**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Компьютерные науки и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**Лабораторная работа № 1**

**по курсу «Программная инженерия»**

Студенты: Чернобаев А. А.

Группа: М8О-114М-23

Преподаватели: Дзюба Д.В.

Миронов Е.С.

Москва

2024

1. **Постановка задачи.**

1. Установить инструменты из списка

* Клиент Git
* Текстовый редактор (рекомендуется Visual Studio Code)
* Плагины к Visual Studio Code C4 DSL

2. Зарегистрироваться на github.com (если еще нет учетной записи)

3. Создать публичный репозиторий для выполнения практической работы у себя в аккаунте

4. Скопировать репозиторий https://github.com/DVDemon/hl\_mai\_lab\_00 с примерами задания

5. Создать файлы с описанием «архитектуры» согласно вашему варианту задания в Structurizr Lite.

6. Требования к диаграммам:

* Должна быть контекстная диаграмма
* Должна быть диаграмма контейнеров
* Должна быть диаграмма развертывания
* Должно быть несколько динамических диаграмм

**Вариант задания - 2 (Магазин)**

Приложение должно содержать следующие данные:

* Пользователь
* Товар
* Корзина

Реализовать API:

* Создание нового пользователя
* Поиск пользователя по логину
* Поиск пользователя по маске имя и фамилии
* Создание товара
* Получение списка товаров
* Добавление товара в корзину
* Получение корзины для пользователя

**2. Диаграммы.**

**Контекстная диаграмма**

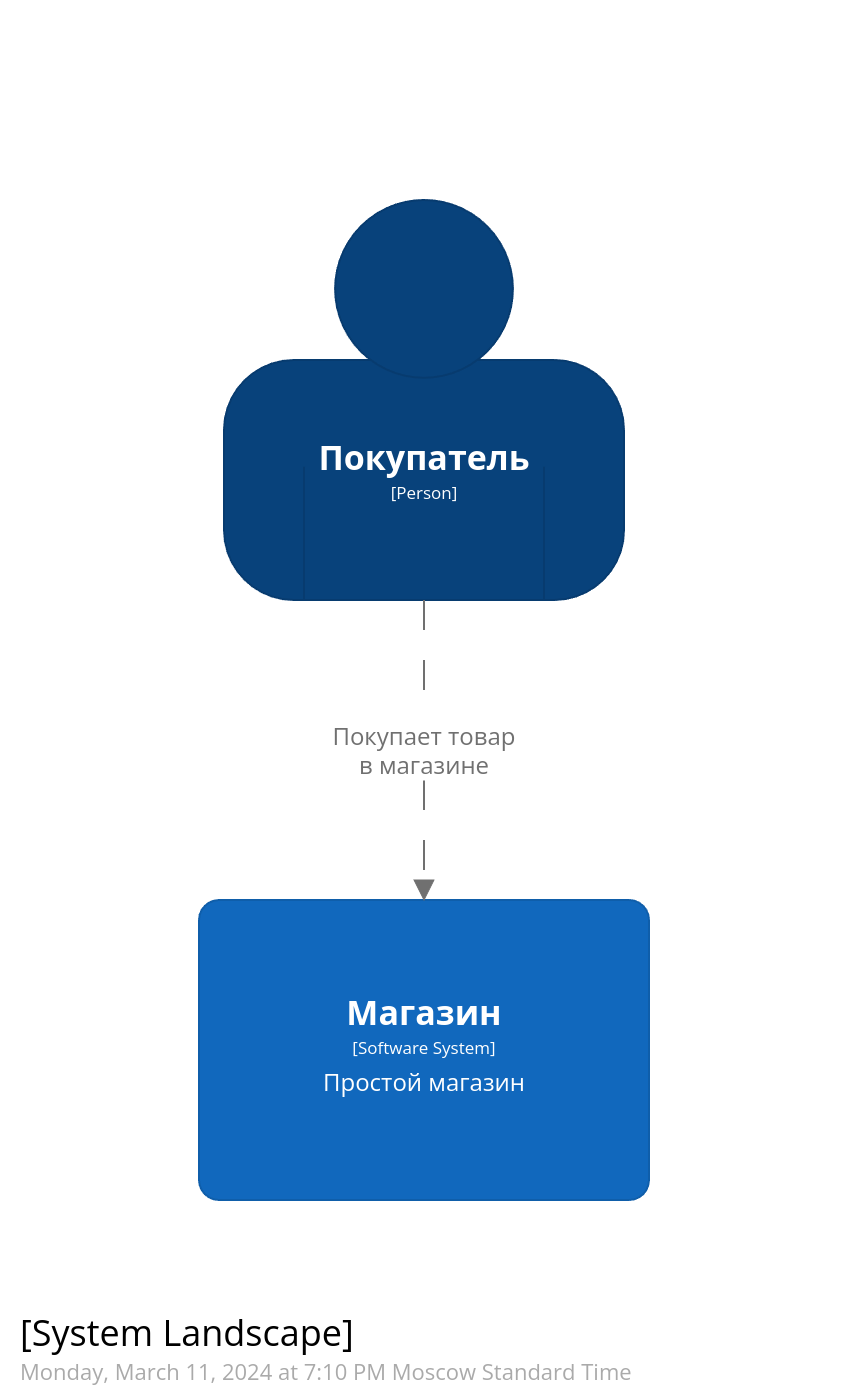
****

Рисунок 1. Контекстная диаграмма

**Диаграмма контейнеров**

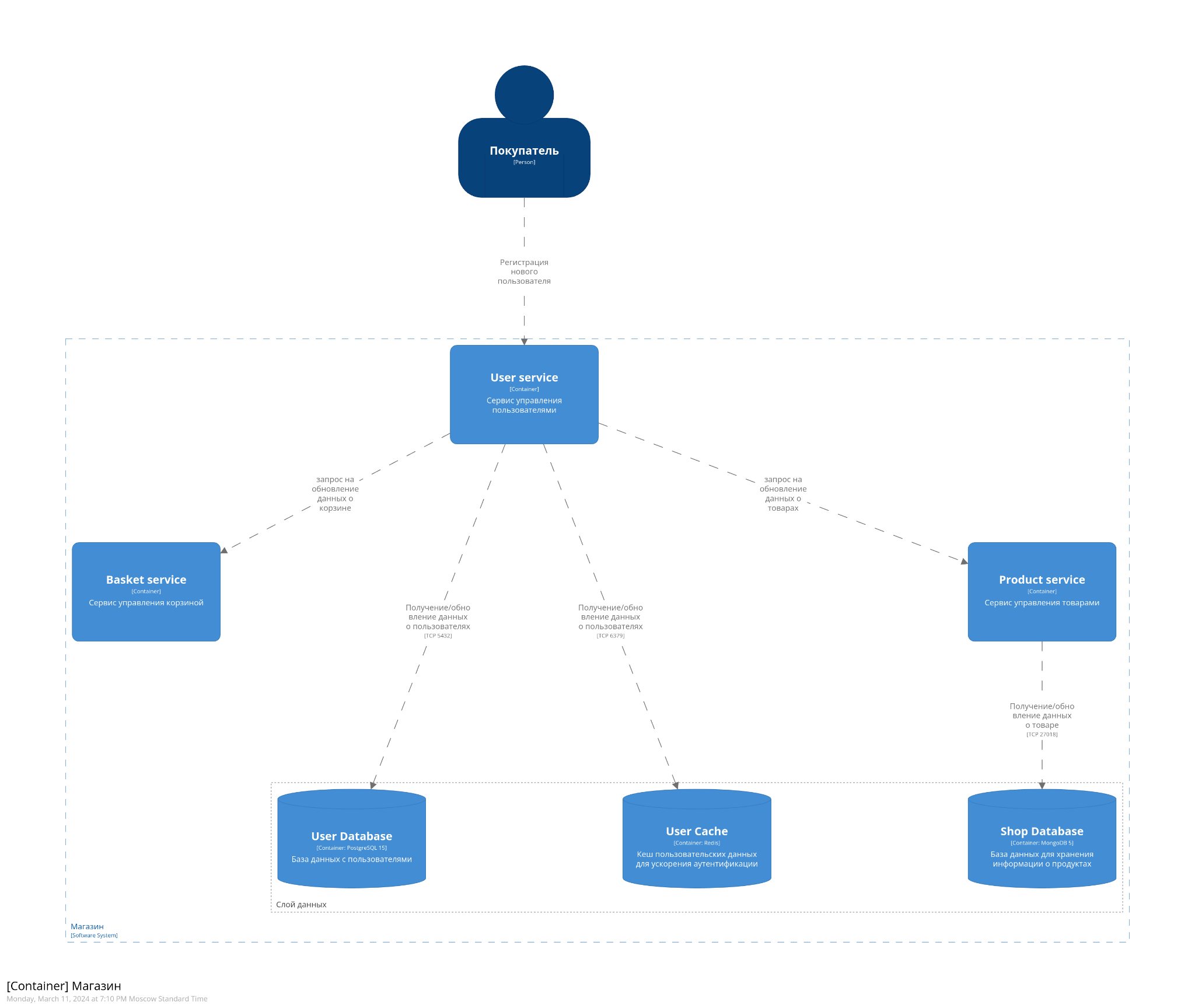


Рисунок 2. Диаграмма контейнеров

**Диаграмма развертывания**

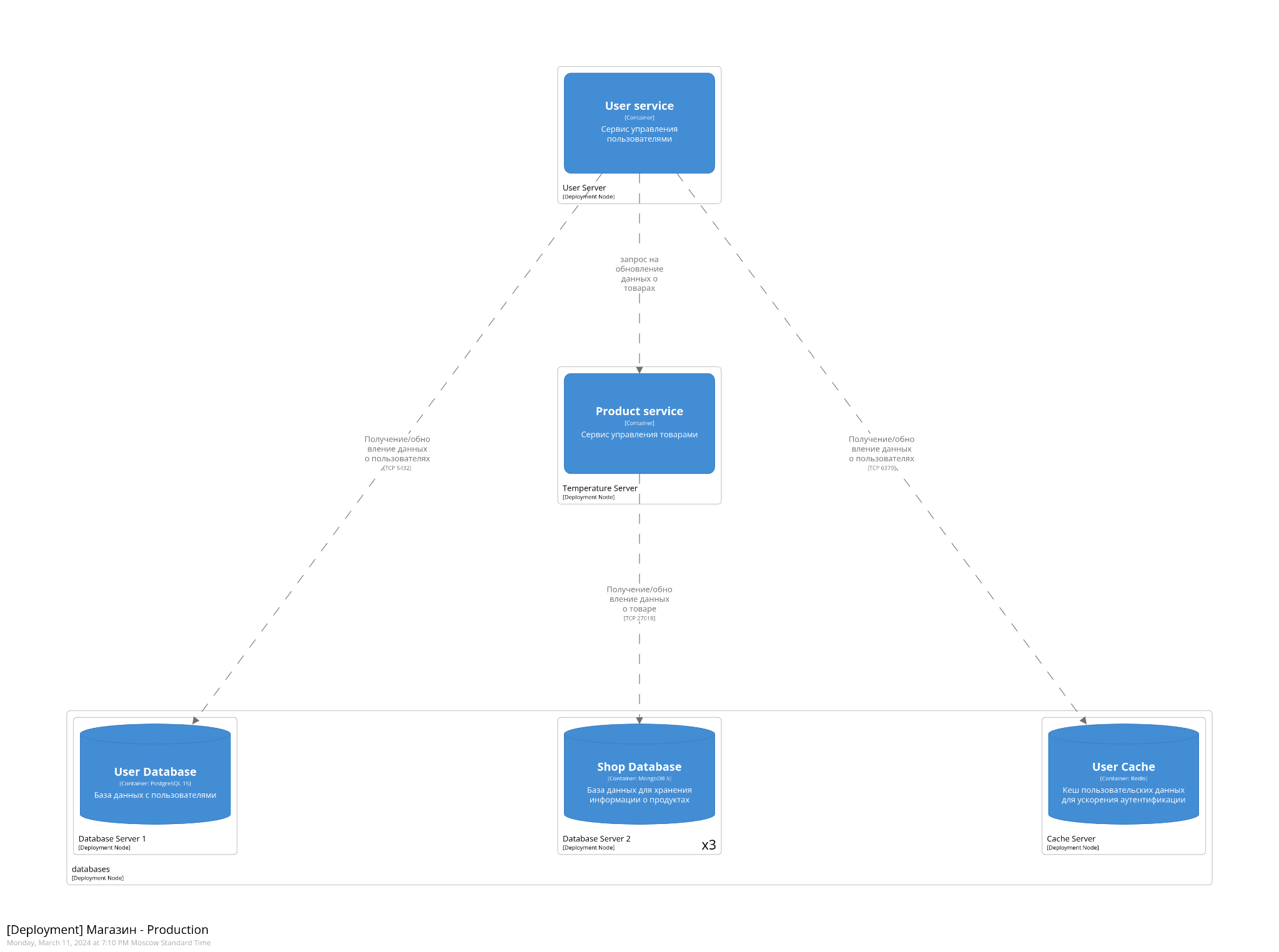


Рисунок 3. Диаграмма развертывания

**Динамические диаграммы**

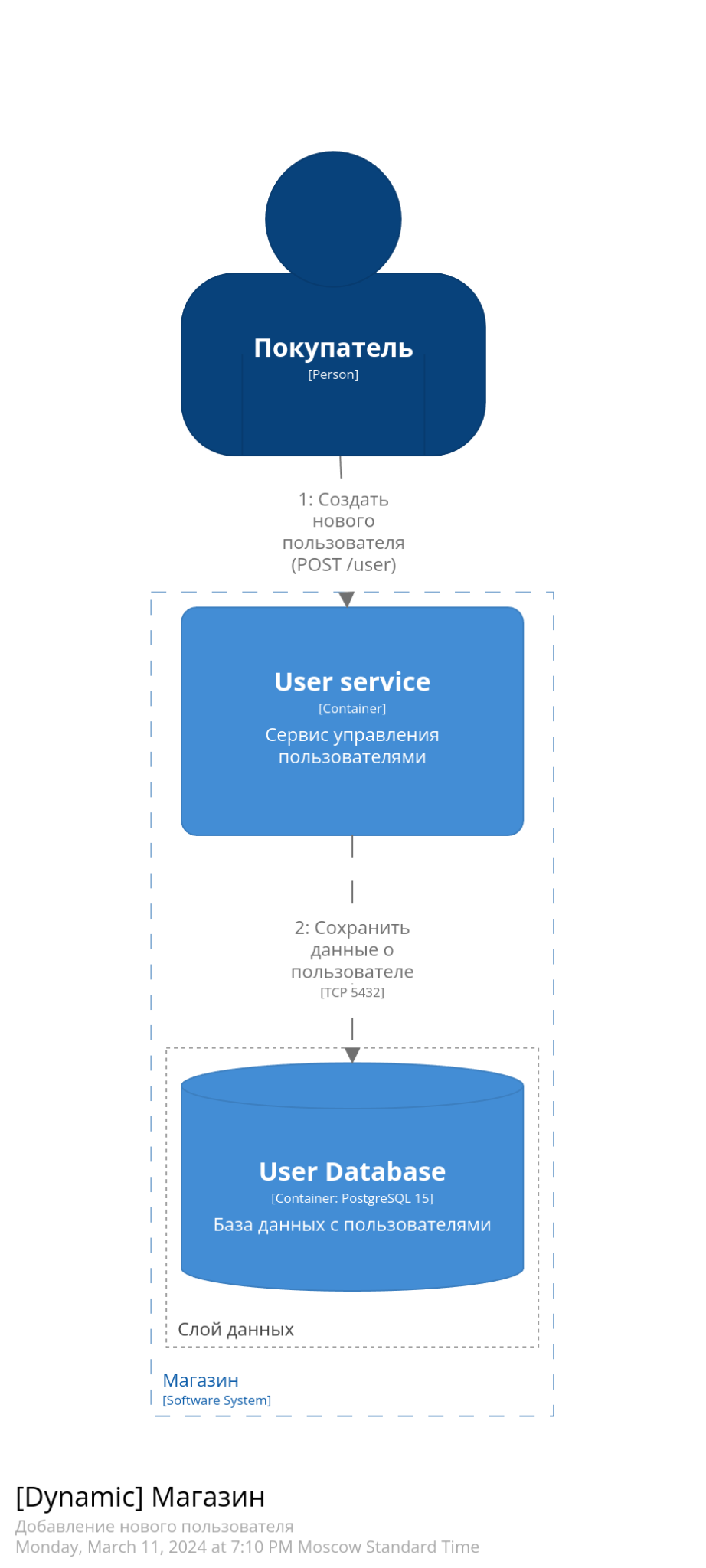


Рисунок 4. Динамическая диаграмма 1

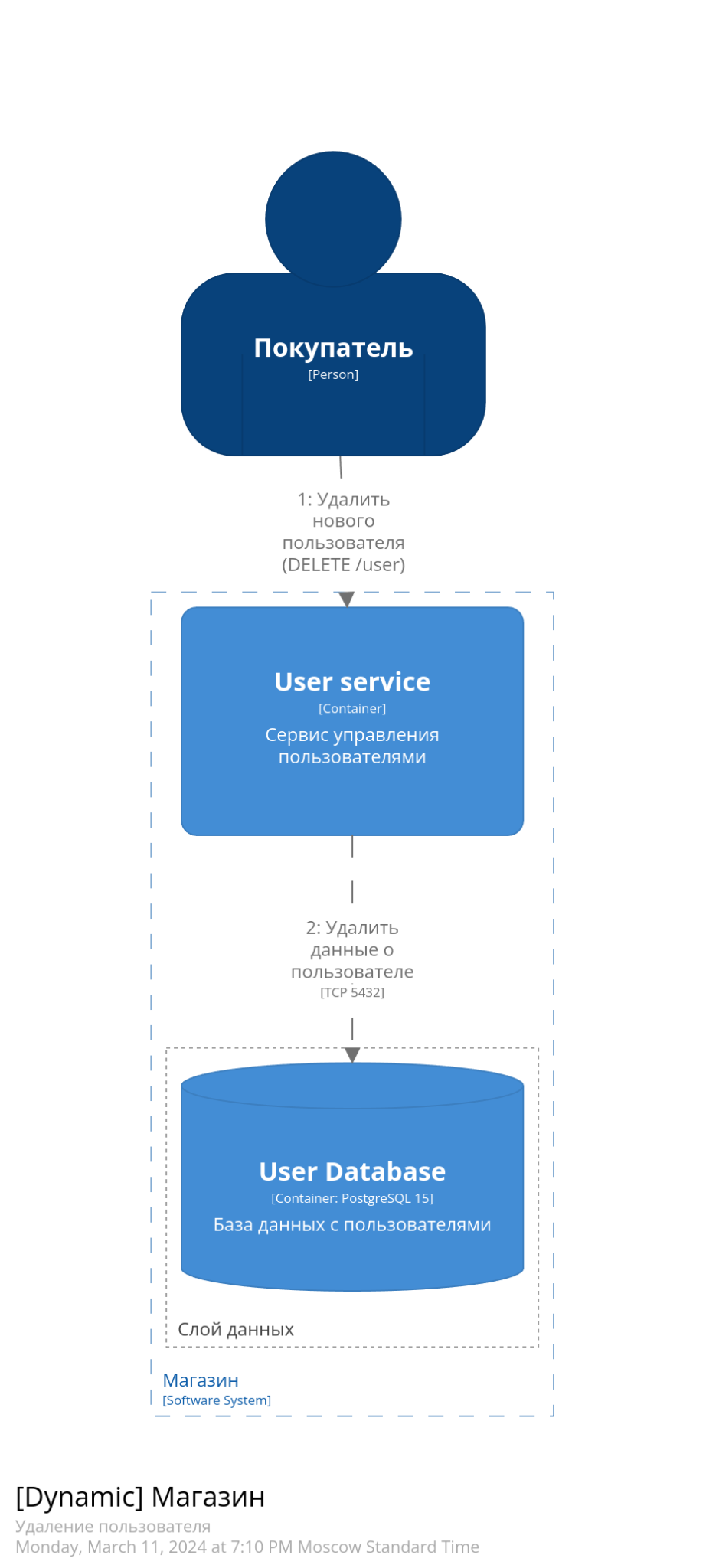


Рисунок 5. Динамическая диаграмма 2

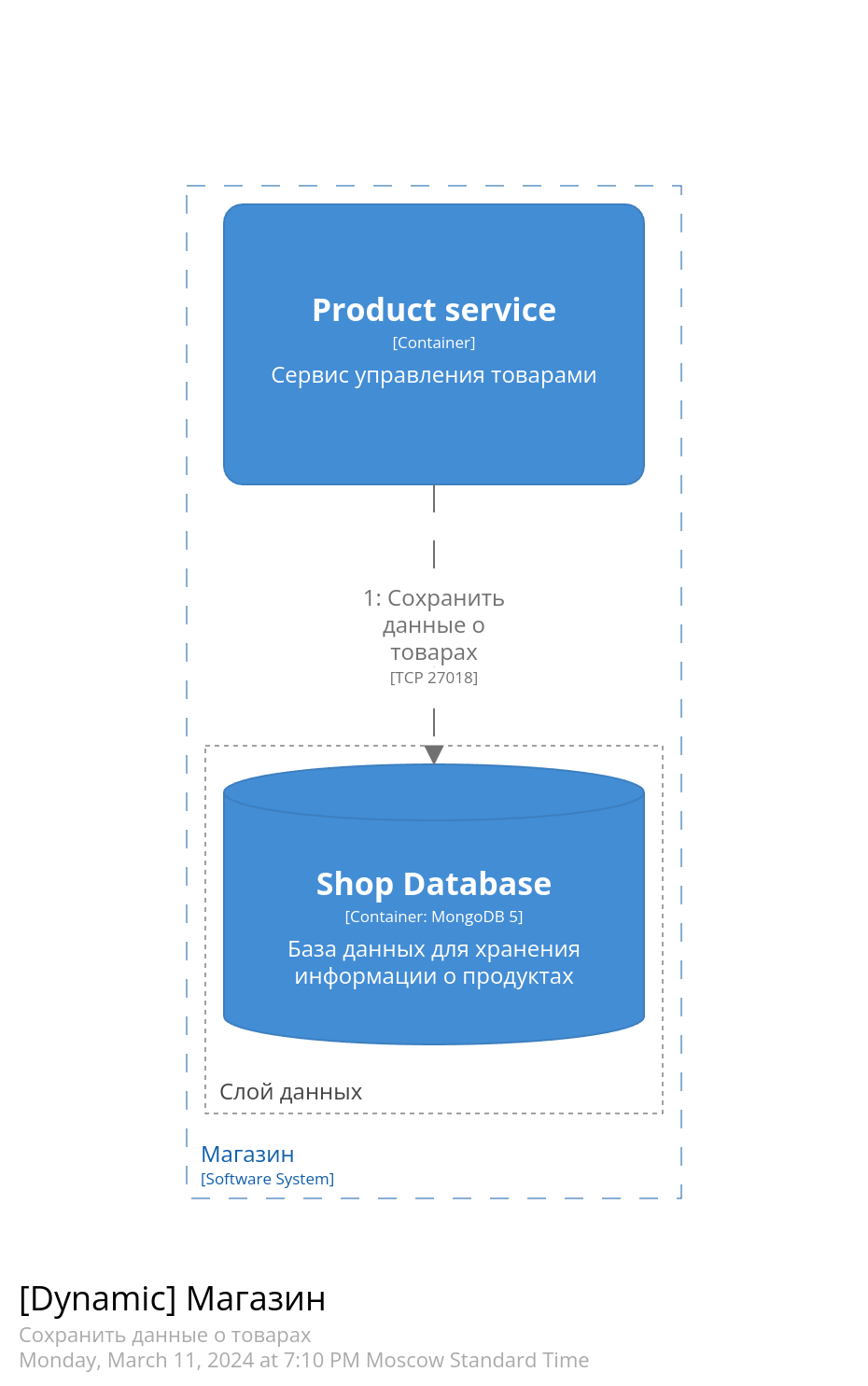


Рисунок 6. Динамическая диаграмма 3

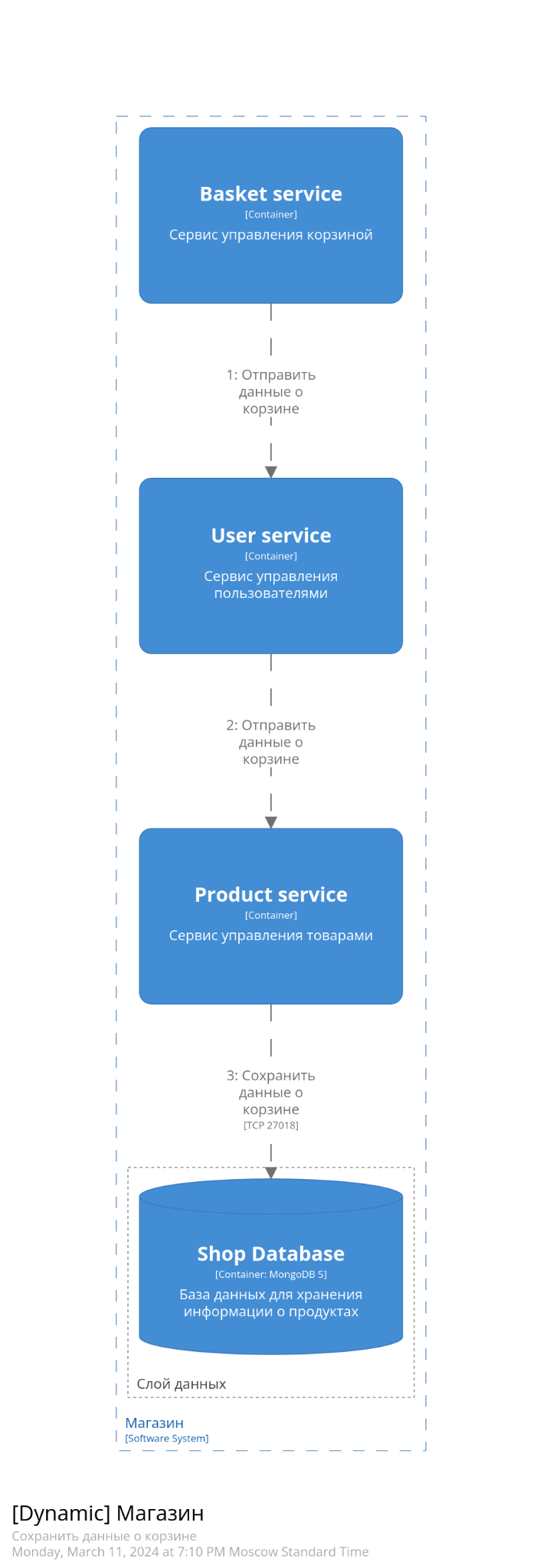


Рисунок 7. Динамическая диаграмма 4

**3. Код**

workspace {

name "Магазин"

description "Простая система магазина с базовыми функциями для покупателей"

# включаем режим с иерархической системой идентификаторов

!identifiers hierarchical

!docs documentation

!adrs decisions

# Модель архитектуры

model {

# Настраиваем возможность создания вложенных груп

properties {

structurizr.groupSeparator "/"

}

# Описание компонент модели

user = person "Покупатель"

shop = softwareSystem "Магазин" {

description "Простой магазин"

user\_service = container "User service" {

description "Сервис управления пользователями"

}

basket\_service = container "Basket service" {

description "Сервис управления корзиной"

}

product\_service = container "Product service" {

description "Сервис управления товарами"

}

group "Слой данных" {

user\_database = container "User Database" {

description "База данных с пользователями"

technology "PostgreSQL 15"

tags "database"

}

user\_cache = container "User Cache" {

description "Кеш пользовательских данных для ускорения аутентификации"

technology "Redis"

tags "database"

}

shop\_database = container "Shop Database" {

description "База данных для хранения информации о продуктах"

technology "MongoDB 5"

tags "database"

}

}

user\_service -> user\_database "Получение/обновление данных о пользователях" "TCP 5432"

user\_service -> user\_cache "Получение/обновление данных о пользователях" "TCP 6379"

user\_service -> basket\_service "запрос на обновление данных о корзине"

user\_service -> product\_service "запрос на обновление данных о товарах"

product\_service -> shop\_database "Получение/обновление данных о товаре" "TCP 27018"

}

user -> shop "Покупает товар в магазине"

user -> shop.user\_service "Регистрация нового пользователя"

deploymentEnvironment "Production" {

deploymentNode "User Server" {

containerInstance shop.user\_service

}

deploymentNode "Product Server" {

containerInstance shop.product\_service

properties {

"cpu" "4"

"ram" "256Gb"

"hdd" "4Tb"

}

}

deploymentNode "databases" {

deploymentNode "Database Server 1" {

containerInstance shop.user\_database

}

deploymentNode "Database Server 2" {

containerInstance shop.shop\_database

instances 3

}

deploymentNode "Cache Server" {

containerInstance shop.user\_cache

}

}

}

}

views {

themes default

properties {

structurizr.tooltips true

}

!script groovy {

workspace.views.createDefaultViews()

workspace.views.views.findAll { it instanceof com.structurizr.view.ModelView }.each { it.enableAutomaticLayout() }

}

dynamic shop "UC01" "Добавление нового пользователя" {

autoLayout

user -> shop.user\_service "Создать нового пользователя (POST /user)"

shop.user\_service -> shop.user\_database "Сохранить данные о пользователе"

}

dynamic shop "UC02" "Удаление пользователя" {

autoLayout

user -> shop.user\_service "Удалить нового пользователя (DELETE /user)"

shop.user\_service -> shop.user\_database "Удалить данные о пользователе"

}

dynamic shop "UC03" "Сохранить данные о товарах" {

autoLayout

shop.product\_service -> shop.shop\_database "Сохранить данные о товарах"

}

dynamic shop "UC04" "Сохранить данные о корзине" {

autoLayout

shop.basket\_service -> shop.user\_service "Отправить данные о корзине"

shop.user\_service -> shop.product\_service "Отправить данные о корзине"

shop.product\_service -> shop.shop\_database "Сохранить данные о корзине"

}

styles {

element "database" {

shape cylinder

}

}

}

}