ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ ЖОҒАРЫ БІЛІМ МИНИСТРЛІГІ

«Л.Н. ГУМИЛЕВ АТЫНДАҒЫ ЕУРАЗИЯ ҰЛТТЫҚ УНИВЕРСИТЕТІ» КЕАҚ

Студенттер мен жас ғалымдардың
«ǴYLYM JÁNE BILIM - 2023»
XVIII Халықаралық ғылыми конференциясының
БАЯНДАМАЛАР ЖИНАҒЫ

СБОРНИК МАТЕРИАЛОВ
XVIII Международной научной конференции
студентов и молодых ученых
«ǴYLYM JÁNE BILIM - 2023»

PROCEEDINGS
of the XVIII International Scientific Conference
for students and young scholars
«ǴYLYM JÁNE BILIM - 2023»

2023
Астана

УДК 001+37
ББК 72+74
G99

«ǴYLYM JÁNE BILIM – 2023» студенттер мен жас ғалымдардың XVIII Халықаралық ғылыми конференциясы = XVIII Международная научная конференция студентов и молодых ученых «ǴYLYM JÁNE BILIM – 2023» = The XVIII International Scientific Conference for students and young scholars «ǴYLYM JÁNE BILIM – 2023». – Астана: – 6865 б. - қазақша, орысша, ағылшынша.

ISBN 978-601-337-871-8

Жинаққа студенттердің, магистранттардың, докоранттардың және жас ғалымдардың жаратылыстану-техникалық және гуманитарлық ғылымдардың өзекті мәселелері бойынша баяндамалары енгізілген.

The proceedings are the papers of students, undergraduates, doctoral students and young researchers on topical issues of natural and technical sciences and humanities.

В сборник вошли доклады студентов, магистрантов, докторантов и молодых ученых по актуальным вопросам естественно-технических и гуманитарных наук.

УДК 001+37
ББК 72+74

ISBN 978-601-337-871-8

конфиденциальности, проблемы интеграции, обучения персонала и затраты. Организации должны тщательно оценить преимущества и проблемы каждой технологии и разработать комплексный план внедрения для успешного внедрения технологии автоматизации в управленческий учет.

В заключение следует отметить, что внедрение технологий автоматизации в управленческий учет имеет важное значение для того, чтобы организации оставались конкурентоспособными и принимали обоснованные решения, основанные на точной и своевременной финансовой информации. Использование ERP-систем, инструментов BI и систем ИИ может значительно повысить эффективность финансовой отчетности, высвободив ресурсы персонала, чтобы сосредоточиться на более стратегических мероприятиях, повышающих ценность. Решая проблемы, связанные с внедрением технологии автоматизации, организации могут в полной мере воспользоваться преимуществами этой технологии и достичь своих финансовых целей.

Список использованной литературы

1. N. G. Sapozhnikova, E. S. Igonina, S. Y. Shamrina, S. A. Tunin, V. S. Germanova Conceptual foundations of management accounting in budgetary institutions // Digital Technologies and Institutions for Sustainable Development. 2022. P. 375-380.
2. S. S. Halbouni, M. A. Nour An empirical study of the drivers of management accounting innovation: A UAE perspective // International Journal of Managerial and Financial Accounting. 2014, 6(1). P. 60-86.
3. Y. Jin WEB platform based ERP financial accounting management system // Energy Education Science and Technology Part A: Energy Science and Research. 2014, 32(5). P. 4531-4538.
4. E. Duçi The relationship between management accounting, strategic management accounting and strategic cost management // Academic Journal of Interdisciplinary Studies, 2021, 10(5). P. 376-389.
5. S. Malinić, M. Todorović How does management accounting change under the influence of ERP? // Ekonomska Istrazivanja. 2012, 25(3). P. 722-751.

УДК 004.432

РЕАЛИЗАЦИЯ БИЗНЕС-ЛОГИКИ ERP-СИСТЕМЫ С ИСПОЛЬЗОВАНИЕМ PYTHON

Уасбекова Жанерке Канаткызы
Магистрант 2 курса факультета информационных технологий ЕНУ им. Л.Н.Гумилева, Нур-Султан, Казахстан
Научный руководитель – Ламашева Ж. Б.

Аннотация. Системы планирования общеорганизационных ресурсов (ERP) необходимы для управления бизнес-процессами в различных функциональных областях организации. Одной из ключевых областей, где ERP-системы могут принести значительную пользу, является планирование производства. В этой статье мы продемонстрируем, как бизнес-логика планирования производства может быть реализована в ERP-системе с использованием языка программирования Python. Мы представляем реализацию бизнес-логики планирования производства на Python, который генерирует выполнимые производственные планы на основе доступных запасов. Мы также приводим пример сценария, в котором наша реализация может быть применена к производственной компании для формирования производственного плана на основе заказов клиентов и текущего уровня запасов. Наша реализация демонстрирует гибкость и простоту использования Python для реализации бизнес-логики в ERP-системе.

Ключевые слова: ERP-система, класс, организация, атрибут.

Введение. Планирование производства — это важнейший аспект производства, который включает в себя планирование производственной деятельности для удовлетворения потребительского спроса при минимизации затрат и максимизации эффективности. Для эффективного управления производственным планированием организациям необходима

комплексная ERP-система, способная интегрировать планирование производства с другими функциональными областями, такими как управление запасами, финансы и человеческие ресурсы. ERP-система может помочь организациям оптимизировать процесс планирования производства, оптимизировать уровень запасов и повысить общую эффективность [1, 2, 3].

Методология. Мы демонстрируем нашу реализацию бизнес-логики планирования производства с использованием кода Python. Мы определяем три класса: “ ProductionPlan”, “ProductionPlanner” и “Order”. Класс “ ProductionPlan” представляет производственный план для конкретного продукта с заданным количеством и сроком выполнения. В нем есть методы проверки того, выполним ли план на основе текущих запасов, и резервирования необходимых запасов для плана. Класс “ProductionPlanner” — это основной класс, который обрабатывает планирование производства. В качестве входных данных он принимает текущие запасы и список заказов, которые необходимо выполнить. Затем он проверяет, выполним ли каждый заказ на основе текущих запасов, и создает производственный план для выполнимых заказов. Класс “Order” представляет заказ клиента на определенный продукт с заданным количеством и сроком выполнения [4].

Реализация. Выполнение работы включает в себе несколько этапов:

1. **Управление заказами**:

Система должна иметь возможность обрабатывать входящие заказы, обрабатывать их и отслеживать ход выполнения от размещения заказа до его завершения.

Для начала нужно создать класс “Order”, который содержит информацию о каждом заказе, включая идентификатор заказа, имя клиента, название продукта, количество и дату оплаты. Атрибуту “ fulfilled “ изначально присвоено значение “False”, указывающее на то, что заказ еще не выполнен. Метод “fulfill_order()” устанавливает атрибуту “fulfilled” значение “True”, когда заказ выполнен (рис. 1).

```
class Order:
    def __init__(self, order_id, customer_name, product_name, quantity, due_date):
        self.order_id = order_id
        self.customer_name = customer_name
        self.product_name = product_name
        self.quantity = quantity
        self.due_date = due_date
        self.fulfilled = False

    def fulfill_order(self):
        self.fulfilled = True
```

Рисунок 1 – создание класса “Order”

Код, показанный на рисунке 2, даст возможность создать класс “OrderManager”, который отвечает за управление заказами. Метод “add_order()” добавляет новый заказ в систему. Метод “get_orders()” возвращает список всех заказов. Метод “get_unfulfilled_orders()” возвращает список заказов, которые еще не были выполнены.

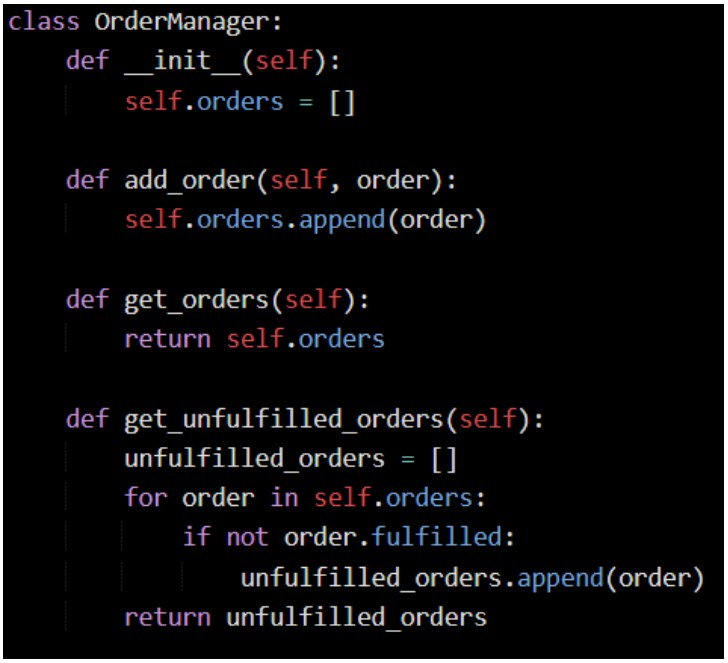

Рисунок 2 – создание класса “ OrderManager”

2. **Управление запасами**:

Система должна иметь возможность отслеживать уровни запасов каждого продукта и корректировать их по мере выполнения заказов.

Создание класса “Inventory” будет содержать информацию о каждом продукте в инвентаре, включая название продукта и количество. Метод “update_quantity()” корректирует количество товара при выполнении заказов (рис. 3).

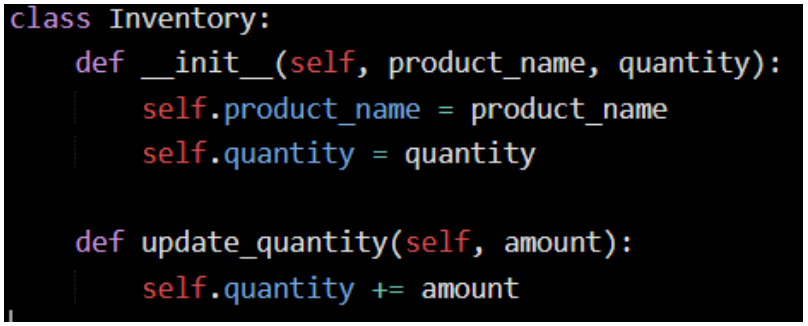

Рисунок 3 – создание класса “ Inventory”

Класс “InventoryManager” отвечает за управление запасами. Метод “add_product()” добавляет новый товар в инвентарь. Метод “get_inventory()” возвращает список всех товаров на складе. Метод “update_inventory()” обновляет уровни запасов при выполнении заказов (рис.4).

```python
class InventoryManager:
    def __init__(self):
        self.inventory = []

    def add_product(self, product):
        self.inventory.append(product)

    def get_inventory(self):
        return self.inventory

    def update_inventory(self, order):
        for product in self.inventory:
            if product.product_name == order.product_name:
                product.update_quantity(-order.quantity)
```

Рисунок 4 – создание класса “ InventoryManager”

3. **Планирование производства**:

Система должна быть способна генерировать производственный план на основе входящих заказов и текущего уровня запасов. Класс “ProductionPlan”, будет содержать информацию о производственном плане, включая название продукта, количество и срок выполнения (рис.5).

```python
class ProductionPlan:
    def __init__(self, product_name, quantity, due_date):
        self.product_name = product_name
        self.quantity = quantity
```

Рисунок 5 – создание класса “ ProductionPlan”

Наконец, класс “ProductionPlanner” будет отвечать за генерацию производственного плана. Метод “generate_plans()” использует информацию о заказе и запасах для создания производственного плана. Он проверяет, достаточно ли запасов для выполнения заказа, и соответствующим образом формирует план. Если запасов недостаточно, планировщик формирует план на основе доступных запасов и соответствующим образом корректирует порядок и уровни запасов. Метод возвращает список производственных планов, которые можно использовать для планирования производства [5].

```python
class ProductionPlanner:
    def __init__(self, order_manager, inventory_manager):
        self.order_manager = order_manager
        self.inventory_manager = inventory_manager
        self.production_plans = []

    def generate_plans(self):
        orders = self.order_manager.get_unfulfilled_orders()
        inventory = self.inventory_manager.get_inventory()
        for order in orders:
            for product in inventory:
                if product.product_name == order.product_name:
                    if order.quantity <= product.quantity:
                        self.production_plans.append(ProductionPlan(order.product_name, order.quantity, order.due_date))
                    else:
                        self.production_plans.append(ProductionPlan(order.product_name, product.quantity, order.due_date))
                        order.quantity -= product.quantity
                        product.quantity = 0
        return self.production_plans
```

Рисунок 6 – создание класса “ ProductionPlanner”

Вывод. В этой статье мы продемонстрировали, как бизнес-логика планирования производства может быть реализована в ERP-системе с использованием языка программирования Python. Мы представили реализацию бизнес-логики планирования производства на Python, включающую классы для управления производственными планами, заказами и запасами, а также “Productionplanner”, который генерирует выполнимые производственные планы на основе доступных запасов. Мы также привели пример сценария, в котором наша реализация может быть применена к производственной компании для формирования производственного плана на основе заказов клиентов и текущего уровня запасов. Наша реализация демонстрирует гибкость и простоту использования Python для реализации бизнес-логики в ERP-системе.

Список использованной литературы

1. J. Ko, M. Comuzzi Fuzzy analytic network process for evaluating ERP post-implementation alternatives // IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). 2017. P. 1-6.
2. P. Kumawat, G. Kalani, N.K. Kumawat Prediction of ERP outcome measurement and user satisfaction using adaptive neuro-fuzzy inference system and SVM classifiers approach // Proceedings of the International Congress on Information and Communication Technology. 2016. P. 229-237.
3. X. Zuo, S. Zhang An ERP system based on E-commerce model // International Symposium on Knowledge Acquisition and Modeling Workshop Proceedings, KAM. 2008. P. 333-335.
4. M. A. Olivero, L. Morales-Trujillo, F.J. Domínguez-Mayo, M. Mejías Systematic development of ERP modules using a model-driven strategy focusing on the users // Web Information Systems and Technologies. 2019. P. 489-492.
5. Y. Zhi, C. Qian, C. Li, J. Li Development of ERP financial statement evaluation system // Journal of Physics: Conference Series. 2019, 1302(2). P 1-7.

УДК 004

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ИНТЕРПРЕТАЦИИ ГЕОЭЛЕКТРИЧЕСКОГО РАЗРЕЗА

Шахатова Алия Талгатовна
shakhatovaa@list.ru
Докторант кафедры Компьютерной и программной инженерии
Евразийского национального университета имени Л.Н. Гумилева
Научный руководитель – Т. Мирғалиқызы

1. Математическое моделирование

Постановка прямой задачи: Рассматривается двумерная задача в области $\Omega = \{(z, y): z \in [0\ м,\ 2\ м],\ y \in [0\ м,\ 3\ м]\}$ [1-3]:

$$\mu\varepsilon\vartheta_{tt} + \mu\sigma\vartheta_t = \Delta\vartheta\,, \qquad (z, y) \in \Omega \tag{1}$$

$$\upsilon|_{t=0} = 0,\ \upsilon_t|_{t=0} = 0; \tag{2}$$

$$\vartheta_z|_{z=0} = \theta(410 - y)\theta(y - 390)\xi(t) \tag{3}$$

$$\upsilon_y|_{y=0} = 0,\ \upsilon|_{y=3} = 0; \tag{4}$$

$$[\upsilon]_{y=0.1} = 0,\ \lfloor \upsilon_y \rfloor_{y=0.1} = 0. \tag{5}$$

Где: $\xi(t)$ – импульс и имеет вид:

$$\xi(t) = \sin(3(t-1))\exp(-(0{,}7)^2(t-2)^2),$$