	Правообладатель Системы
СУБД	Система управления базами данных
IDE	Интегрированная среда разработки, система программных средств, используемая программистами для разработки программного обеспечения.

1 Жизненный цикл процессов реализации (разработки) Системы

1.1 Ландшафт ведения разработки

Ландшафт системы разработки представлен в виде следующих сред, которые на практике будут являться просто набором виртуальных машин, однако, у каждого из них будет свое особенное предназначение:

- DEV (сервер разработки);
- QA (сервер тестирования);
- PROD (продуктивный сервер).

Процесс разработки имеет следующий порядок:

- Разработчики работают над новой feature (задачей) и делают отладку в своей DEV среде;
- Разработчики проверяют и тестируют готовые feature (задачи), собранные в DEV репозитории, в QA среде;
- Разработчики выпускают новый релиз, и ответственный делает миграцию успешного релиза в окружение PROD.

1.2 Требования к оформлению программы

Программа разрабатывается в средах:

- Visual Studio Code;
- Vim;

Настройки лежат в репозиториях проекта Bergen IPA.

Перечень репозиториев проекта Bergen IPA представлен ниже, см. Таблица 1

Таблица 1 – Список репозиториев

Репозиторий в Git	Ссылки	Сервер разработки	Сервер тестирования	Продуктивный сервер
api-gw	https://gitlab.bergen.tech/ips/api-gw/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
base-lib	https://gitlab.bergen.tech/ips/base-lib/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
camunda-wrapper	https://gitlab.bergen.tech/ips/camun	10.7.10.18	10.7.10.19	10.7.10.20

	da-wrapper/-/tree/ros			
docker-compose	https://gitlab.bergen.tech/ips/docker-compose/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
file-upload	https://gitlab.bergen.tech/ips/file-upload/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
ips-ui	https://gitlab.bergen.tech/ips/ips-ui/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
keycloack-role-wrapper	https://gitlab.bergen.tech/ips/keycloack-role-wrapper/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
ml-runner	https://gitlab.bergen.tech/ips/ml-runner/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
model-service	https://gitlab.bergen.tech/ips/model-service/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
platforma-api	https://gitlab.bergen.tech/ips/platforma-api/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
reference-book	https://gitlab.bergen.tech/ips/reference-book/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20
reporting	https://gitlab.bergen.tech/ips/reporting/-/tree/ros	10.7.10.18	10.7.10.19	10.7.10.20

1.3 Регламент ведения разработки и тестирования

Регламент ведения разработки представлен ниже:

a) Репозиторий с исходным кодом хранится локально, с возможностью доступа через сеть Internet;
b) Git-репозитории разделены по типам серверов;
c) Каждый репозиторий един;

d) В core git-репозиториях должны отсутствовать:

- Генерируемые файлы и результаты сборки, включая документацию, логи, upload файлы;
- node_modules и другие скачиваемые библиотеки;
- Любые пароли, сертификаты, API ключи;
- Настройки локального окружения, IDE.

e) Для документации по всем репозиториям создаётся отдельный репозиторий, который обновляется при необходимости. Каждый метод должен быть задокументирован в системе документации (Jsdoc, Doxygen или иной);

f) Пароли, сертификаты и ключи доступа могут быть сохранены в «личном» репозитории только при шифровании с помощью публичного ключа соответствующего получателя;

g) По умолчанию для проверки оформления кода на соответствие проектным стандартам запускается linter;

h) Разработчик отвечает за отсутствие merge-конфликтов при слиянии в dev веток;

i) Контроль технической документации и код-ревью перед переносом в PROD осуществляет архитектор проекта;

j) MR (merge request) в DEV ветку создает разработчик, выполняющий задачу. Само выполнение MR должен делать не тот разработчик, который создал MR. Разработчик, который создал MR должен отправить ссылку на MR проверяющему. MR обрабатываются по порядку в сторону увеличения номера. При наличии замечаний MR отклоняется. Задача переводится на сотрудника, который делает MR.

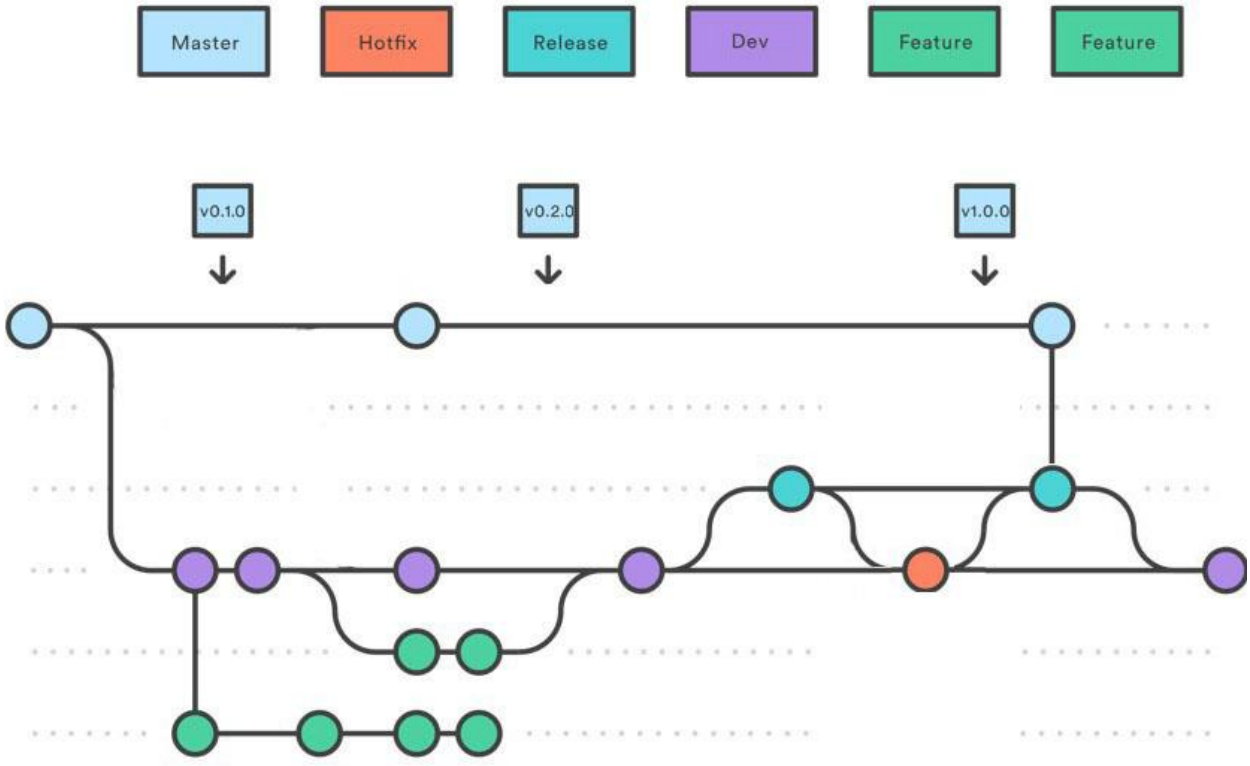
1.4 Работа с Git

Работа с репозиторием производится в соответствии с регламентом, описанным в руководстве разработчика:

- master: в данной ветке хранится продакшн-версия проекта, при этом изменения в этой ветке не производятся напрямую, а вливаются из develop ветки.
- dev: в данной ветке интегрируются изменения, внесённые всеми разработчиками. При этом разработчики не могут вносить изменения непосредственно в этой ветке, а должны переносить в неё изменения из веток feature с помощью merge request.
- release/release-x.x.x: как только dev ветка набрала в себя достаточно изменений и релиз менеджер считает, что пора выпустить

новую версию, необходимо ответвится от dev в release. С этого момента, в release ветку больше нельзя добавлять никаких изменений функционала, исключение составляют только срочная правка ошибок (fix ветки). Как только release бранч готов к эксплуатации, ему присваивается версия, и через merge request он сливается с master веткой.

- feature/<name>: каждая такая ветка создаётся отдельным разработчиком при добавлении функционала. Одна задача - одна ветка. Не допускать выполнения многих задач в одной ветке, так как это затруднит последующий merge. Разработчик вносит изменения в код только на ветке feature. После выполнения работ на данной ветке разработчик переносит изменения в develop через merge request. При выполнении большой задачи в течении долгого времени обязательно каждые 1-2 дня мерджить к себе основную ветку разработки, чтобы избежать накопления большой разницы в коде и тяжело решаемых конфликтов изменений.
- fix/<name>: ветки для внесения исправлений в release ветки. Могут создаваться только от release веток. После исправления вливается через merge request в ту же ветку от которой была создана и дополнительно в master и dev ветку, чтобы актуальный репозиторий содержал в себе все исправления.



Разработчик может выполнять commit в ветку ros - * самостоятельно. Для включения разработки в release, разработчик создаёт merge request в ветку ros. Подтверждение (approve) merge request в ветку ros должно осуществляться не тем разработчиком, кто создал merge request.

- Ветка ros формируется через merge request.
- Ветка ros содержит последнюю рабочую версию разрабатываемого ПО.
- При создании merge request необходимо выбрать ветку в которой шла разработка и целевую ветку, см. Рисунок 3

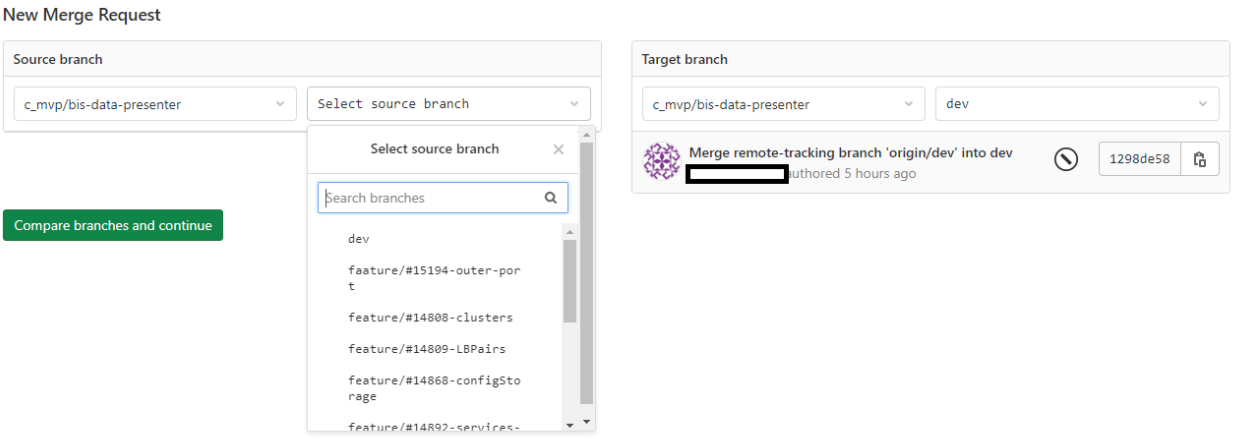


Рисунок 1 – Выбор ветки разработки

В настройках необходимо указать Approvals. Список людей, кто будет подтверждать merge. Это должен быть человек с ролью АРП. Если необходимо, то выставляем галочку для удаления ветки разработки, после подтверждения слияния, см. Рисунок 2:

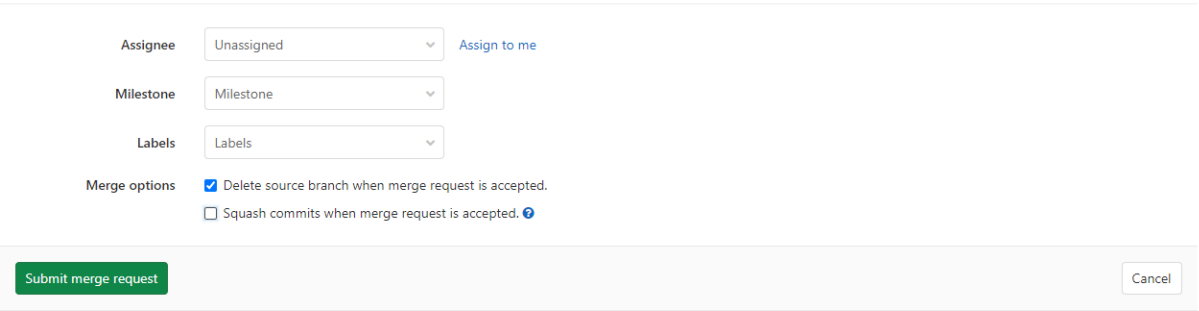


Рисунок 2 – Простановка чекбокса удаления ветки

1.5 Сборка ПО

Описание и состав сборки ПО, см. Таблица 2.

Таблица 2 – Описание и состав сборки ПО

Имя файла или каталога	Описание
apps	Каталог - содержит docker-compose файлы описывающие настройки запуска контейнеров приложения
services	Каталог - содержит docker-compose файлы, описывающие настройки запуска контейнеров баз данных файлового хранилища и других сервисов, требующихся для обеспечения работы приложения
envs	Каталог – содержит файлы, в которых можно отредактировать переменные, используемые для установки Системы в целом.
docker-compose.yml	Основной docker-compose - файл указывающий контекст запуска приложения, а также настройки внутренней сети docker
Makefile	Файл описывающий автоматический запуск или остановку приложения, а так же реализующий авторизацию на серверах registry.bergen.tech для получения docker образов системы.

1.6 Порядок выпуска релизов

Порядок выпуска релизов представлен на см. Рисунок 3:

Время формирования Release среда 16:00-17:00. По согласованию release (перенос на QA) можно выполнять каждый день с 16:00 до 17:00.

- DEV сервер

На этот сервер разворачивается Система средствами CI/CD Gitlab из веток:

https://gitlab.bergen.tech/ips/deploy

- QA сервер

Обновляется только в среду 16:00-17:00, далее на протяжении остальных дней доступен для тестировщиков без изменений. То есть по факту релизная версия делается в среду, а в четверг принимается решение о выпуске релиза на

продуктивный сервер. В среду допускается перенос фиксов в утреннее время с 9 до 10 по МСК.

- PROD сервер

В пятницу утром осуществляется перенос на PROD. Перенос осуществляется запуском shell-скрипта на сервере. Обновление производит только системный администратор проекта.

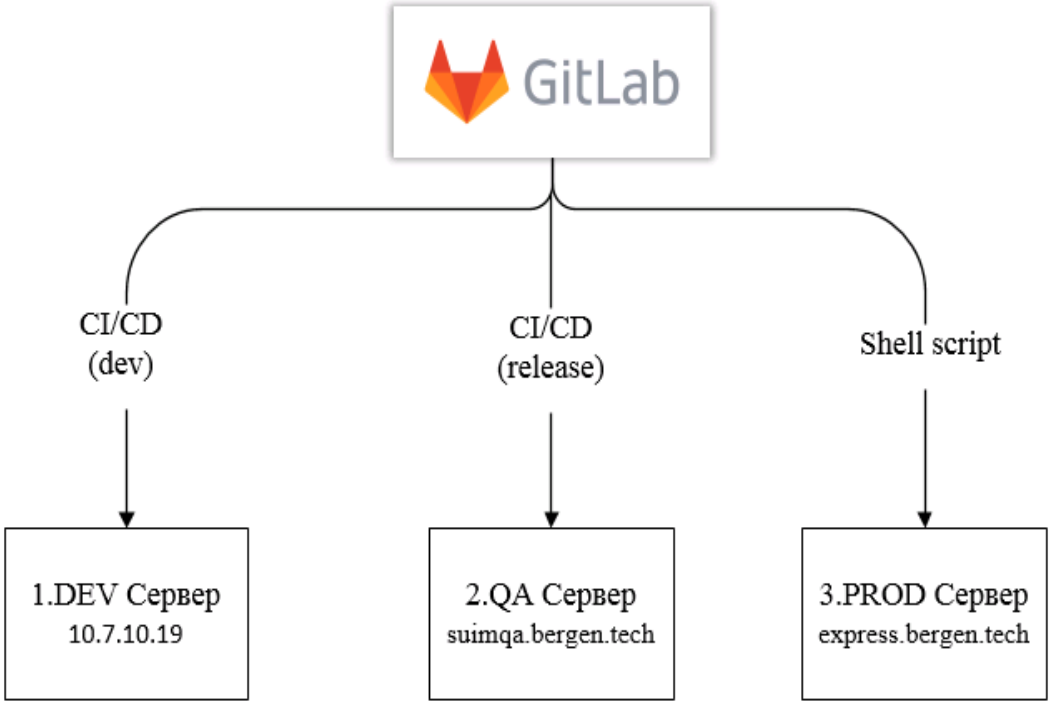


Рисунок 3 – Порядок выпуска релизов

2 Поддержание жизненного цикла Системы

Поддержание жизненного цикла Системы осуществляется за счет сопровождения Системы и включает в себя проведение модернизаций Системы в соответствии с собственным планом доработок и по заявкам клиентов, консультации по вопросам установки и эксплуатации (по электронной почте) Системы.

В рамках технической поддержки Системы оказываются следующие услуги:

- Помощь в установке Системы;
- Помощь в настройке и администрировании;
- Помощь в установке обновлений Системы;
- Помощь в поиске и устранении проблем в случае некорректной установки обновления Системы;
- Пояснение функционала модулей Системы, помощь в эксплуатации Системы.

2.1 Данные о персонале, задействованном в процессе разработки и тестирования

В процессе разработки и тестирования задействовано 15 штатных сотрудников. Данные о персонале, задействованном в процессе разработки и тестирования, представлены в таблице ниже, см. Таблица 3:

Таблица 3 – Данные о персонале, задействованном в процессе разработки и тестирования

Отдел	Должность
Администрация	Директор по консалтингу
Администрация	Системный администратор
Администрация	Администратор проекта
Администрация	Специалист по кадрам
Администрация	Руководитель проекта
Отдел разработки программного обеспечения	Младший разработчик 2 уровня
Отдел разработки программного обеспечения	Младший разработчик 3 уровня
Отдел разработки программного обеспечения	Старший разработчик 2 уровня
Отдел разработки программного обеспечения	Старший разработчик 5 уровня
Отдел разработки программного обеспечения	Старший разработчик 5 уровня
Отдел бизнес и системной аналитики	Аналитик 5 уровня
Отдел бизнес и системной аналитики	Аналитик 1 уровня
Отдел бизнес и системной аналитики	Тестировщик 1 уровня

Отдел	Должность
Отдел бизнес и системной аналитики	Руководитель отдела и Архитектор
Отдел разработки программного обеспечения	Руководитель разработки

С точки зрения разработки в процессе создания, развития и ввода в эксплуатацию компонентов системы участвуют следующие роли, см. Таблица 4.

Таблица 4 – Роли и зоны ответственности ролей

Роль	Зона ответственности
Руководитель проекта	Ответственный за организацию работ в рамках проектной задачи.
IT-лидер	Ответственный за техническую и процессную составляющую разработки Системы. Ответственный за техническую архитектуру разрабатываемой Системы.
Администратор (АДМ)	Ответственный за организацию и поддержку ландшафта системы, ведение бэкапов, ведение прав для разработчиков, настройку интеграции с внешними системами.
	Ответственный за сборку релиза и заливку на QA и PROD.
Разработчик	Ответственный за разработку в рамках назначенных задач.
Тестировщик	Ответственный за тестирование и качество разрабатываемой Системы.
Специалист службы поддержки	Ответственный за принятие заявок от заказчика и консультирование Пользователей Системы.

Фактический адрес, по которому осуществляется процесс разработки и тестирования заявляемого ПО:

Россия: г. Москва, ул. Мясницкая, дом 38, стр. 1.

2.2 Информация о процессе сопровождения и поддержки

Данные о возможных средствах коммуникации и о режиме работы службы поддержки:

- Коммуникация со службой поддержки ведется по электронной почте по адресу: support@bergen.tech или по телефону 8 (495) 664-37-83;
- Служба поддержки работает по будним дням с 10:00 до 19:00 по МСК;

В процессе сопровождения задействовано 4 штатных сотрудника. Данные о персонале, задействованном в процессе сопровождения, представлены в таблице ниже, см. Таблица 5.

Таблица 5 – Данные о персонале, задействованном в процессе сопровождения

Отдел	Должность
Администрация	Системный администратор
Отдел бизнес и системной аналитики	Ведущий аналитик-исследователь
Отдел разработки программного обеспечения	Старший специалист разработчик 5 уровня
Отдел техподдержки	Ведущий специалист службы поддержки 4 уровня

Фактический адрес, по которому осуществляется процесс сопровождения: Россия: г. Москва, ул. Мясницкая, дом 38, стр. 1.

Для работы с Системой пользователю необходимо ознакомиться с Руководством пользователя.

Пользователи Системы должны обладать навыками работы с персональным компьютером и уметь пользоваться браузером.

3 Устранение неисправностей, выявленных в ходе эксплуатации Системы

Неисправности, выявленные в ходе эксплуатации Системы, могут быть исправлены двумя способами:

- Массовое плановое обновление компонентов Системы;
- Единичная работа специалиста службы технической поддержки по запросу пользователя.

В случае возникновения неисправностей в Системе, или необходимости в её доработке, Заказчик направляет Разработчику запрос. Запрос должен содержать тему запроса, суть (описание) и по мере возможности снимок экрана со сбоем (если имеется сбой).

Запросы могут быть следующего вида:

– Наличие Инцидента – произошедший сбой в системе у одного Пользователя со стороны Заказчика;
– Наличие Проблемы – сбой, повлекший за собой остановку работы/потерю работоспособности Программы;
– Запрос на обслуживание – запрос на предоставление информации;
– Запрос на развитие – запрос на проведение доработок Программы.

Запрос направляется Заказчиком либо Пользователями Заказчика через запрос в службу поддержки по электронной почте на электронный адрес support@bergen.tech

Пример:
– Пользователь Заказчика обнаружил проблему и зафиксировал ее письмом на службу поддержки Системы;
– Служба поддержки создает в системе управления проектом RedMine задачу с номером №100000 и названием «Доработка backend части авторизации»; указывается критичность задачи исходя из влияния проблемы на работоспособность Системы; задача назначается на руководителя разработки, имеющего проектную роль Архитектора проекта (АРП); статус задачи «Открытая»;
– Руководитель разработки назначает задачу №100000 на аналитика для уточнения постановки или дополняет постановку самостоятельно и назначает задачу сразу на разработчика;
– Когда разработчик приступает к задаче, то должен изменить статус задачи на «В Работе» и создать ветку от dev с названием «feature/<name>-100000», в котором он будет вести свою работу. В данном примере <name> задается семантически по смыслу задачи;
– После окончания разработки и локального тестирования, и проверки на DEV сервере разработчик должен создать merge request и влить изменения в ветку dev. Merge request подтверждает сам разработчик. Текст коммита должен соответствовать следующему формату:

- «#NNNNN <Описание изменений>», где NNNNN – номер задачи в RedMine;
- В тексте коммита желательно придерживаться обезличенных формулировок. То есть, «исправлена проблема, связанная с ….», «добавлен функционал, позволяющий …..»;

– Далее разработчик или администратор (по запросу разработчика) запускает pipeline для разворачивания задачи на QA сервере и разработчик переводит задачу в статус «В тесте» и назначает на ответственного тестировщика. Статус задачи «В тесте» означает, что разработка закончена, локально протестирована, код смержен в dev ветку и развернут на QA сервере;

– Тестировщик проверяет задачу на QA сервере и если тест пройден, то ставит статус «В релиз» и назначает на Администратора (АДМ). Если тест не пройдет, возвращает разработчику, ставит статус «На доработке» и оставляет комментарий с описанием проблемы;

– После формирования релиза Администратор переводит задачи из статуса «В релиз» в статус «Решена». То есть статус «Решена» означает, что код находится в release ветке, был успешно протестирован на QA сервере и готов к развороту на конечном PROD сервер;

– «Решена» – код находится в release ветке, успешно был протестирован на QA сервере и готов к развороту на конечном PROD сервер;

– Администратор Системы по договоренности с Заказчиком обновляет Систему Заказчика.

4 Совершенствование Системы

Система регулярно развивается: в ней появляются новые дополнительные возможности, оптимизируется нагрузка ресурсов ПК, обновляется интерфейс.

В процессе совершенствования задействовано 5 штатных сотрудника. Данные о персонале, задействованном в процессе совершенствования, представлены в таблице ниже, см. Таблица 6.

Таблица 6 – Данные о персонале, задействованном в процессе совершенствования

Отдел	Должность
Администрация	Системный администратор
Отдел бизнес и системной аналитики	Ведущий аналитик-исследователь
Отдел разработки программного обеспечения	Старший специалист разработчик 2 уровня
Отдел разработки программного обеспечения	Старший специалист разработчик 5 уровня
Отдел разработки программного обеспечения	Старший специалист разработчик 5 уровня

Фактический адрес, по которому осуществляется процесс совершенствования: Россия: г. Москва, ул. Мясницкая, дом 38, стр. 1.

Пользователь может самостоятельно повлиять на совершенствование продукта, для этого необходимо направить предложение по усовершенствованию на электронную почту технической поддержки по адресу support@bergen.tech.

Предложение будет рассмотрено и, в случае признания его эффективности, в Систему будут внесены соответствующие изменения.

5 Техническая поддержка Системы

Для оказания технической поддержки Системы, Пользователи Системы могут направлять возникающие вопросы на электронную почту технической поддержки по адресу support@bergen.tech или по телефону 8 (495) 664-37-83. Техническая поддержка осуществляется в будние дни с 10:00 до 19:00 по МСК.

5.1 Данные о персонале, задействованном в процессе техподдержки

В процессе техподдержки задействовано 3 штатных сотрудника. Данные о персонале, задействованном в процессе техподдержки, представлены в таблице ниже, см. Таблица 7.

Таблица 7 – Данные о персонале, задействованном в процессе техподдержки

Отдел	**Должность**
Администрация	Системный администратор

Отдел	**Должность**
Отдел разработки программного обеспечения	Сетевой инженер
Отдел разработки программного обеспечения	Старший разработчик 2 уровня

Фактический адрес, по которому осуществляется процесс техподдержки: Россия: г. Москва, ул. Мясницкая, дом 38, стр. 1.

Лист регистрации изменений									
Изм.	Номера листов (страниц)				Всего листов (стр.в докум.)	№ докумен-та	Входящий № сопро-водительного документа и дата	Под-пись	Да та
	изме-ненных	заме-ненных	новых	аннули-рованных					

Составили

Версия	Дата	Фамилия, имя, отчество	Должность исполнителя	Подпись
1.0	29.07.2021.	Суркина Лилия	Аналитик	
1.0	29.07.2021	Жарков Анатолий	Руководитель разработки	
1.0	29.07.2021	Вострухина Елизавета	Аналитик	
1.1	04.08.2021	Суркина Лилия	Аналитик	
1.1	04.08.2021	Вострухина Елизавета	Аналитик	