|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Главная | | | | | |
| ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ  Государственное бюджетное общеобразовательное учреждение  города Москвы **«Школа № 1533 «ЛИТ»** | | | | | |
| **ВЫПУСКНОЙ ПРОЕКТ** | | | | | |
| **Профиль**: Программирование  **Тема**: Приложение для обработки первичных данных системы автоматизации | | | | | |
| **Исполнитель**: | Левин Максим Денисович, группа 10.3 | | | | |
| **Заказчик**: | Мугандин Юрий Валерьевич, ОАО «КБК «Черёмушки», директор | | | | |
| **Консультант**: | Гиглавый Александр Владимирович, ГБОУ «Школа № 1533 «ЛИТ», заместитель директора по науке | | | | |
| **Руководитель**: | Кисляков Антон Юрьевич, РТУ МИРЭА, преподаватель дополнительного образования в ГБОУ «Школа №1533 «ЛИТ» | | | | |
|  |  | | | | |
|  | |  |  |  |  |
|  |  | | |
|  | | | | | |
| Москва, 2023 г | | | | | |

СОДЕРЖАНИЕ

[Введение и актуальность 3](#__RefHeading___Toc848_28543932)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 4](#__RefHeading___Toc852_28543932)

[2. ВЫБОР И ОБОСНОВАНИЕ ТЕХНОЛОГИЙ РАЗРАБОТКИ 6](#__RefHeading___Toc854_28543932)

[3. РАЗРАБОТКА / РЕШЕНИЕ ЗАДАЧИ 7](#__RefHeading___Toc856_28543932)

[заключение 8](#__RefHeading___Toc858_28543932)

[список использованных источников 9](#__RefHeading___Toc860_28543932)

[приложение 10](#__RefHeading___Toc862_28543932)

# Перечень терминов и сокращений

IDE – (англ. Integrated Development Environment) среда разработки приложений на определённом языке программирвания.

БД – база данных.

СУБД – система упраления базой данных.

Библиотека – сторонний набор иснтрументов для выполнения определённой задачи, подключаемый к проекту.

GUI – (англ. Graphical User Interface) графический пользовательский интерфейс.

Платформа – операционная система и/или некоторое программное обеспечение, на основе котороых работает описываемое программное обеспечение.

Python - проект – совокупность скриптов, библиотек и расширений на языке программировани Python, вместе образующих определённое пользовательское приложение.

# 1.Введение и актуальность

В настоящее время всё чаще различные процесы подвергаются частичной или полной автоматизации с помощью различного программного обеспечения, в результате чего частично проблемы решаются, но вместо них появляются новые, требующие новых способов решения.

С одной стороны автоматизация заменяет труд человека делая процесс производства дешевле и проще, с другой стороны, возникают трудности в общении человека с машиной там, где автоматизация лишь частичная. Зачастую, проблема заключается в том, что формат данных, выдаваемых компъютером непригоден для человека.

В таких случаях помогает приложение принимающее на вход данные, понятные для машины, обрабатывающее их, и выдающее на выходе понятные для человека.

# 2. ЗАДАЧА

**2.1. Проблема**

На предприятии «КБК «Черёмушки» для изготовления хлеба используются автоматизированные печи с системой, которая считает на выходе из печи готовые батоны, а также распознаёт бракованные. Первичные данные этой системы представляют собой поток записей, непригодных к чтению и использованию живым человеком ввиду их огромного количества и бесполезности одной или нескольких отдельных записей, а человеку нужна статистика, основанная на всём массиве данных или существенной его части.

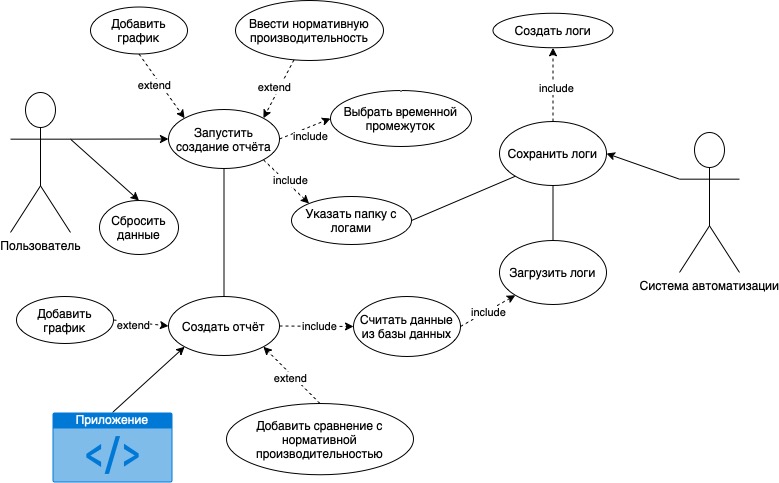
**2.2. Постановка задачи**

Требуется разработать оконное приложение, для выбора параметров и дополнительных функций, по которым, на основе загруженных данных будет создаваться отчёт в формате Excel-таблицы.

Основные параметры это период времени, за который нужен отчёт, единица времени в отчёте (так как в исходных данных это минута, и отчёт, например, за год, расписанный по минутам бесполезен) и номера производственных линий, для которых будет делаться выборка. Также должна быть возможность добавить к таблице график на основе этих данных и ввод нормативного производства для сравнения его с действительным.

Одна из основных проблем заключается в неспособности большинства приложений для статистического анализа автоматически обнаруживать и загружать данные.

На рисунке 1 изображена диаграмма возможных действий приложения и пользователя.

Рис. 1 — Диаграмма Use-case

# 3. АКТУАЛЬНОСТЬ И ЦЕЛЕВАЯ АУДИТОРИЯ

Приложение будет использоваться сотрудниками «КБК «Черёмушки» для просмотра статистики производства продукции и облегчения контроля  
за процессом производства.

# 4. ОБЗОР АНАЛОГОВ

**4.1 Develve**

Давно не обновлялось, имеет скудный функционал, но, с другой стороны, одно из немногих бесплатных.

**4.2 SPSS IBM**

Самое первое, и до сих пор поддерживаемое приложение с обширным функционалом, но не решающее одну из основных проблем проекта (автозагрузка данных).

**4.3 STATA**

Способно автоматически загружать данные, но не сортирует их, и не предоставляет также нужные функции.

**4.4 Minitab**

Не решает основную проблему, также не имеет возможности выбрать определённую часть данных

**4.5 Wizard Mac**

Также решает основную проблему, однако доступно только на MacOS, в то время, как заказчику нужно приложение для Windows.

**4.6 Таблица сравнения аналогов**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Приложение | Автозагрузка данных | Выбор параметров | Кросс- платформенное | Бесплатное |
| Develve | Нет | Нет | Нет | Да |
| SPSS IBM | Нет | Да | Да | Частично\* |
| STATA | Да | Нет | Да | Нет |
| Minitab | Нет | Нет | Да | Нет |
| Wizard Mac | Да | Да | Нет | Частично\* |
| Моё | Да | Да | Да | Да |

Рис. 3 — Таблица сравнения аналогов

# 5. Выбор технологий разработки

**5.1 Python 3.9**

Был выбран как несложный в работе язык программирования, для которого существует множество библиотек, в том числе, все, которые нужны для решения задачи. Данная версия языка была выбрана, т. к. версии 2.X поддерживаются не каждой библиотекой по причине того, что устарели, а версии от 3.10 и новее наоборот, могут не везде поддерживаться так как вышли недавно.

**5.2 PyCharm**

Самая популярная и вполне удобная IDE для языка Python, единственный минус которой в возможности возникновения сложностей при подключении библиотек, что, впрочем, и для других IDE не редкость.

**5.3 OpenPyXL**

Достаточно функциональная библиотека для работы с xlsx файлами и большим набором инструментов для создания графиков, но, местами, сложным синтаксом, была выбрана, так как является самой доступной и популярной.

**5.4 PyQt6**

Реализация для языка Python GUI - библиотеки Qt, изначально написанной на С++. Имеет широкий выбор инструментов и относительно простой синтаксис, а также совместима со многими платформами и, в отличие от ранее использованных аналогов, стабильно работает на всех поддерживаемых платформах.

**5.5 СУБД SQLite3**

В проекте не используется сложная структура таблиц, но нужна быстрая обработка большого массива данных, и SQLite идеально подходит для таких задач.

**5.6 DBeaver**

Универсальный менеджер БД, использовался в процессе проектирования БД и в ходе тестирования приложения, для проверки корректности работы СУБД.

**5.7 PyInstaller**

Самый популярный пакет инструментов для сборки компиляции  
Python – проектов.

**6. РЕШЕНИЕ**

**6.1 Ход работы**

|  |  |
| --- | --- |
| Январь 2023 года | Получение темы проекта, встреча с заказчиком, моделирование приложения, подбор библиотек. |
| Февраль 2023 года | Работа над поиском, загрузкой и обработкой файлов первичных данных, изучение некоторых особенностей кроссплатформенности языка Python. |
| Март 2023 года | Обнаружение низкой эффективности ситуационной загрузки файлов, поиск иного решения, работа над БД. |
| Апрель 2023 года | Дизайн интерфейса, работа над ним, переход с PySimpleGUI на Tkinter из-за технических проблем, переработка интерфейса ввиду его неудобности, переход с Tkinter на PyQt |
| Май 2023 года | Работа над функциональной частью, соединение интерфейса и функций в работающее приложение. |
| Июнь 2023 года | Завершение работы, тестирование приложения и доработка. |

Рис. 4 – Ход работы

**6.2 Проблемы**

**6.2.1 Графическая библиотека**

Изначально для разработки GUI была выбрана библиотека PySimpleGUI, но из-за проблем с совместимостью она была заменена на Tkinter, и, вскоре, по тем же причинам на PyQt..

**6.2.2 Формат даты и времени**

В исходных данных используются два разных формата записи даты и времени, ни один из которых не совпадает со стандартом ISO 8601 (ГГГГ-ММ-ДД ЧЧ:ММ:СС), который используется во всех билиотеках, из за этого приходилось постоянно приводить формат к этому стандарту, что затрудняло разработку.

**6.2.3 Ситуационная загрузка**

Изначально не было планов использовать БД, однако тогда при каждом создании отчёта пришлось бы загружать нужные файлы и брать информацию из них, а при наличии БД каждый файл обрабатывается единожды, после чего информация из него попадает в СУБД и затем вне зависимости от настроек взаимодействие идёт только с БД. При каждом запуске приложение проверяет наличие ранее не загруженных файлов, и добавляет в БД инфомацию из них. Так как в имени файла указывается дата его создания, можно запоминать лишь имя последнего загруженного файла, и начинать со следующего за последним записаным в предыдущий сеанс.

**6.2.4 Сборка проекта**

Уже в июне, когда менять язык было поздно, выяснилось, что стандартных средств сборки Python – проектов не существует, поэтому пришлось искать сторонние средства сборки.

# 7. Результаты

**7.1 Итог**

Удалось достичь всех поставленных задач.

**7.2 Выводы**

Были улучшены навыки программирования на Python и изучены новые технологии разработки.

**8. НАПРАВЛЕНИЕ ДАЛЬНЕЙШЕЙ РАЗРАБОТКИ**

Персепктивной выглядит идея разделить приложение на модуль загрузки данных в БД и модуль генерации отчёта и связать их через локальную сеть, чтобы имея данные на одном компъютере можно было получить отчёт на любом, находящемся в сети.

**9. ПРИМЕР ИСПОЛЬЗОВАНИЯ**

На вход приложение принимает заданные пользователем параметры итогового отчёта и файлы в формате .csv, на выходе выдаёт отчёт в формате .xlsx таблицы.

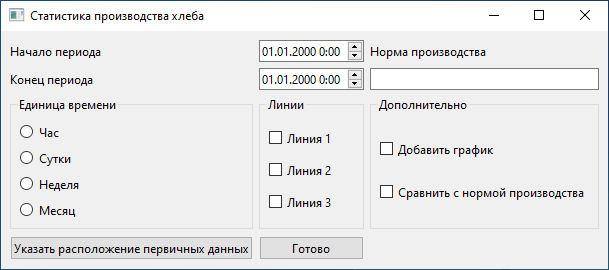


Рис. 5 – Основное окно приложения

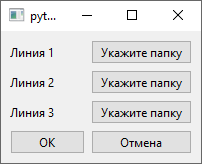


Рис. 6 – Окно выбора расположения исходных данных

При первом запуске приложения надо указать папки в которых находятся первичные данные (файлы в формате .csv), в последующие запуски это делать не обязательно, однако можно их изменить.

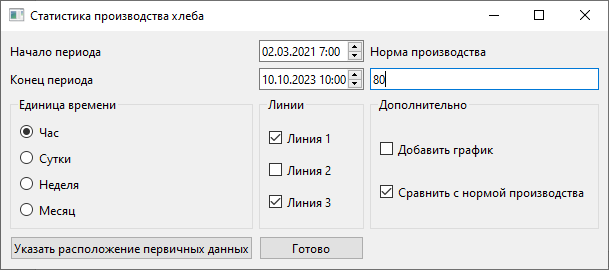


Рис. 7 – Основное окно приложения c заданными параметрами

Также надо выбрать период времени, для которого будет создан отчёт, единицу времени, по которой записи будут группироваться, линию или линии, данные которой(-ых) будут в отчёте и по желанию включить добавление графика и/или сравнения с нормативными данными.

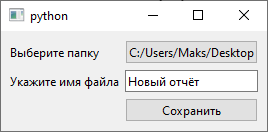


Рис. 8 – Окно сохранения файла

После указания всех необходимых данных надо нажать кнопку “Готово”, и в появившемся окне указать папку, куда будет сохранен файл и его имя, после чего нажать кнопку “Сохранить”.

# 10. список использованных источников

**Статьи о PyQt6**

PyQt6 — полное руководство для новичков // Хабр URL: <https://habr.com/ru/companies/skillfactory/articles/599599/>

Creating additional windows // PythonGUIS URL: <https://www.pythonguis.com/tutorials/pyqt6-creating-multiple-windows/>

Set QLineEdit to accept only numbers // StackOverFlow URL: <https://stackoverflow.com/questions/13422995/set-qlineedit-to-accept-only-numbers>

**Документация библиотек**

PyQt Documentation v6.5.1 // RiverbankComputing URL: <https://www.riverbankcomputing.com/static/Docs/PyQt6/index.html>

CSV File Reading and Writing // Python documentation URL: <https://docs.python.org/3/library/csv.html>

sqlite3 — DB-API 2.0 interface for SQLite databases // Python documentation URL: <https://docs.python.org/3/library/sqlite3.html>

json — JSON encoder and decoder // Python documentation URL: <https://docs.python.org/3/library/json.html>

openpyxl - A Python library to read/write Excel 2010 xlsx/xlsm files // OpenPyXL URL: <https://openpyxl.readthedocs.io/en/stable/index.html>

**Вспомогательные статьи**

Поймай меня, если сможешь: руководство по обработке исключений в Python // Хабр URL: <https://habr.com/ru/companies/wunderfund/articles/736526/>

How can I parse (read) and use JSON in Python? // StackOverFlow URL: <https://stackoverflow.com/questions/7771011/how-can-i-parse-read-and-use-json-in-python>

Date And Time Functions // SQLite URL: <https://www.sqlite.org/lang_datefunc.html>

**Сборка проекта**

PyInstaller Manual // PyInstaller URL: <https://pyinstaller.org/en/stable/index.html>

Как создать exe файл для Python кода с помощью PyInstaller // PythonRU URL: <https://pythonru.com/biblioteki/pyinstaller?ysclid=liwzd3etom168946065>

# 11. Приложение

Исходный код находится по адресу <https://github.com/Leksin0/Bread>

Листинг 1 – Исходный код файла fileops.py (частичный)

def LoadData(line):

dirpath = LoadSettings(f"{line}path")

nextime = LoadSettings(f"{line}time")

while True:

try:

pattern = "/Detailed min</>.csv.csv"

name = pattern.replace('</>', nextime)

path = dirpath + name

LoadLogFile(line, path)

except FileNotFoundError:

break

else:

log = nextime.split('-')

nextime = datetime(int(log[0]), int(log[1]), int(log[2]), int(log[3]))

if nextime.hour == 7:

nextime = nextime.replace(hour=19)

else:

try:

nextime = nextime.replace(day=nextime.day + 1, hour=7)

except:

try:

nextime = nextime.replace(month=nextime.month + 1, day=1, hour=7)

except:

nextime = nextime.replace(year=nextime.year + 1, month=1, day=1, hour=7)

temp = []

temp.append(str(nextime.year))

temp.append(str(nextime.month))

if len(temp[1]) == 1:

temp[1] = "0" + temp[1]

temp.append(str(nextime.day))

if len(temp[2]) == 1:

temp[2] = "0" + temp[2]

temp.append(str(nextime.hour))

if len(temp[3]) == 1:

temp[3] = "0" + temp[3]

temp.append("00")

temp.append("00")

nextime = "-".join(temp)

log = nextime.split('-')

nextime = datetime(int(log[0]), int(log[1]), int(log[2]), int(log[3]))

if nextime.hour == 7:

nextime = nextime.replace(hour=19)

else:

nextime = nextime.replace(hour=7)

try:

nextime = nextime.replace(day=nextime.day + 1)

except:

try:

nextime = nextime.replace(month=nextime.month + 1, day=1)

except:

nextime = nextime.replace(year=nextime.year + 1, month=1, day=1)

temp = []

temp.append(str(nextime.year))

temp.append(str(nextime.month))

if len(temp[1]) == 1:

temp[1] = "0" + temp[1]

temp.append(str(nextime.day))

if len(temp[2]) == 1:

temp[2] = "0" + temp[2]

temp.append(str(nextime.hour))

if len(temp[3]) == 1:

temp[3] = "0" + temp[3]

temp.append("00")

temp.append("00")

nextime = "-".join(temp)

SaveSettings(f"{line}time", nextime)

def LoadLogFile(lnum, filepath):

if lnum == 1:

line = 'LineOne'

elif lnum == 2:

line = 'LineTwo'

elif lnum == 3:

line = 'LineThree'

doc = open(filepath, encoding='Windows-1251')

reader = csv.reader(doc)

reader.\_\_next\_\_()

reader.\_\_next\_\_()

for row in reader:

row = row[0].split(';')

log = row[0]

dt = log.split(' ')[0].split('.')

tm = log.split(' ')[1].split(':')

dtm = datetime(int(dt[2]), int(dt[1]), int(dt[0]), int(tm[0]), int(tm[1]))

DBInsert(f"INSERT INTO {line} (time, loafs, defective) VALUES ('{str(dtm)}', {row[2]}, {row[1]})")

def SearchData(timestart, timeend, line, gu):

if line == '1':

lname = 'LineOne'

elif line == '2':

lname = 'LineTwo'

elif line == '3':

lname = 'LineThree'

query = f"""SELECT strftime('%{gu}', time) as grtime,

time,

SUM(loafs) as sumloafs,

SUM(defective) as sumdefective

FROM {lname}

WHERE time BETWEEN'{str(timestart)}' AND '{str(timeend)}' GROUP BY grtime"""

return DBCursor().execute(query)

**Листинг 2 – Исходный код файла main.py (частичный)**

class MainWindow(QMainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.dirwindow = None

self.cbl1 = QCheckBox("Линия 1")

self.cbl1.clicked.connect(self.rborcb\_click)

self.cbl2 = QCheckBox("Линия 2")

self.cbl2.clicked.connect(self.rborcb\_click)

self.cbl3 = QCheckBox("Линия 3")

self.cbl3.clicked.connect(self.rborcb\_click)

linhelp = QVBoxLayout()

linhelp.addWidget(self.cbl1)

linhelp.addWidget(self.cbl2)

linhelp.addWidget(self.cbl3)

lineselect = QGroupBox("Линии")

lineselect.setLayout(linhelp)

self.rbu1 = QRadioButton("Час")

self.rbu1.clicked.connect(self.rborcb\_click)

self.rbu2 = QRadioButton("Сутки")

self.rbu2.clicked.connect(self.rborcb\_click)

self.rbu3 = QRadioButton("Неделя")

self.rbu3.clicked.connect(self.rborcb\_click)

self.rbu4 = QRadioButton("Месяц")

self.rbu4.clicked.connect(self.rborcb\_click)

unhelp = QVBoxLayout()

unhelp.addWidget(self.rbu1)

unhelp.addWidget(self.rbu2)

unhelp.addWidget(self.rbu3)

unhelp.addWidget(self.rbu4)

unitselect = QGroupBox("Единица времени")

unitselect.setLayout(unhelp)

self.cbchart = QCheckBox("Добавить график")

self.cberp = QCheckBox("Сравнить с нормой производства")

dophelp = QVBoxLayout()

dophelp.addWidget(self.cbchart)

dophelp.addWidget(self.cberp)

addition = QGroupBox("Дополнительно")

addition.setLayout(dophelp)

lbts = QLabel("Начало периода")

lbte = QLabel("Конец периода")

self.timestart = QDateTimeEdit()

self.timeend = QDateTimeEdit()

lberp = QLabel("Норма производства")

self.txerp = QLineEdit()

self.txerp.setValidator(QIntValidator())

btchdir = QPushButton("Указать расположение первичных данных")

btchdir.clicked.connect(self.chdir\_click)

self.btresult = QPushButton("Готово")

self.btresult.clicked.connect(self.GetResult)

layout = QGridLayout()

layout.addWidget(lbts, 0, 0)

layout.addWidget(self.timestart, 0, 1)

layout.addWidget(lbte, 1, 0)

layout.addWidget(self.timeend, 1, 1)

layout.addWidget(lberp, 0, 2)

layout.addWidget(self.txerp, 1, 2)

layout.addWidget(unitselect, 2, 0, 1, 1)

layout.addWidget(lineselect, 2, 1, 1, 1)

layout.addWidget(addition, 2, 2, 1, 1)

layout.addWidget(btchdir, 3, 0)

layout.addWidget(self.btresult, 3, 1)

self.setWindowTitle("Статистика производства хлеба")

self.container = QWidget()

self.container.setLayout(layout)

self.setCentralWidget(self.container)

def GetResult(self):

if LoadSettings("1path") is None or LoadSettings("2path") is None or LoadSettings("3path") is None:

self.btresult.setText("Укажите папки первичных данных")

return

else:

pass

if not self.rbu1.isChecked() and not self.rbu2.isChecked() and not self.rbu3.isChecked() and not self.rbu4.isChecked():

self.btresult.setText("Выберите еденицу времени")

return

else:

if self.rbu1.isChecked():

unit = 'H'

elif self.rbu2.isChecked():

unit = 'd'

elif self.rbu3.isChecked():

unit = 'W'

elif self.rbu4.isChecked():

unit = 'm'

if not self.cbl1.isChecked() and not self.cbl2.isChecked() and not self.cbl3.isChecked():

self.btresult.setText("Выберите хотя бы одну линию")

return

else:

lines = []

if self.cbl1.isChecked():

lines.append('1')

if self.cbl2.isChecked():

lines.append('2')

if self.cbl3.isChecked():

lines.append('3')

try:

a = int(self.txerp.text())

except:

if self.cberp.isChecked():

self.btresult.setText("Укажите норму производства")

return

if self.cberp.isChecked():

erp = [True, a]

else:

erp = [False]

if self.cbchart.isChecked():

chart = True

else:

chart = False

dtms = self.timestart.text()

log = []

for j in dtms.split(' ')[0].split('.'):

log.append(j)

for h in dtms.split(' ')[1].split(':'):

log.append(h)

dtms = datetime(int(log[2]), int(log[1]), int(log[0]), int(log[3]), int(log[4]))

dtme = self.timeend.text()

log = []

for i in dtme.split(' ')[0].split('.'):

log.append(i)

for k in dtme.split(' ')[1].split(':'):

log.append(k)

dtme = datetime(int(log[2]), int(log[1]), int(log[0]), int(log[3]), int(log[4]))

Result(dtms, dtme, lines, unit, chart, erp)