# Задание №3. Завод по переработке мусора

**Введение**

Основной задачей системы является автоматизация процесса учета, сортировки и переработки отходов. Важной частью будет управление логами состояния оборудования, а также мониторинг и проверка корректности поступающих отходов. Для достижения этих целей будет использоваться система, включающая базы данных для учета данных, технические средства контроля и визуализацию через интерфейс для отображения текущих данных и уведомлений.

**1. Схема процесса приема и учета промышленных отходов**

Процесс приема отходов можно разделить на несколько этапов:

* **Прием отходов** — На этом этапе производится регистрация отходов, их первичная проверка на наличие признаков сортировки (например, упаковка в пакеты или контейнеры с указанием типа отходов). Все отходы маркируются и поступают на линию сортировки.
* **Сортировка отходов** — Система автоматически или вручную сортирует отходы на различные категории, такие как стекло, пластик, металл, бумага и прочее.
* **Проверка корректности сортировки** — После сортировки производится проверка корректности. Если отходы неправильно отсортированы, они отправляются на дополнительную сортировку.
* **Распределение в цеха переработки** — После проверки отходы распределяются по цехам переработки в зависимости от их типа.
* **Логирование состояния оборудования** — Ведется учет работы оборудования на каждом этапе.

**2. Роли пользователей и их функции в системе**

* **Оператор линии** — Регистрация отходов, первичная сортировка, запуск процесса переработки, контроль за исправностью оборудования.
* **Инженер по контролю качества** — Проверка правильности сортировки, отправка на переработку неправильных отходов.
* **Менеджер по производству** — Общее управление процессом, оценка эффективности работы системы, анализ логов.
* **Администратор системы** — Настройка системы, управление правами доступа, настройка уведомлений.
* **Технический специалист** — Обслуживание и ремонт оборудования.

**3. Описание архитектуры системы**

* **Технические средства**: для контроля за состоянием оборудования будут использоваться датчики на каждом этапе сортировки и переработки отходов (например, датчики веса, температуры, вибрации и камер для идентификации типа отходов).
* **Программные средства**: для обработки данных будет использоваться Python с библиотеками для обработки данных (Pandas, NumPy), а также база данных PostgreSQL для хранения информации о состояниях, логах и данных о сырье.
* **Интерфейс**: для визуализации данных и мониторинга состояния можно использовать веб-интерфейс, например, с использованием библиотеки Dash или Streamlit, с отображением графиков и уведомлений.

**4. Инфологическая и даталогическая схема базы данных**

Инфологическая схема будет включать следующие сущности:

**Waste** (Отходы): ID, тип, дата поступления, количество.

**Sorting** (Сортировка): ID, Waste ID, статус, дата.

**EquipmentStatus** (Состояние оборудования): ID, статус оборудования, время.

**Logs** (Логи): ID, время, событие, описание.

**Пример даталогической схемы для PostgreSQL:**

CREATE TABLE Waste (

id SERIAL PRIMARY KEY,

type VARCHAR(50),

quantity INT,

arrival\_date DATE

);

CREATE TABLE Sorting (

id SERIAL PRIMARY KEY,

waste\_id INT REFERENCES Waste(id),

status VARCHAR(50),

sort\_date DATE

);

CREATE TABLE EquipmentStatus (

id SERIAL PRIMARY KEY,

equipment\_name VARCHAR(50),

status VARCHAR(50),

last\_checked TIMESTAMP

);

CREATE TABLE Logs (

id SERIAL PRIMARY KEY,

event\_time TIMESTAMP,

event\_description TEXT

);

**5. Интерфейс для визуализации данных**

Интерфейс будет представлять собой дашборд с такими данными, как:

* Текущий статус оборудования (работает/не работает).
* Количество переработанных отходов по категориям.
* Логи системы и уведомления о необходимых действиях.

**Пример интерфейса с использованием Dash:**

import dash

from dash import dcc, html

import plotly.graph\_objs as go

import pandas as pd

app = dash.Dash(\_\_name\_\_)

# Пример данных

data = pd.DataFrame({

'type': ['Plastic', 'Glass', 'Metal', 'Paper'],

'quantity': [200, 150, 300, 100]

})

app.layout = html.Div([

html.H1("Система переработки отходов"),

dcc.Graph(

id='waste-bar-chart',

figure={

'data': [

go.Bar(

x=data['type'],

y=data['quantity'],

name='Количество отходов'

)

],

'layout': go.Layout(

title='Количество отходов по категориям',

xaxis={'title': 'Тип отходов'},

yaxis={'title': 'Количество'}

)

}

)

])

if \_\_name\_\_ == '\_\_main\_\_':

app.run\_server(debug=True)

**Заключение**

Предложенное решение представляет собой систему автоматизации процессов сортировки и переработки отходов. Используемая архитектура позволяет эффективно управлять данными, отслеживать состояние оборудования и выполнять необходимые действия на каждом этапе. Реализованный интерфейс позволяет отслеживать данные в реальном времени и принимать своевременные решения.