НИЯУ МИФИ

РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА ДЛЯ УПРАВЛЕНИЯ ПРОЦЕССАМИ ОБРАБОТКИ ПЕРВИЧНЫХ ДАННЫХ ЭКСПЕРИМЕНТА SPD

Студент А. В. Плотников

Научный консультант Д. А. Олейник

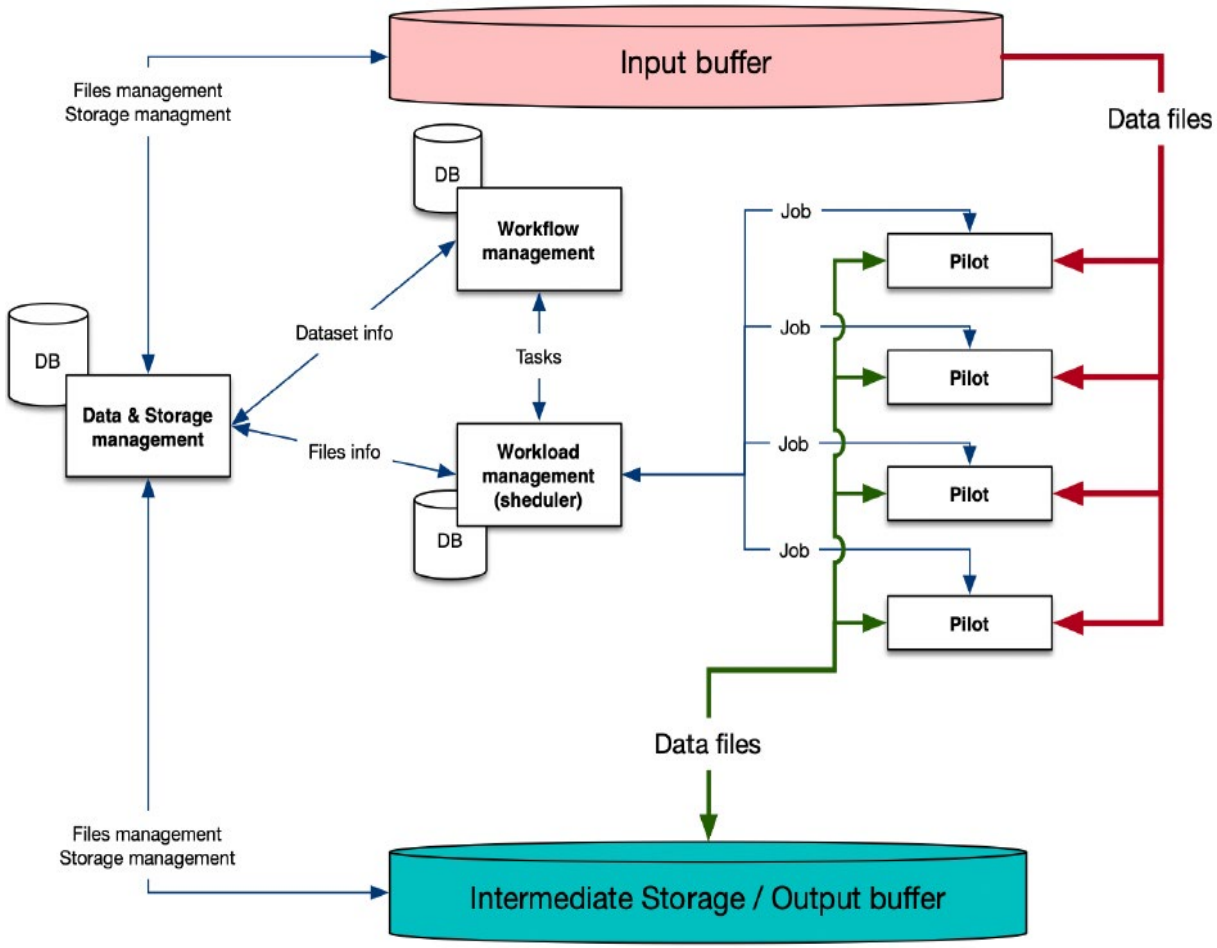
Москва 2025

SPD Online Filter

SPD Online Filter — специализированная программно-аппаратная система, предназначенная для предварительной обработки данных эксперимента SPD.

Система реализует многоступенчатую, высокопропускную методику обработки.

Основная, но не единственная, **цель обработки**: существенно уменьшить объём данных для последующего анализа и долговременного хранения.

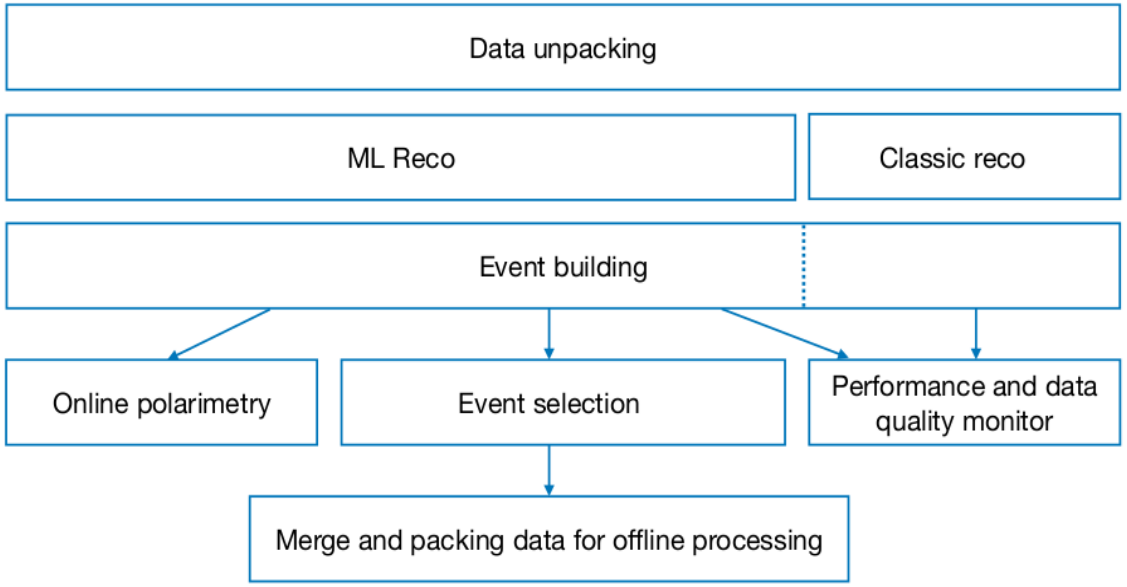


Архитектура Online Filter

Управление процессами обработки данных

Многоступенчатая обработка — набор последовательных этапов обработки данных. При этом на каждом этапе может обрабатываться достаточно большой объем данных.

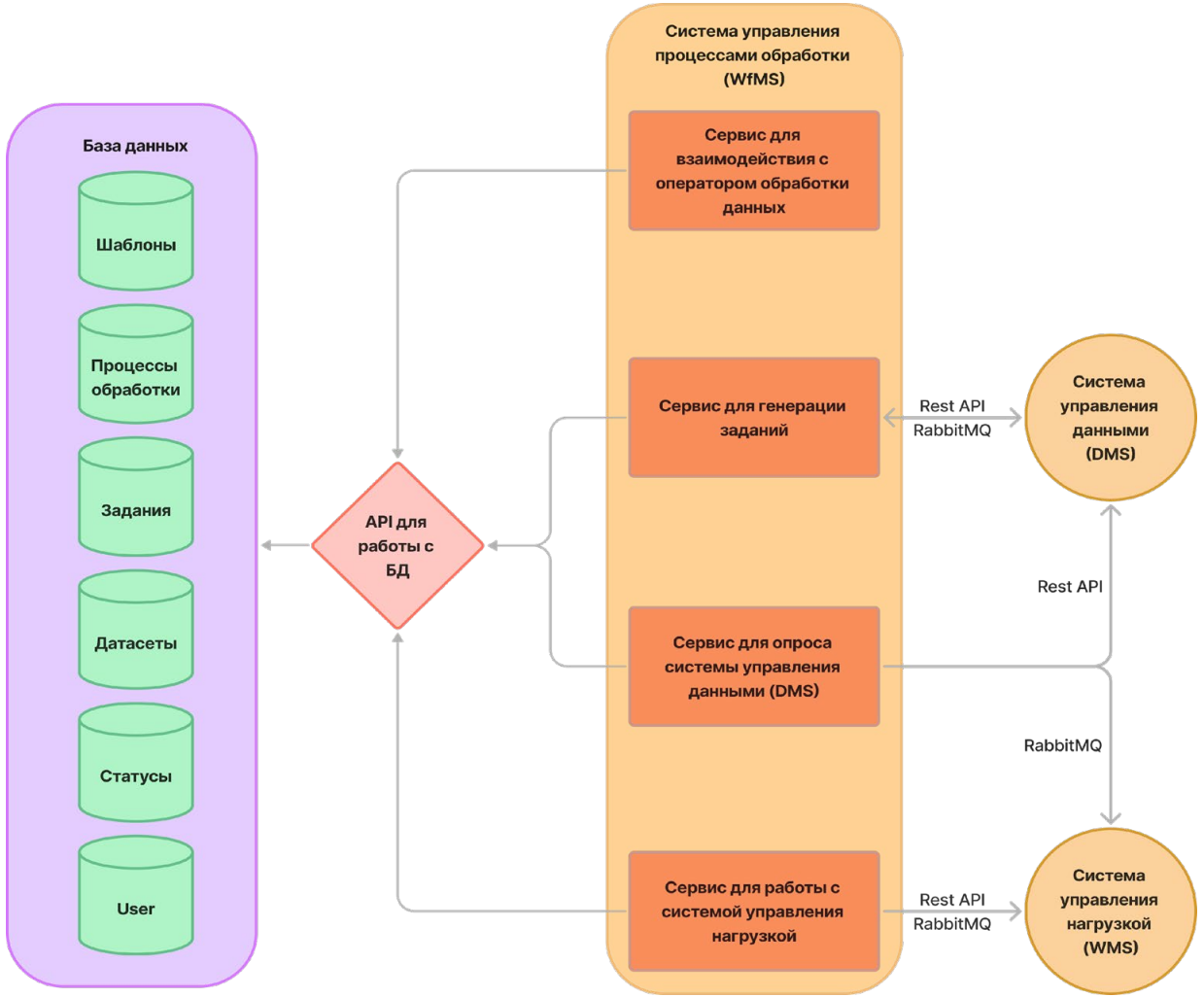
Каждый шаг, кроме первого, принимает данные, обработанные на предыдущем этапе, и передает результаты следующему, обеспечивая преобразование информации из одного представления в другое.



Пример последовательных шагов обработки

Система управления процессами обработки (WfMS)

Основная цель системы:
Декларирование формализованного описания процессов обработки. Управление и контроль выполнения параллельных процессов обработки.



Архитектура WfMS

Взаимодействия микросервисов WfMS с БД

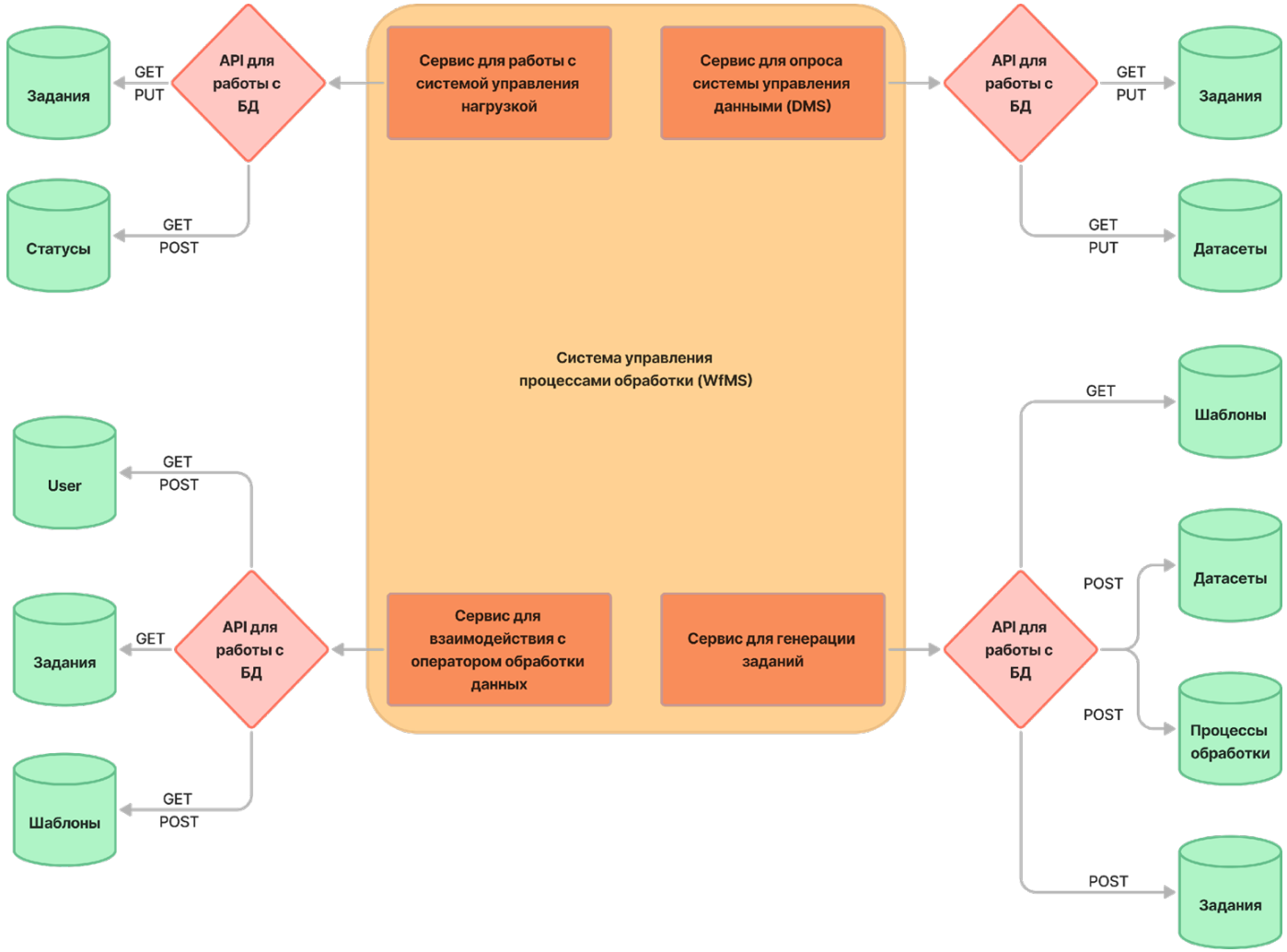
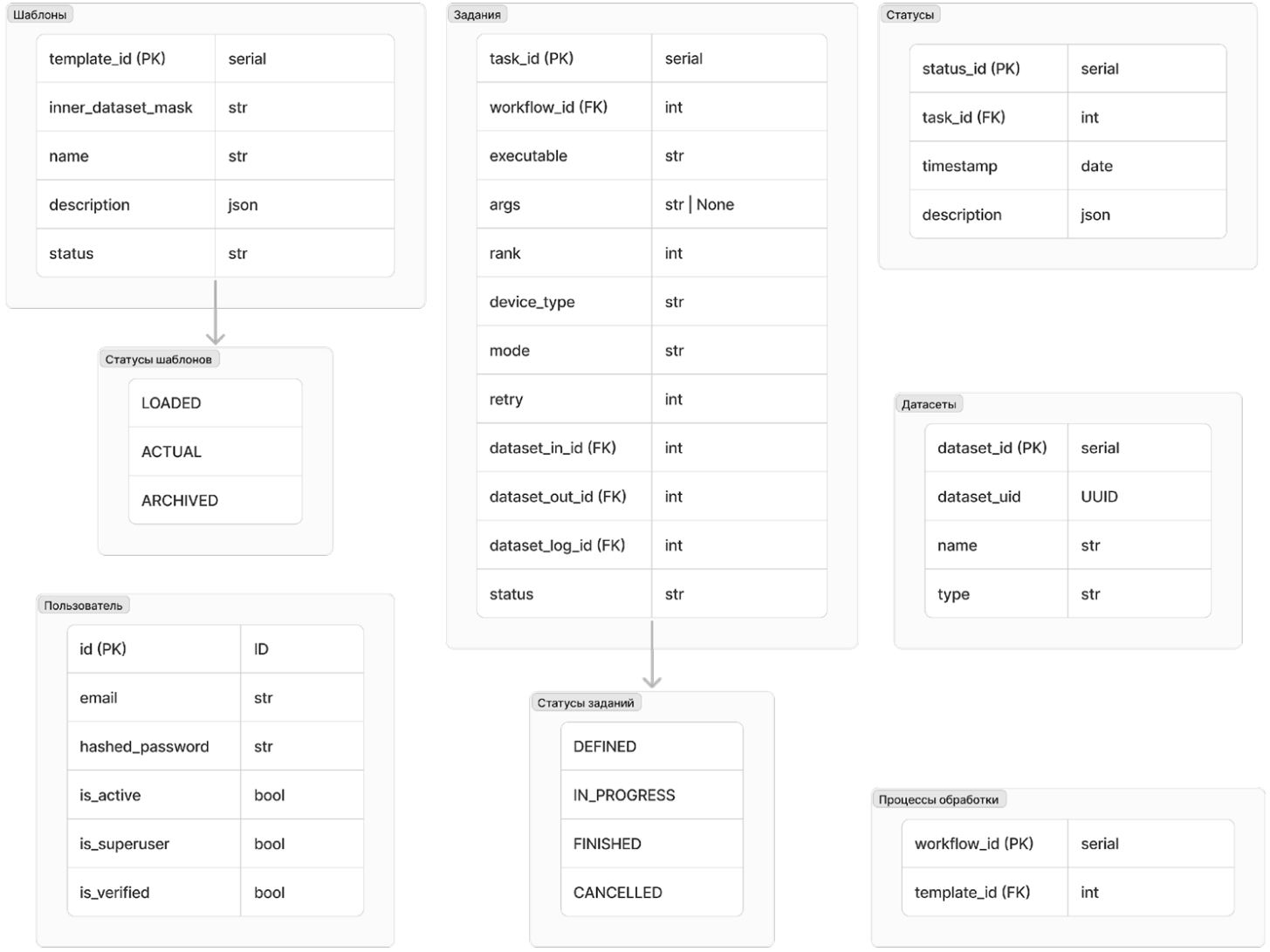


Схема взаимодействия микросервисов с БД

- База данных PostgreSQL развернута на отдельной виртуальной машине.
- Взаимодействие с базой данных осуществляется через REST API.

- SQLAlchemy ORM
- +
- асинхронные сессии и asyncpg
- +
- миграции Alembic

Структура базы данных



API к БД

Templates

- GET `/template/all` Get All Templates
- GET `/template/actual` Get Actual Templates
- GET `/template/{template_id}` Template Response
- POST `/template/create` Create Template
- PUT `/template/{template_id}/change` Change Template
- DELETE `/template/{template_id}/delete` Delete Template

Datasets

- GET `/dataset/all` Get All Datasets
- GET `/dataset/{dataset_id}` Dataset Response
- POST `/dataset/create` Create Dataset
- PUT `/dataset/{dataset_id}/dataset_uid` Change Rank

Workflows

- GET `/workflow/all` Get All Workflows
- GET `/workflow/{workflow_id}` Workflow Response
- POST `/workflow/create` Create Workflow

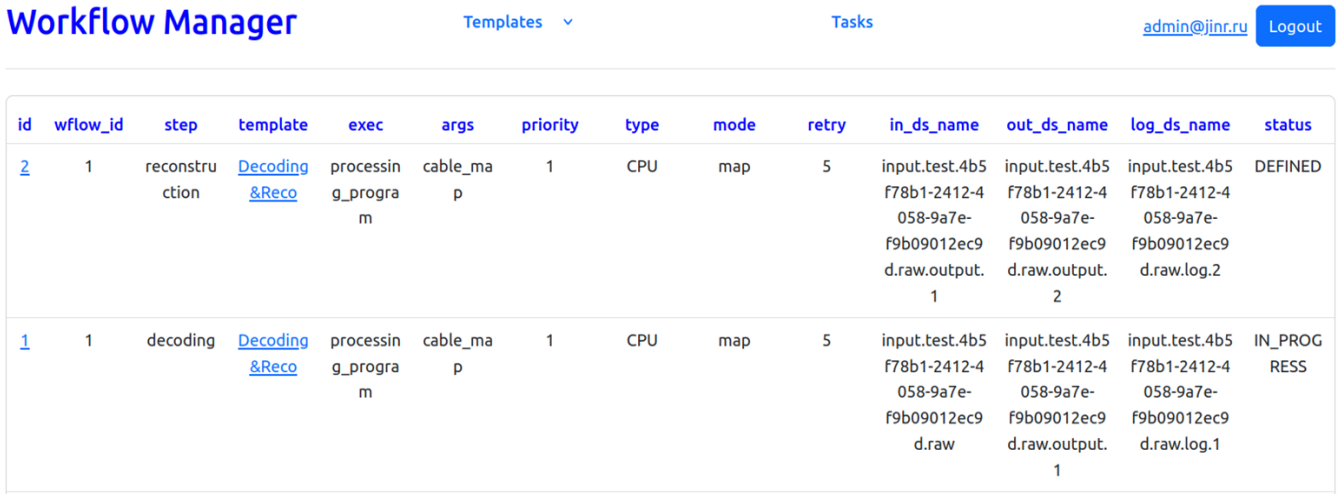
Tasks

- GET `/task/all` Get All Tasks
- GET `/task/{task_id}` Task Response
- GET `/task/status/{status_name}` Get Defined Tasks
- POST `/task/create` Create Task
- PUT `/task/{task_id}/rank` Change Rank
- PUT `/task/{task_id}/status` Change Rank
- DELETE `/task/{task_id}/delete` Delete Task

Сервис для взаимодействия с оператором обработки данных

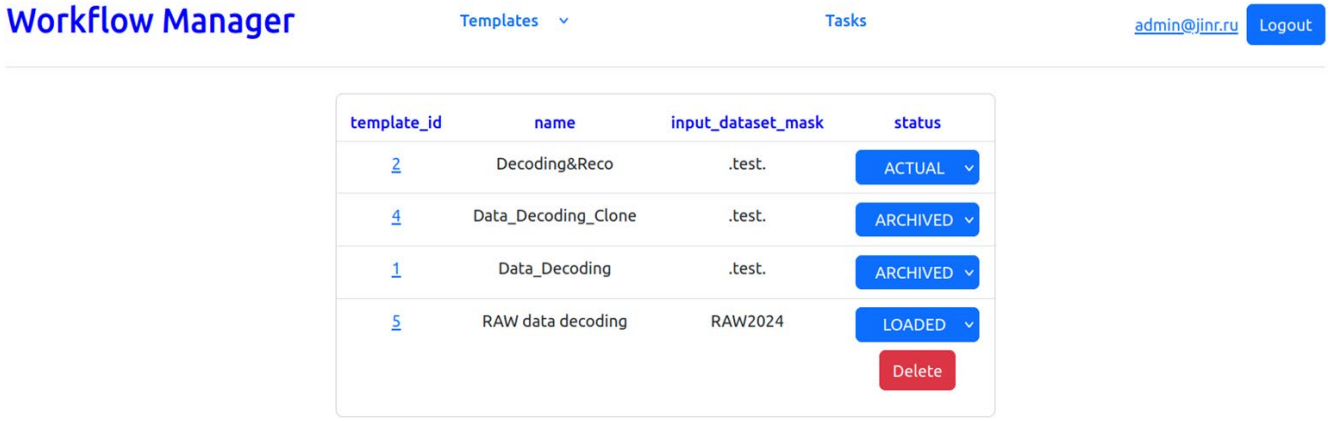
Основной функционал сервиса

1. Регистрация и авторизация пользователей с разными правами;
2. Вывод CWL-шаблонов;
3. Вывод заданий;
4. Создание CWL-шаблонов суперпользователями;
5. Предварительная валидация и запись в БД CWL-шаблонов;
6. Изменение статусов шаблонов суперпользователями.



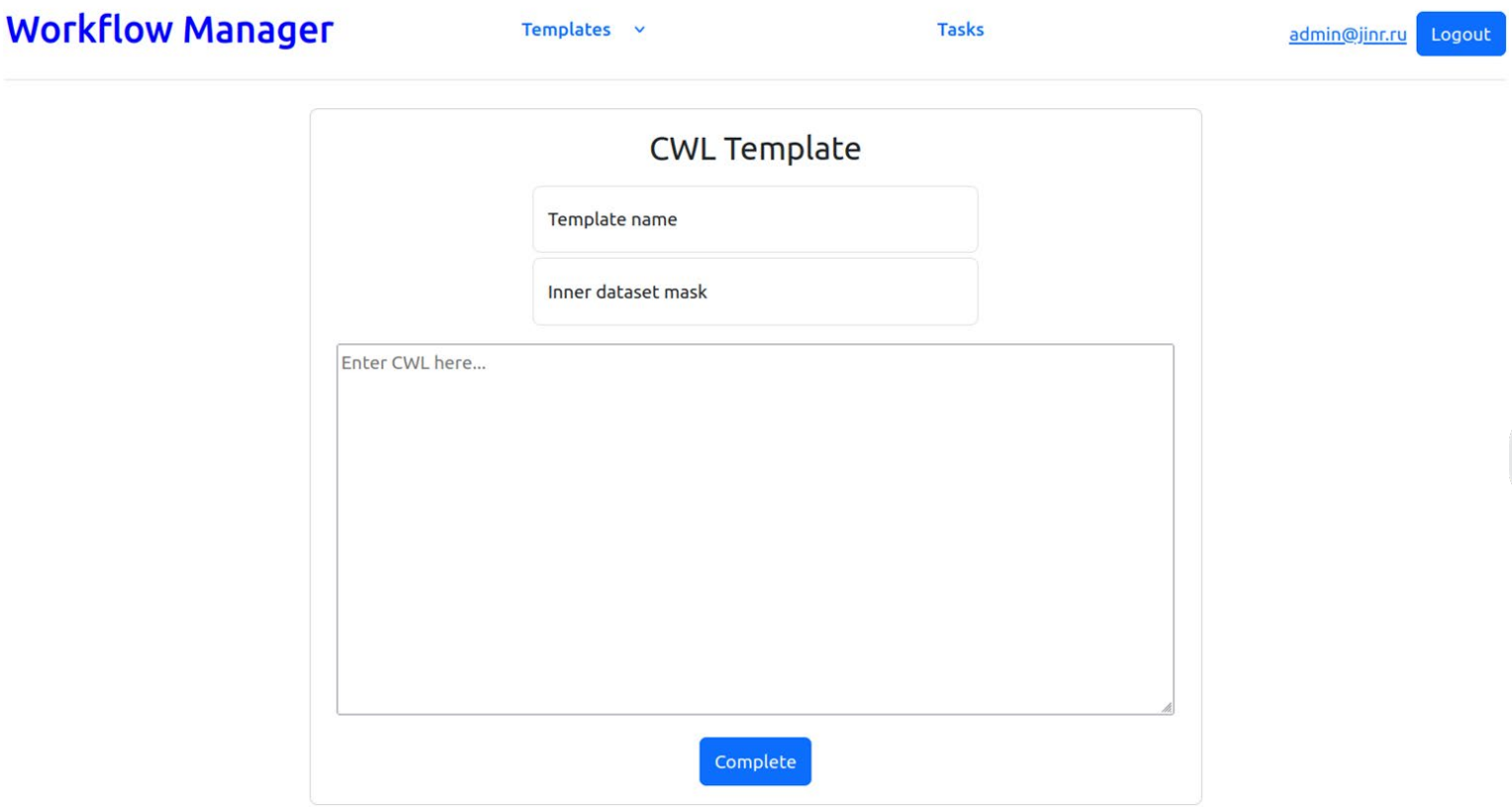
- Пользовательский интерфейс

- Backend: FastAPI
- Шаблонизатор: Jinja2
- Frontend: Bootstrap (HTML + CSS + JS)



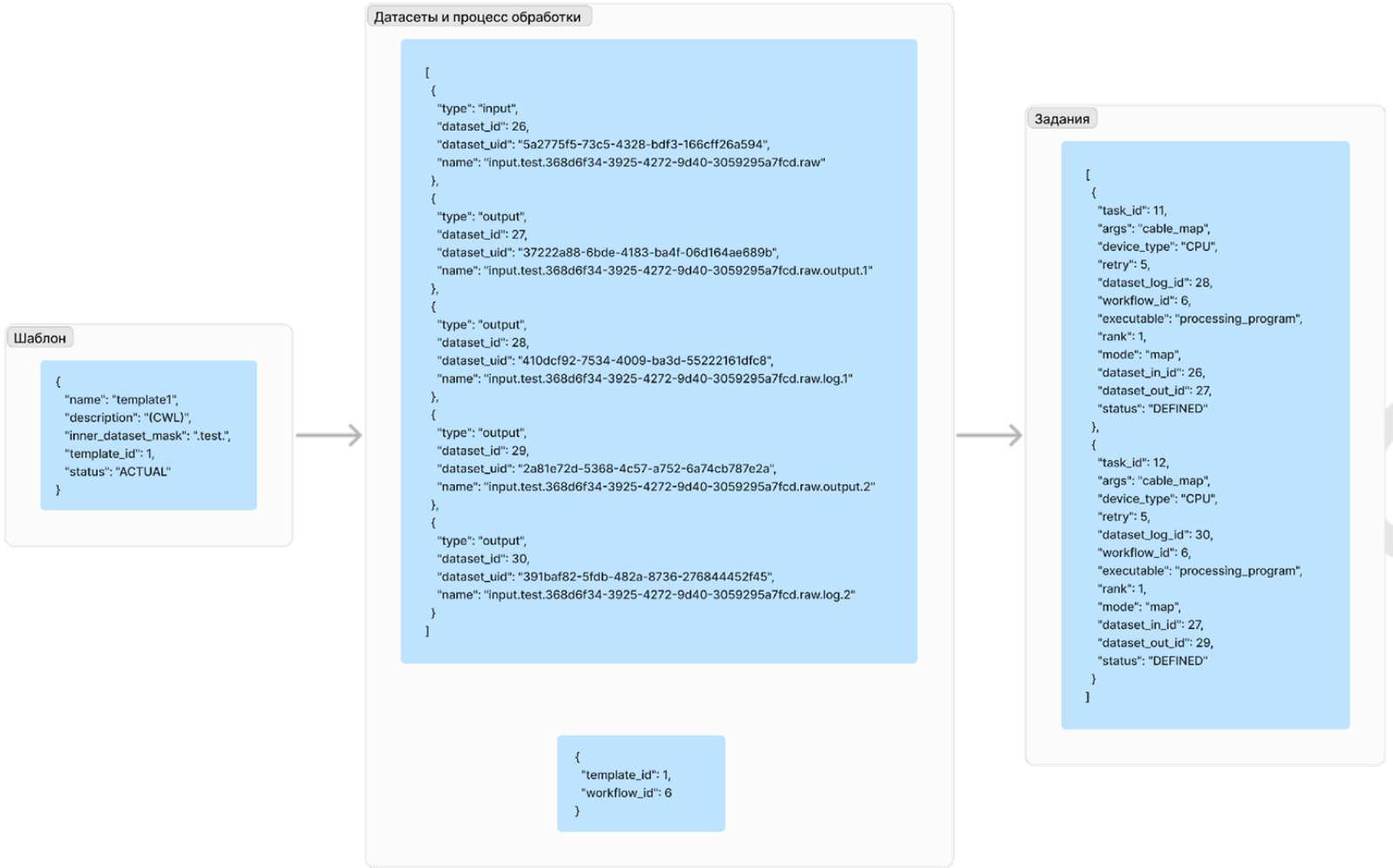
Создание шаблонов

- Создание шаблонов доступно только суперпользователям (в частности, отсутствует кнопка в интерфейсе)
- Предварительная валидация шаблонов с помощью инструментов cwltools
- Сохранение шаблона в БД



Сервис для генерации заданий

1. Получение зарегистрированных датасетов от DMS из RabbitMQ;
2. Сопоставление датасета по маске имени с нужным шаблоном;
3. Регистрация входного датасета в системе;
4. Создание процесса обработки по шаблону;
5. Создание выходного датасета и датасета логов в системе;
6. Создание заданий.



От шаблона к заданиям

Сервис для опроса системы управления данными (DMS)

1. Итерация по заданиям в статусе “DEFINED”;
2. Опрос DMS о статусе входного датасета (“CLOSED”);
3. Создание выходных датасетов и датасетов логов в DMS;
4. Отправление задания в RabbitMQ для последующей обработки в WMS;
5. Смена статуса задания на “IN_PROGRESS”.

Задания до


```
[
 {
  "task_id": 11,
  "args": "cable_map",
  "device_type": "CPU",
  "retry": 5,
  "dataset_log_id": 28,
  "workflow_id": 6,
  "executable": "processing_program",
  "rank": 1,
  "mode": "map",
  "dataset_in_id": 26,
  "dataset_out_id": 27,
  "status": "DEFINED"
 },
 {
  "task_id": 12,
  "args": "cable_map",
  "device_type": "CPU",
  "retry": 5,
  "dataset_log_id": 30,
  "workflow_id": 6,
  "executable": "processing_program",
  "rank": 1,
  "mode": "map",
  "dataset_in_id": 27,
  "dataset_out_id": 29,
  "status": "DEFINED"
 }
]
```


RabbitMQ

Задания после


```
[
 {
  "task_id": 11,
  "args": "cable_map",
  "device_type": "CPU",
  "retry": 5,
  "dataset_log_id": 28,
  "workflow_id": 6,
  "executable": "processing_program",
  "rank": 1,
  "mode": "map",
  "dataset_in_id": 26,
  "dataset_out_id": 27,
  "status": "IN_PROGRESS"
 },
 {
  "task_id": 12,
  "args": "cable_map",
  "device_type": "CPU",
  "retry": 5,
  "dataset_log_id": 30,
  "workflow_id": 6,
  "executable": "processing_program",
  "rank": 1,
  "mode": "map",
  "dataset_in_id": 27,
  "dataset_out_id": 29,
  "status": "DEFINED"
 }
]
```


Отправление заданий на обработку

Логирование, контейнеризация и оркестрация

- Логирование: custom logger

```
task_generator 2024-09-24 20:59:44,380 INFO Logger started
task_generator 2024-09-24 20:59:44,665 INFO Templates successfully initialised.
```

- Контейнеризация и оркестрация: Docker и Docker Compose

docker-compose

```
services:
  db_api:
    ...
  task_generator:
    ...
  task_manager:
    ...
  template_manager
    ...

networks:
  wfms:
```

Результаты и дальнейшие планы

Результаты:

1) разработан API для работы с базой данных, открывающий доступ для сервисов WfMS к определенному набору функций;

2) создан сервис для работы с оператором обработки данных, позволяющий просматривать задания и шаблоны, а также предоставляющий возможность суперпользователям генерировать шаблоны и управлять их статусами;

3) разработан сервис для генерации заданий, сопоставляющий датасеты с шаблонами по маске имени, создающий цепочки обработки и создающий задания;

4) создан сервис для опроса DMS, опрашивающий его о готовности входного датасета для каждого готового задания, создающий выходные датасеты для таких заданий и отправляющий задания на обработку в WMS;

5) произведена контейнеризация всех приложений и настроена их оркестрация с помощью docker-compose;

6) внедрен logger, сохраняющий информацию об актуальном состоянии системы в каждый момент времени.

Дальнейшие планы:

1) Интеграция с SPD-IAM (identity and access management service);

2) Добавление поддержки загрузки шаблонов из файла;

3) Разработка сервиса для взаимодействия с WMS;

4) Переход к полной асинхронности;

5) Тестирование системы.

Спасибо за внимание!