Московский государственный технический университет им. Н. Э. Баумана

|  |  |
| --- | --- |
| УТВЕРЖДАЮ  Доцент МГТУ им. Н.Э. Баумана, к.ф.‑м.н.  Романова Татьяна Николаевна  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  Личная подпись Расшифровка   подписи  Печать  Дата | УТВЕРЖДАЮ  Разработчик  Кузьмин Артем Юрьевич  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  Личная подпись Расшифровка  подписи  Печать  Дата |

Распределённая система заказа такси

Шифр – «Система заказа такси»

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

На \_\_\_\_\_ листах

Действует с \_\_\_\_\_\_\_

СОГЛАСОВАНО

|  |  |
| --- | --- |
| Руководители | Романова Т. Н. |
|  | Вишневская Т. И. |

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

Личная Расшифровка

подпись подписи

Печать

Дата

# Перечень используемых сокращений и аббревиатур

**БД** – база данных

**ПО** – программное обеспечение

**СПО** – специальное программное обеспечение

**РСОИ** – распределённая система обработки информации

**СУБД** – система управления базами данных

**АИС** – автоматизированная информационная система

# Глоссарий

**РСОИ** – система независимых (с точки зрения администрирования) взаимодействующих автоматизированных систем;

**Программное обеспечение** – совокупность программ системы обработки информации и документации, необходимой для эксплуатации этих программ;

**Заявка** – единица обслуживания внутри РСОИ.

# Введение

Данное техническое задание составлено для проектирования РСОИ «Распределённая система заказа такси» (далее система заказа такси). Техническое задание выполнено на основе ГОСТ 19.201-78. ЕСПД. Техническое задание. Требования к содержанию и оформлению.

## Наименование предприятий разработчика и заказчика системы

Разработчиком системы является Кузьмин А.Ю., студент группы ИУ7-29 кафедры ИУ-7 «Программное обеспечение вычислительной техники и автоматизированных систем» МГТУ им. Н.Э. Баумана.

Заказчиком системы является кафедра ИУ-7 «Программное обеспечение вычислительной техники и автоматизированных систем. МГТУ им. Н.Э. Баумана» (далее Кафедра).

## Описание предметной области

Система заказа такси предназначена для подбора клиентам (пользователям системы) ближайших к ним свободных таксистов. Наибольшую сложность процесса заказа такси для клиента представляет поиск такси среди большого числа таксопарков. Клиенту необходимо связаться с несколькими таксопарками, чтобы сделать удовлетворяющий его выбор такси. Таксопарки, в свою очередь, заинтересованы в предоставлении своих услуг как можно большему числу клиентов. Чем ближе в данный момент времени находится свободный таксист к клиенту, сделавшему заказ, тем выгоднее как для клиента – такси приедет быстрее, так и для таксопарка – уменьшаются затраты на топливо для автомобилей. Каждая из организаций создает собственный узел проектируемой РСОИ – АИС, или подключается к существующему узлу. АИС разных организаций объединяются каналами связи и представляют собой системы, действующие независимо друг от друга. Проектируемая РСОИ позволит автоматизировать обработку заявок на подбор автомобилей, тем самым уменьшив человеческие трудозатраты.

Общий вид модели предметной области представлен на Рис. 1.

Рис 1. Модель предметной области



## Существующие аналоги

Крупнейшим существующим аналогом является сервис «Яндекс.Такси». Сервис позволяет вызывать официальное такси и наблюдать за его движением на карте. Вызов такси осуществляется через веб-интерфейс.

## Описание системы

Система заказа такси предназначена для предоставления пользователям возможности оперативно формировать вызов таксистов по указанному адресу в кратчайшее время. При вызове таксиста осуществляется связь с диспетчерским центром, где в автоматизированном режиме для пользователя подбираются ближайшие к нему и удовлетворяющие запросу такси.

## Состав системы

Система заказа такси должна состоять из следующих узлов:

- диспетчерская АИС;

- АИС в таксопарках;

- АИС, устанавливающиеся на автомобили таксистов;

### Основные задачи диспетчерской АИС

Основные задачи диспетчерской АИС – предоставление клиенту информации о наличии рядом с ним свободных таксистов, удовлетворяющих его условиям заказа. А также связь с различными таксопарками и оформление заказа такси при подтверждении клиента.

### Основные задачи АИС таксопарков

Основные задачи АИС таксопарков – поиск свободных таксистов, готовых принять заказ. А также создание заявки для таксиста при подтверждении клиента.

### Основные задачи АИС на автомобилях таксистов

Основные задачи АИС на автомобилях таксистов – взаимодействовать с АИС таксопарка, сообщая текущие координаты водителя и статус его готовности принять заказ.

### Архитектура системы

Топология системы заказа такси представлена на рис. 2.

Рис. 2 Топология системы заказа такси



## Сроки выполнения работ по созданию системы

Начало выполнения работ - 1 неделя весеннего семестра 2013 г.

Окончание выполнения работ - 14 неделя весеннего семестра 2013 г.

# Основания для разработки

Разработка ведется в рамках выполнения лабораторных работ по курсу «Методология программной инженерии», а так же в рамках выполнения курсового проекта по курсу «Распределенные системы обработки информации» с учетом указаний, описанных в методическом пособии «Распределенные системы обработки информации. Указания к курсовому проектированию».

# Назначение разработки

Распределенная система заказа такси предназначена для оформления заявок клиентов на предоставление автомобиля в указанную точку в кратчайшее время, посредством взаимодействия клиентов с web-интерфейсом.

В системе может присутствовать любое количество географически разнесенных узлов, которые могут иметь различные технические платформы и находиться под управлением различных ОС.

Диспетчерская система предназначена для взаимодействия с клиентом и осуществления связи с различными таксопарками. БД системы хранит информацию о зарегистрированных пользователях и осуществляемых ими заказов.

АИС таксопарков предназначена для предоставления сведений о наличии таксистов, готовых принять заказ, и удовлетворяющих заявке клиента. БД системы хранит информацию о таксистах, работающих в данном таксопарке.

АИС, устанавливающаяся на автомобили таксистов, предназначена для предоставления таксопарку текущих координат таксиста и статуса его готовности принять заказ.

# Требования к программному комплексу

## Требования к РСОИ

### Требования к функциональным характеристикам

#### Система должна иметь возможность добавления новых узлов.

#### Каждый узел системы должен иметь возможность конфигурирования для обеспечения взаимодействия с участниками РСОИ.

#### Узлы РСОИ должны проектироваться исходя из предположения об отсутствии абсолютно надежной связи.

#### Система должна предоставлять пользователю функции по формированию заявки на вызов такси, ее подтверждению или отмене.

## Требования к диспетчерской системе

### Требования к функциональным характеристикам

#### Система должна обеспечивать доступность на уровне 23/7. Допускаются ежедневные профилактические работы сроком не более часа.

#### Система должна обеспечивать одновременную обработку до 100 заявок на вызов такси.

#### Время отклика системы не должно превышать 3 секунд.

### Функциональные требования с точки зрения пользователя

Пользователями системы являются физические и юридические лица, желающие заказать такси.

#### Система должна предоставлять web-интерфейс для взаимодействия.

#### Система должна предоставлять возможность регистрации и авторизации пользователей.

#### Система должна предоставлять авторизированному пользователю возможность перехода на страницу с историей предыдущих заказов.

#### Система должна предоставлять пользователю возможность выбора фильтра поиска такси по классу комфорта. Возможные значения: «Эконом», «Комфорт», «Бизнес».

#### Система должна предоставлять пользователю возможность выбора фильтра поиска такси по цене за километр.

#### Система должна предоставлять пользователю возможность выбора фильтра поиска такси по цене за минуту.

#### Система должна предоставлять пользователю возможность осуществления заказа такси посредством web-интерфейса.

#### Система должна возвращать незарегистрированному пользователю код доступа для возможности получения текущего статуса его заказа.

#### Система должна предоставлять авторизованному пользователю возможность перехода на страницу с отображением текущего статуса заказа.

#### Система должна предоставлять незарегистрированному пользователю возможность ввода кода доступа для получения текущего статуса заказа.

#### Система должна предоставлять незарегистрированному пользователю возможность ввода кода доступа для просмотра текущего статуса заявки.

#### Система должна предоставлять пользователю возможность подтверждения или отмены заявки посредством web-интерфейса.

### Входные параметры системы

#### Система должна предоставлять возможность использования фильтра поиска по следующим полям:

|  |  |  |
| --- | --- | --- |
| **Имя параметра** | **Тип параметра** | **Комментарии** |
| Класс комфорта автомобиля | Список | Допустимые значения списка: «Эконом», «Комфорт», «Бизнес» |
| Верхняя граница цены за километр | Действительное число | Можно задать, если не задано поле «Верхняя граница цены за минуту» |
| Верхняя граница цены за минуту | Действительное число | Можно задать, если не задано поле «Верхняя граница цены за километр» |

#### Система должна предоставлять возможность регистрации нового пользователя с использованием следующих параметров

|  |  |  |
| --- | --- | --- |
| **Имя параметра** | **Тип параметра** | **Комментарии** |
| Логин пользователя | Строковы |  |
| Пароль пользователя | Строковый |  |
| Имя пользователя | Строковый |  |
| Фамилия пользователя | Строковый |  |
| Отчество пользователя | Строковый |  |

#### Система должна предоставлять возможность авторизации пользователя с использованием следующих параметров

|  |  |  |
| --- | --- | --- |
| **Имя параметра** | **Тип параметра** | **Комментарии** |
| Логин пользователя | Строковы |  |
| Пароль пользователя | Строковый |  |

### Выходные параметры системы

#### Система должна выводить информацию в web-интерфейсе.

#### Система должна предоставлять возможность просмотра следующих параметров:

|  |  |  |
| --- | --- | --- |
| **Имя параметра** | **Тип параметра** | **Комментарии** |
| Текущий статус заявки | Строковый | Может содержать следующие значения: «Заявка принята таксистом», «Ожидание», «Не найдено подходящего такси» |
| Текущие координаты автомобиля | Метка на карте | Данный параметр присутствует, если таксист принял заказ и выехал к клиенту |
| Код доступа для возможности просмотра текущего статуса заявки | Строковый | Предоставляется незарегистрированным пользователям |
| Список автомобилей такси, найденных по заданному фильтру | Список ссылок | Предоставляется возможность перехода по ссылкам для просмотра детального описания автомобиля такси |
| История предыдущих заказов | Список строк | Возможность просмотра истории предыдущих заказов предоставляется зарегистрированным пользователям |

## Требования к АИС таксопарков

### Требования к функциональным характеристикам

#### Система должна предоставлять информацию о таксистах, удовлетворяющих условиям заявки клиента.

#### Система должна предоставлять возможность назначения таксистам нового задания.

#### Время отклика системы не должно превышать 30 секунд.

### Функциональные требования с точки зрения пользователя

Пользователем системы является диспетчерская АИС.

#### Система должна обеспечивать пользователю возможность получения списка свободных таксистов.

#### Система должна обеспечивать пользователю возможность получения списка свободных таксистов.

## Требования к АИС такси

### Требования к функциональным характеристикам

#### Система должна предоставлять текущие координаты автомобиля.

#### Система должна предоставлять текущий статус такси.

#### Время отклика системы не должно превышать 10 секунд.

### Функциональные требования с точки зрения пользователя

Пользователем системы является диспетчерская АИС и АИС таксопарка.

#### Система должна обеспечивать пользователю возможность получения текущих координат такси.

#### Система должна обеспечивать пользователю возможность получения текущего статуса такси.

### Выходные параметры системы

|  |  |  |
| --- | --- | --- |
| **Имя параметра** | **Тип параметра** | **Комментарии** |
| Текущий статус такси | Строковый | Может содержать следующие значения: «Выполняется заказ», «Свободен» |
| Текущие координаты автомобиля | Строковый | Строка с координатами автомобиля |

## Сценарии функционирования

Для выполнения любого сценария необходимо войти в систему через интерфейс пользователя. В системе допустимы 2 типа пользователей – гость и зарегистрированный пользователь. Сценарии функционирования для пользователей с категорией прав «Гость» и «Зарегистрированный пользователь» отражены на рис. 3. и рис.4 соответственно.



Рис. 3 Диаграмма использования для пользователя с категорией прав «Гость».



Рис.4 Диаграмма использования для пользователя с категорией прав «Зарегистрированный пользователь».

### Спецификации прецедентов использования

#### **Спецификация прецедента «Регистрация»**

Прецедент использования позволяет действующему лицу зарегистрироваться в системе.

#### **Сценарий**:

1. Выполнить вход в web-интерфейс.
2. В web-интерфейсе перейти к регистрации по ссылке.
3. Заполнить поля «Логин пользователя», «Пароль», «Подтверждение пароля», «Фамилия», «Имя», «Отчество».
4. Зарегистрироваться, нажав кнопку «Подтвердить» или отказаться от регистрации, нажав кнопку «Отмена».

#### Спецификация прецедента «Авторизация»

Прецедент использования позволяет действующему лицу войти в систему.

**Сценарий:**

**Основной сценарий,** возможен, если пользователь является зарегистрированным пользователем

1. Выполнить вход в web-интерфейс.
2. В web-интерфейсе перейти ко входу по ссылке.
3. Заполнить поля «Логин» и «Пароль».
4. Подтвердить вход в систему, нажав кнопку «Войти» или отменить вход, нажав кнопку «Отмена».

**Альтернативный сценарий**: если пользователь не является зарегистрированным пользователем, то необходимо вначале выполнить регистрацию и попробовать войти в систему после.

#### Спецификация прецедента «Осуществление заказа»

Прецедент использования позволяет действующему лицу сформировать заявку на заказ такси.

#### **Основной сценарий**: возможен, если пользователь имеет категорию прав «Зарегистрированный пользователь»

1. Выполнить вход в web-интерфейс.
2. Перейти на страницу заказа такси.
3. Заполнить поля фильтра поиска: «Класс комфорта», «Цена за километр», «Цена за минуту»
4. Выполнить поиск такси, нажав кнопку «Поиск».
5. Выбрать из списка подходящее такси
6. Оформить заявку на выбранное такси, нажав кнопку «Заказать».

**Альтернативный сценарий,** если пользователь имеет категорию прав «Гость»

1. Выполнить вход в web-интерфейс.
2. Перейти на страницу заказа такси.
3. Заполнить поля фильтра поиска: «Класс комфорта», «Цена за километр», «Цена за минуту»
4. Выполнить поиск такси, нажав кнопку «Поиск».
5. Выбрать из списка подходящее такси
6. Оформить заявку на выбранное такси, нажав кнопку «Заказать».
7. Получить код доступа для возможности просмотра статуса заказа
8. Сохранить код доступа

#### Спецификация прецедента «Просмотр текущего статуса»

Прецедент использования позволяет пользователю посмотреть текущий статус сделанного заказа.

#### **Основной сценарий**: возможен, когда пользователь имеет категорию прав «Зарегистрированный пользователь»

1. Выполнить вход в web-интерфейс.
2. Если вход в систему не выполнен, выполнить авторизацию.
3. Перейти на страницу просмотра текущего статуса заказа.

**Альтернативный сценарий,** если пользователь имеет категорию прав «Гость»

1. Выполнить вход в web-интерфейс.
2. Перейти на страницу просмотра текущего статуса заказа.
3. Ввести сохраненный код доступа в поле «Код для просмотра статуса»
4. Нажать кнопку «Посмотреть статус»

#### Спецификация прецедента "Просмотр истории"

Прецедент использования позволяет зарегистрированному пользователю посмотреть историю предыдущих заказов.

#### **Основной сценарий**: возможен, когда пользователь имеет категорию прав «Зарегистрированный пользователь».

1. Выполнить вход в web-интерфейс.
2. Если вход в систему не осуществлен, выполнить авторизацию.
3. Перейти на страницу просмотра истории предыдущих заказов.

# Диаграмма последовательности взаимодействия



Рис. 5.1 Диаграмма последовательности взаимодействия систем, входящих в распределенную систему заказа такси.

# Диаграмма деятельности



Рис. 6.1. Диаграмма деятельности для варианта использования «Заказ такси».

# Диаграмма развертывания



Рис. 7.1 Диаграмма развертывания систем, входящих в распределенную систему заказа такси.

# Диаграмма коммуникаций



Рис. 8.1 Диаграмма коммуникаций систем, входящих в распределенную систему заказа такси.

# Тестирование

## Базовое тестирование

Для базового тестирования использовался дополнительный инструмент SOAP UI, который позволяет производить тестирование методов веб-сервисов с определенными заданными параметрами и анализировать результат обработки запросов к веб-сервису.

### Модульное тестирование

Модуль диспетчерской АИС представляет собой веб-сервис и содержит классы, реализующие функциональность диспетчерской АИС, а именно: регистрация пользователя (registerUser), авторизация пользователя (authorizeUser), осуществление заказа такси (addUserApplication), получение статуса заказа (getStatus), получение истории предыдущих заказов пользователя (getUserHistory). Результаты тестирования методов веб-сервиса приведены в таблице 9.1.

**Таблица 9.1 — Модульное тестирование Диспетчерской АИС.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ теста** | **Тестируемый метод** | **Описание теста** | **Степень важности** | **Ожидаемый результат** | **Результат теста** |
| 1. | registerUser | Попытка регистрации нового пользователя с корректно заполненными полями: имя, логин, пароль | Фаталь-  ная | Успешная регистрация | Тест  пройден |
| 2. | registerUser | Попытка регистрации нового пользователя при отсутствии значений некоторых необходимых полей | Фаталь-  ная | Получение сообщения об ошибке <Данные для регистрации заполнены не полностью>. | Тест пройден |
| 3. | authorizeUser | Проверка авторизации пользователя с корректно заполненными полями логина и пароля | Фаталь-  ная | Успешная авторизация | Тест не  пройден |
| 4. | authorizeUser | Проверка авторизации пользователя с некорректно заполненными полями логина или пароля | Фаталь-  ная | Получение сообщения об ошибке <Проверьте правильность введенных логина и пароля> | Тест  пройден |
| 5. | addUserApplication | Создание заявки на заказ такси для зарегистрированного пользователя, с корректно заполненными полями фильтра поиска такси | Фаталь-  ная | Система успешно добавляет новую заявку в локальную БД, начинает обработку новой заявки и возвращает статус об успешном принятии заявки в обработку | Тест пройден |
| 6. | addUserApplication | Создание заявки на заказ такси для зарегистрированного пользователя, с некорректно заполненным полем даты доставки (указана прошедшая дата) | Фаталь-  ная | Система проверяет фильтр поиска такси и возвращает сообщение о невозможности принятия заказа в обработку с описанием ошибки | Тест  пройден |
| 7. | getStatus | Проверка получения статуса текущего заказа пользователя, если заказ находится в обработке | Фаталь-  ная | Диспетчерская система проверяет текущий статус заказа пользователя в локальной БД и возвращает его | Тест не  пройден |
| 8. | getStatus | Проверка получения статуса текущего заказа пользователя, при отсутствии заказов у пользователя | Фатальная | Диспетчерская система проверяет текущий статус заказа пользователя в локальной БД, не находит текущего заказа и возвращает сообщение с соответствующим описанием | Тест пройден |
| 9. | getUserHistory | Проверка получения истории предыдущих заказов пользователя при наличии таких заказов | Фатальная | Диспетчерская система проверяет историю заказов пользователя в локальной БД и возвращает список заказов | Тест пройден |
| 10. | getUserHistory | Проверка получения истории предыдущих заказов пользователя при отсутсвии таких заказов | Фатальная | Диспетчерская система проверяет историю заказов пользователя в локальной БД, не находит заказов и возвращает сообщение с соответствующим описанием | Тест пройден |

**Вывод:** тестирование модуля Диспетчерской АИС, за исключением двух тестов, пройдено успешно.

**Исходный код модульного тестирования для метода регистрации нового пользователя.**

|  |
| --- |
| import org.junit.Assert.assertEquals;  import com.ibm.db2.jcc.\*;  import java.sql.\*;  public class DispAISTest {  /\*\*  \* Запрос на добавление нового пользователя.  \*/  **private** **static** **final** String *ADD\_NEW\_USER* =  "insert into rsoi\_disp.user (user\_guid, user\_login, user\_pwd, user\_name, user\_lname, user\_mname) " +  "values (?, ?, ?, ?, ?, ?)";  @UnitTest  public void testRegisterNewUser ( ) {  Connection conn = **null**;  PreparedStatement stmt = **null**;  **boolean** result = **false**;  **if** (newUser == **null** ||  newUser.getLogin() == **null** || "".equals(newUser.getLogin()) ||  newUser.getPwd() == **null** || "".equals(newUser.getPwd()) ||  newUser.getName() == **null** || "".equals(newUser.getName())) {  **return** result;  }    **try** {  conn = ConnectionFactory.*getInstance*().getConnection();  stmt = conn.prepareStatement(*ADD\_NEW\_USER*);    **int** paramIndex = 1;  stmt.setString(paramIndex++, UUID.*randomUUID*().toString());  stmt.setString(paramIndex++, newUser.getLogin());  stmt.setString(paramIndex++, newUser.getPwd());  stmt.setString(paramIndex++, newUser.getName());  **if** (newUser.getLname() == **null** || "".equals(newUser.getLname())) {  stmt.setNull(paramIndex++, Types.*VARCHAR*);  } **else** {  stmt.setString(paramIndex++, newUser.getLname());  }    **if** (newUser.getMname() == **null** || "".equals(newUser.getMname())) {  stmt.setNull(paramIndex++, Types.*VARCHAR*);  } **else** {  stmt.setString(paramIndex++, newUser.getMname());  }    result = stmt.executeUpdate() == 1 ? **true** : **false**;    } **catch** (SQLException e) {  *logger*.error("Error accessing DB", e);  result = **false**;    } **finally** {  SQLUtils.*closeSQLObjects*(conn, stmt, **null**);  }  **assertEquals(result, true);**  } |

## Интеграционное тестирование

Данный раздел содержит результаты интеграционного тестирования всей распределенной системы в целом, а именно взаимодействия Диспетчерской системы с АИС Таксопарков и АИС Такси. Все системы представлены в виде веб-сервисов, и тестирование будет производиться с помощью инструмента SOAP UI, позволяющего создать имитацию тестовых запросов для инициирования взаимодействия систем. Результаты интеграционного тестирования приведены в таблице 9.2.

**Таблица 9.2 — Интеграционное тестирование распределенной системы в целом.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ теста** | **Тестируемый метод** | **Описание теста** | **Степень важности** | **Ожидаемый результат** | **Результат теста** |
| 1. | updateOrderStatus | Проверка обновления кэша статуса заказа клиента по истечении таймаута | Фаталь-ная | Кэш ресурсов обновляется через заданные промежутки времени и закэшированный статус текущего заказа совпадает с текущим статусом АИС Такси после обновления | Тест  пройден |
| 2. | addNewApplication | Проверка  успешного  оформления  тестовой  заявки | Фаталь-ная | Заявки в Диспетчерской АИС, АИС Таксопарка и АИС Такси успешно добавлены и обработаны. Их параметры совпадают с параметрами, указанными в тестовом запросе. | Тест  пройден |
| 3. | addNewApplication | Проверка  неуспешного  оформления  тестовой  заявки | Фаталь-ная | Заявки в Диспетчерской АИС, АИС Таксопарка, АИС Такси, соответствующие исходной тестовой заявке, находятся в состоянии <отменена> и их параметры совпадают с параметрами,  указанными в тестовом запросе. | Тест  пройден |
| 4. | canncelApplicati | Проверка отмены тестовой заявки | Фаталь-ная | После запроса на отмену тестовой заявки  состояние соответствующих  заявок в Диспетчерской АИС, АИС Таксопарка и АИС Такси переходят в состояние <отменена>. | Тест  пройден |

**Вывод:** интеграционное тестирование распределенной системы пройдено успешно. Диспетчерская АИС, АИС Таксопарка, АИС Такси взаимодействуют в соответствии с разработанными диаграммами взаимодействия. Все тесты пройдены успешно.

**Исходный код интеграционного тестирования для кэширования текущего статуса заказа.**

|  |
| --- |
| import org.junit.Assert.assertEquals;  import com.ibm.db2.jcc.\*;  import java.sql.\*;  public class PaymentSystemTest {  @UnitTest  public void testPayment ( ) {  ApplicationDAO appDao = **new** ApplicationDAO();  String orderDetailGuid = appDao.getOrderDetailGuid(correlationID);    OrderDetailsDAO odDao = **new** OrderDetailsDAO();  String status = odDao.checkStatus(orderDetailGuid);    // Нужно еще кэшировать  **if** (!"completed".equals(status)) {  Map<String, String> properties = **new** LinkedHashMap<String, String>();  properties.put(MsgProps.*ACTION\_PROP*, "status");    **try** {  MsgSender.*sendMessage*(dest, **null**, properties, *mqProps*.getProperty("dispTaxiQueue"), correlationID);  StatusCache.*cacheStatus*(seconds, dest, correlationID);  } **catch** (JMSException e) {  *logger*.error("Error", e);  }  } } |

## Системное тестирование

### Тестирование функциональных возможностей

Набор тестов для тестирования функциональных возможностей получен по диаграмме прецедентов выделением для каждого прецедента классов эквивалентности. Классом эквивалентности называется совокупность входных параметров, приводящих систему в одно и то же состояние. Основной и альтернативный сценарии, описанные при создании диаграммы прецедентов, принадлежат к одному классу эквивалентности. Выделенные классы эквивалентности представлены в таблице 9.3. На основании этого было проведено тестирование функциональных возможностей (см. таблицу 9.4)

**Таблица 9.3 — Классы эквивалентности для прецедента авторизации пользователя и заказа такси.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Прецедент** | **Описание класса эквивалентности** | **Состояние системы** | **Пример входных данных** |
| 1 | Авторизация  пользователя | Зарегистрированная в системе пара <логин-пароль> (основной сценарий) | Авторизация  пользователя  прошла  успешно | Логин:  arkuzmin  Пароль: qwe123 |
| 2 | Авторизация  пользователя | Зарегистрированный в системе логин и неверный пароль (альтернативный сценарий) | Пользователь  не авторизован | Логин:  arkuzmin  Пароль: qwe1 |
| 3 | Авторизация  пользователя | Неверный логин и верный пароль (альтернативный сценарий) | Пользователь  не авторизован | Логин:  Arkuzmin2  Пароль: qwe123 |
| 4 | Авторизация  пользователя | Не зарегистрированная в системе пара <логин-пароль>  (альтернативный сценарий) | Пользователь  не авторизован | Логин:  Arkuzmin2  Пароль: qwe1 |
| **Тестирование проверки вводимых данных** | | | | |
| 5 | Заказ такси | Оформление заявки на заказ такси с корректно заполненными полями фильтра поиска автомобиля | Оформление заявки проходит успешно | Время доставки: 25.05.2013 18:00  Класс комфорта: Эконом  Макс. Цена за минуту: 30р. |
| 6 | Заказ такси | Оформление заявки на заказ такси с корректно заполненными полями фильтра поиска автомобиля, кроме поля времени доставки – указана прошедшая дата | Оформление заявки проходит неуспешно. | Время доставки: 25.05.1888 18:00  Класс комфорта: Эконом  Макс. Цена за минуту: 30р. |
| 7 | Заказ такси | Оформление заявки на заказ такси с наличием пустых обязательных для заполнения полей | Оформление заявки проходит неуспешно. | Время доставки: Класс комфорта: Эконом  Макс. Цена за минуту: 30р. |

**Таблица 9.4 — Результаты тестирования функциональных возможностей.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Описание теста** | **Входные данные** | | **Ожидаемый результат** | **Реальный**  **результат** | **Резу-льтат**  **теста** |
| **Тестирование авторизации** | | | | | | |
| 1 | Авторизация с зарегестри-рованной  парой (логин, пароль) | (arkuzmin, qwe123) | Авторизация проходит  успешно; система выполняет переход со  страницы авторизации  на главную страницу;  в верхнем левом углу  отображается имя  пользователя и ссылка для выхода из системы;  По центру страницы пользователю доступны пункты: заказ такси, просмотр текущего статуса заказа, просмотр истории предыдущих заказов | | Система перешла  на главную  страницу  и вывела все  требуемые данные. | Тест  пройден |
| 2 | Авторизация с верным паролем и неверным логином | (kuzmin, qwe123) | Авторизация проходит  неуспешно; система  отображает страницу  авторизации и выводит  сообщение об ошибке  <Пожалуйста, введите  верные логин и пароль>. | | Система осталась  на странице  авторизации,  сообщение  об ошибке  выведено на  экран. | Тест  пройден |
| 3 | Авторизация с верным логином и неверным паролем | (arkuzmin, qww) | Авторизация проходит  неуспешно; система  отображает страницу  авторизации и выводит  сообщение об ошибке  <Пожалуйста, введите  верные логин и пароль>. | | Система осталась  на странице  авторизации,  сообщение  об ошибке  выведено на  экран. | Тест  пройден |
| 4 | Авторизация с незарегистрированной  парой (логин, пароль) | (afn, qwewe) | Авторизация проходит  неуспешно; система  отображает страницу  авторизации и выводит  сообщение об ошибке  <Пожалуйста, введите  верные логин и пароль>. | | Система осталась  на странице  авторизации,  сообщение  об ошибке  выведено на  экран. | Тест  пройден |

**Вывод:** тестирование функциональных возможностей не выявило расхождений с техническим заданием, тесты пройдены успешно.

### Тестирование формы

Для тестирования граничных значений и проверки вводимых данных выделены классы эквивалентности, представленные в таблице 9.3. Результаты тестирования формы авторизации пользователя представлены в таблице 9.5)

**Таблица 11.5 — Результаты тестирования формы авторизации пользователя.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Описание теста** | **Входные данные** | **Ожидаемый результат** | **Реальный**  **результат** | **Резу-льтат**  **теста** |
| **Тестирование проверки вводимых данных** | | | | | |
| 1 | Вход в  систему | Введены верные  логин и пароль | Авторизация проходит  успешно; система выполняет переход с формы  авторизации на главную форму. | Система перешла на главную форму | Тест  пройден |
| 2 | Вход в систему | Введены неверные логин и пароль | Авторизация проходит  неуспешно; система  остается на форме авторизации и выводит сообщение об ошибке <Пожалуйста, введите верные логин и пароль>. | Система осталась  на форме  авторизации  и вывела сообщение об  ошибке. | Тест  пройден |
| 3 | Заказ такси | Введены неверные данные | Система выполняет переход на форму с описанием ошибки, новая заявка на заказ такси не формируется. | Система перешла на форму с описанием ошибки, новая заявка не сформировалась | Тест  пройден |
| 4 | Заказ такси | Введены верные данные | Система переходит на страницу с сообщением об успешном формировании новой заявки. Новый заказ появляется в таблице с историей заказов пользователя. | Система перешла на страницу с сообщением об успешном формировании новой заявки. Новый заказ появился в таблице с историей заказов. | Тест  пройден |
| 5 | Отмена  заказа | Идентификатор текущего заказа | Кнопка отмены заказа доступна только для текущего заказа в статусе «Выполняется» | Кнопка доступна только для заказов с указанным статусом. | Тест пройден |

## Вывод

Все тесты по модульному, интеграционному тестированию и тестированию пользовательских функций пройдены успешно. Никаких недостатков или отклонений от технического задания не обнаружено. Приложение может быть введено в эксплуатацию.

# Требования к надежности

1. Подключить к серверу системы заказа такси источник бесперебойного питания, обеспечивающий автономную работу не менее 20 мин.
2. Выполнять резервное копирование базы данных один раз в неделю.
3. Хранить дату удаления узлов системы.
4. Производить журналирование нормального и критического хода работы системы.
5. Обеспечить работу системы в режиме 22/7/365. В течение дня в нерабочее время предусмотреть возможность проведения профилактических работ, не превышающих 2 часов. В случае возникновения внештатных ситуаций, не связанных с работой СПО, время восстановления после сбоя не должно превышать 5 часов.

# Условия эксплуатации

Требования к функционированию СПО на сервере:

Для обеспечения бесперебойного функционирования установить кондиционеры, поддерживающие следующие климатические условия:

Обеспечить температурный режим от 15 до 30 °C.

Относительная влажность – не более 80%.

Расположить компоненты сервера на жёстких металлических стойках.

Оборудование должно быть заземлено.

# Требования к составу и параметрам технических средств

1. Минимальные технические требования:

* 2-х ядерный процессор с тактовой частотой 2 ГГц;
* ОЗУ 4 ГБ;
* ПЗУ 300 ГБ для СПО;
* сетевая карта Ethernet стандарта 1000BASE-T;
* дисковый массив с возможностью дополнительного подключения жёстких дисков;
* ИБП.

1. Программное окружение:

* операционные системы:
  + семейства Windows NT: Microsoft Windows Server 2008 архитектуры x64;
  + на ядре Linux: Ubuntu 12.04, Red Hat 6.2;
* Java 1.6 архитектуры x64;
* Web-сервер Apache Tomcat 7.21;

# Требования к информационной и программной совместимости

Разработка серверной части СПО должна вестись с использованием платформенно-независимого ПО.

Разработка клиентской части СПО должна производиться с учётом особенностей ОС: ОС Windows или ОС на базе ядра Linux.

# Требования к программной документации

Список программной документации, представляемый к системе, должен включать в себя:

1. техническое задание – 1 шт. в печатном виде и 1 шт. в электронном виде в формате .PDF;
2. программный продукт на Flash накопителе – 1 шт.;
3. документация пользователя – 1 шт. в печатном виде и 1 шт. в электронном виде в формате .PDF.

# Технико-экономические показатели

Технико-экономические показатели рассчитываются заказчиком в расчётно-пояснительной записке.

# Стадии и этапы разработки

В процессе разработки СПО необходимо использовать следующие стадии разработки (таблица 1).

Таблица 1 – Этапы разработки

|  |  |  |
| --- | --- | --- |
| Этап | Описание | Сроки выполнения |
| Техническое задание | Производится постановка задачи, уточняются требования к СПО, анализируется предметная область, существующие системы бронирования и выделяются достоинства и недостатки. Выбираются языки программирования, ОС и ПО, необходимые для разработки. Формируется введение и аналитический раздел расчётно-пояснительной записки. | 14 календарных дней |
| Эскизный проект | Выделение необходимых структур данных, проектирование протокола взаимодействия между узлами системы. Формализация логики обработки сообщений в виде конечного автомата. Проектирование алгоритмов работы с использованием нотаций. Объектная декомпозиция и прототипирование основных алгоритмов. Тестирование созданных структур. Создание интерфейса взаимодействие между серверной и клиентской части СПО. Формируется раздел проектирования расчётно-пояснительной записки. | 28 календарных дней |
| Технический проект | Детальное проектирование СПО. Разработка модульных тестов. Использование итерационного подхода в разработке СПО. Формируется технологический раздел расчётно-пояснительной записки. | 35 календарных дней |
| Рабочий проект | Закупка (при необходимости) и установка необходимого ПО. Исправление ошибок, обнаруженных на предыдущем этапе. Подготовка к развёртыванию системы. Проведение системного тестирования. Окончательное формирование расчётно-пояснительной записки. | 14 календарных дней |
| Внедрение | Развертывание системы. Подготовка программной документации. Сдача системы. | 14 календарных дней |

# Порядок контроля и приемки системы

1. В процессе разработки СПО проводить модульное тестирование.
2. Испытания СПО должны проводиться на объектах заказчика и исполнителя в оговоренные сроки.
3. На завершающем этапе разработки провести системное тестирование СПО.

# Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Планирование основных работ по внедрению подсистемы включает:

1. Установка ПО.
2. Инструктаж персонала к работе.

## Функциональные требования к системе с точки зрения пользователя

РСОИ должна обеспечить реализацию следующих функций:

1. Регистрацию новых пользователей;
2. Идентификацию и аутентификацию пользователей системы на основе имени и пароля;
3. Возможность сохранения мест, в которые пользователь уже вызывал такси и предоставление удобного доступа к сохраненным местам для упрощения работы пользователя с системой.
4. Обеспечить работу системы в режиме 23/7/365.

# Список стандартов и литературы

Использованные нормативные документы и литература при разработке системы:

1. ГОСТ 19.201-78. ЕСПД. Техническое задание. Требования к содержанию и оформлению.
2. ГОСТ 19.102-77. ЕСПД. Стадии разработки
3. Вишневская Т.И., Романова Т.Н. Технология программирования: Метод. указания к лабораторному практикуму. - Ч. 2. – М: Изд-во МГТУ им. Н.Э. Баумана, 2009