**Система оценки лабораторных работ**

1. Есть понятие – «сдача в срок», если срок нарушается, то происходит снижение оценки. Сроки проведения каждой лабораторной видны в ОРИОКСе.
2. Баллы, указанные в ОРИОКСе - это максимальная оценка, такую оценку студент получает, если работа сдана в срок, также учитывается качество исполнения и защиты работы. Если работа выполнена небрежно или же у преподавателя есть другие замечания, оценка снижается, даже если работа сдана в срок.
3. Если работа сдается на текущем или следующем занятии, то это «сдача в срок»

Например, без снижения оценки:

1-ую лабораторную работу можно сдать на 1-ом или 2-ом занятии;

4-ую лабораторную работу можно сдать на 4-ом или 5-ом занятии и т.п.

Таким образом, дается запас по времени, чтобы вы спокойно, без спешки

выполняли работы.

1. Если студент сдает 1-ую лабораторную на 3-ем занятии (с опозданием на одно занятие), то оценка снижается на 1 балл, если 1-ая работа сдается на 4-ом занятии, то оценка снижается на 2 балла и т.д.
2. За одно занятие (2 пары) можно сдать не более **ДВУХ** лабораторных работ!
3. Лабораторные работы можно защищать только по порядку. То есть, после первой ЛР **нельзя** сразу сдавать четвертую или пятую.
4. **Обязательно** использование для команды системы контроля версий *(например GitVerse).* При сдаче лабораторных работ демонстрировать репозиторий с распределением и выполнением задач каждым участником команды.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **№ ЛР** | **Неделя** | | | | | | | | |
| **2** | **4** | **6** | **8** | **10** | **12** | **14** | **16** | **18** |
| **1** | 5 | 5 | 4 | 3 | 2 | 1 | 1 | 1 | 1 |
| **2** |  | 5 | 5 | 4 | 3 | 2 | 1 | 1 | 1 |
| **3** |  |  | 5 | 5 | 4 | 3 | 2 | 1 | 1 |
| **4** |  |  |  | 5 | 5 | 4 | 3 | 2 | 1 |
| **5** |  |  |  |  | 5 | 5 | 4 | 3 | 2 |
| **6** |  |  |  |  |  | 5 | 5 | 4 | 3 |
| **7** |  |  |  |  |  |  | 5 | 5 | 4 |
| **8** |  |  |  |  |  |  |  | 5 | 5 |

**8 Лабораторных работ**

**4 блока**

**1 блок: Информационная модель**

Команды максимум по 3 человека. Проект для выполнения выбираете самостоятельно.

1. Use-case (минимум 3 сценария)
2. Диаграммы классов (общая диаграмма)
3. Диаграммы взаимодействий (для каждого сценария)

**2 блок: Figma** https://www.figma.com

1. Прототипы экранных форм
2. Схема экранов (с дизайном)
3. Схема экранов (с дизайном) часть 2

**3 блок: Тестирование**

1. Goms-модель (временные затраты для каждого сценария)

**4 блок: Защита проекта**

1. Продайте свой продукт! 5-7 минут выступление не более 10 слайдов (или другой вид презентации)

**Голосование:** 1место – экзамен автомат «отлично»!

2 место – экзамен автомат «хорошо»!

Дополнительные поощрения лучшим проектам!

**Последнее слово за преподавателем.**

**P.S. тестирования, контрольные работы и баллы за посещение лекций выставляет лектор Федоров П.А.**

**Лабораторная работа №1**

**Структура сценария использования**

Сценарии использования включают в себя следующие разделы:

1. **Название.** Краткое, максимально понятное. Описывающее общее действие пользователя.

*Пример:*

UC-1. Регистрация в личном кабинете

UC-2. Регистрация в программе лояльности

UC-3. Добавление товара в корзину

1. **Акторы (пользователи).** *Например*, в магазине онлайн в качестве акторов выступают и продавцы, и покупатели, а также компании, осуществляющие доставку товара или обеспечивающие поступление платежей.
2. **Предусловие.** Формулировка условий, при которых данный вариант использования может быть инициирован.

Условие, помимо прочего, может быть упоминанием о выполнении других вариантов использования. Также в предусловии необходимо указывать, в какой части системы находится пользователь, кратко-какие действия уже выполнил.

*Пример:* пользователь находится в “Корзине”, в “Корзине добавлено 2 товара”.

Данное предусловие мы можем указать для описания кейса работы пользователя в “Корзине”. Если мы описываем кейс “Добавление товара в корзину” или “Оформление заказа”, где необходимо указать всю цепочку шагов пользователя – то данное предусловие не подойдет.

1. **Основной сценарий.** Сценарий – это последовательность шагов, описывающая процесс решения задачи, которой посвящен вариант использования. Шаги удобно последовательно нумеровать.

**Альтернативные сценарии**, в которых процесс развития событий на каком-либо шаге чем-либо заметно отличается от основного, то есть имеет место ветвление.

Сценарий использования должен отвечать на вопрос “Что делает пользователь?” “Что делает система?”

При описании сценария использования важно соблюдать пошаговый план действий пользователя, указывая физическое действие пользователя.

*Например*, формулировка“добавил товар в корзину” неверная.

*Правильно:* “нажимает на кнопку “Добавить товар в корзину” и далее – реакцию системы на действия пользователя.

|  |  |
| --- | --- |
| **Пользователь** | **Система** |
| Какое физическое действие произвел пользователь? | Как отреагировала система? |
| Нажимает “Добавить в корзину” | Система добавляет выбранный товар в корзину. В иконке “Корзина” система выводит маркер- кол-во добавленного товара в корзину.  Изменяет кнопку “Добавить в корзину” у выбранного товара на кнопку “Перейти в корзину” |
| Пользователь нажимает “перейти в корзину” | Система переводит пользователя в корзину, где отображается добавленный товар. |

1. **Альтернативные сценарии**

При проработке основного сценария, все варианты действий пользователя и поведения системы, отличных от основного сценария необходимо выносить в альтернативный сценарий. То есть, везде, где можно указать “если” – это и будет альтернативный сценарий.

Важно! Альтернативный сценарий должен ссылаться только на один успешный сценарий. Недопустимо прописывать альтернативный сценарий для альтернативного сценария.

*Рассмотрим на примере авторизации*

Предусловие: неавторизованный пользователь находится на странице авторизации и регистрации

|  |  |
| --- | --- |
| **Пользователь** | **Система** |
| Какое физическое действие произвел пользователь? | Как отреагировала система? |
| Пользователь нажал кнопку “Зарегистрироваться” | Система вывела форму регистрации, поле “email” |
| Пользователь вводит данные в поле “email” | 3. Система производит проверку введенных данных на валидацию. Данные проходят по условиям валидации  Если данные не прошли проверку валидации, запускается альтернативный сценарий №1. |
|  | Система производит поиск введенных данных “email” по учетным записям в системе.  Учетных записей с такими данными “email” не найдено.    Если в системе найдена учетная запись с таким логином, запускается альтернативный сценарий №2 |
|  | Система отправляет пользователю код подтверждения на email  Система выводит пользователю поле “код подтверждения”  Сценарий “ввод кода подтверждения” вынесен в отдельный сценарий. - обязательно указываем, если какой-либо функционал выносим в отдельный кейс, более подробный. |
| Пользователь вводит корректный код подтверждения в поле “код подтверждения” | Система производит проверку кода подтверждения. Код введен верно.  Пользователь зарегистрирован.  Альтернативный сценарий с неверным кодом подтверждения выносим в сценарий “Ввод кода подтверждения” |

*Пример альтернативного сценария*

**Альтернативный сценарий №1**

1)На шаге №3 успешного сценария, введенные данные не прошли проверку валидации.

2)Система выводит информер с указанием запрещенных символов.

3)Пользователь вводит корректные данные в поле “email”.

4)Далее сценарий продолжается от шага №3 успешного сценария.

**Что прочитать:**

1. https://gb.ru/blog/use-case/
2. https://o2k.ru/blog/use-case#:~:text=Use%20Case%2C%20%D1%81%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%B8%D0%B9%20%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F%2C%20%D1%8E%D0%B7%D0%BA%D0%B5%D0%B9%D1%81,%D1%81%D0%B4%D0%B5%D0%BB%D0%B0%D1%82%D1%8C%20%D1%81%20%D0%BA%D0%B5%D0%BC%20%D0%B8%20%D1%87%D0%B5%D0%BC.

**Лабораторная работа №2**

**Class diagram, или диаграмма классов** — статическая диаграмма, являющаяся основным логическим представлением системы. Данная диаграмма определяет типы классов системы и связи, существующие между ними. Также на диаграмме могут отображаться атрибуты и операции классов, ограничения связей, различные спецификации. Вид диаграммы и набор элементов, изображенных на ней, зависит от уровня абстракции.

Для проектируемой системы изображается диаграмма, отображающая основные классы-сущности и связи между ними.

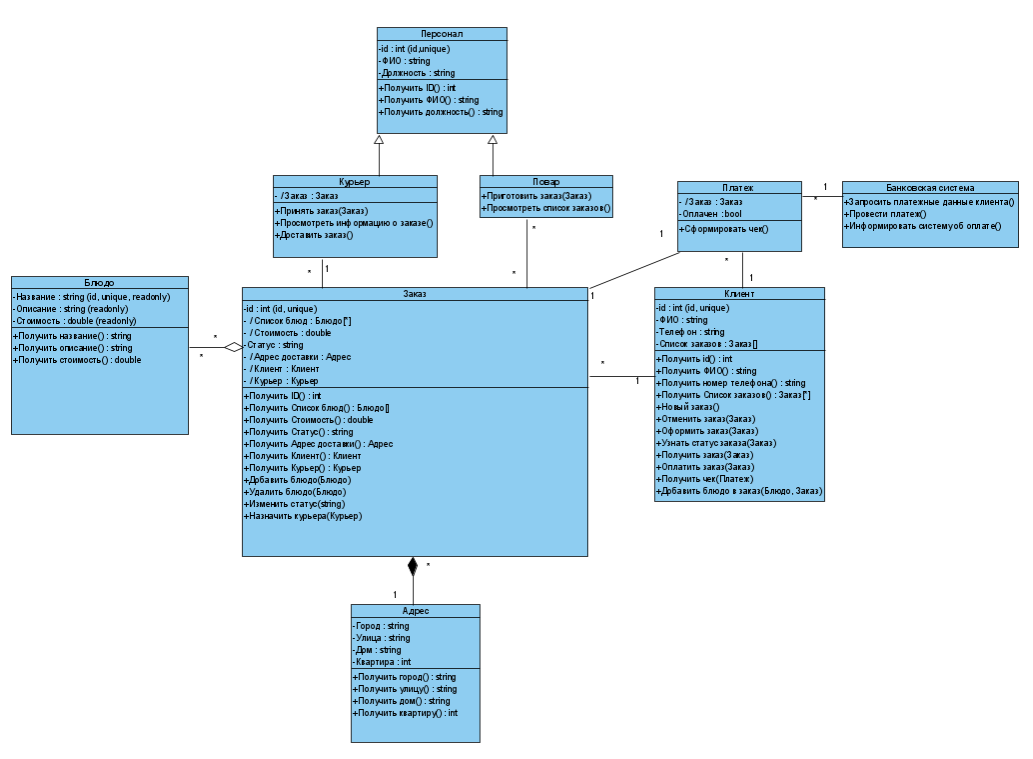


Рисунок 1. Диаграмма классов

**Что почитать:**

1. <https://flexberry.github.io/ru/gpg_class-diagram.html>
2. <https://habr.com/ru/articles/511798/>

**Лабораторная работа №3**

**Sequence diagram, или диаграмма последовательности** — предназначена для моделирования взаимодействия объектов информационной системы во времени и обмена сообщения между ними. По аналогии с остальными UML-диаграммами, для элементов данной диаграммы могут быть указаны различные спецификации (виды сообщений, временные ограничения, стереотипы классов) в зависимости от уровня абстракции.

На рис. 2 представлена диаграмма, отражающая взаимодействие актеров, граничных классов (экранных форм) и контроллера (класса, который координирует действия других элементов системы).

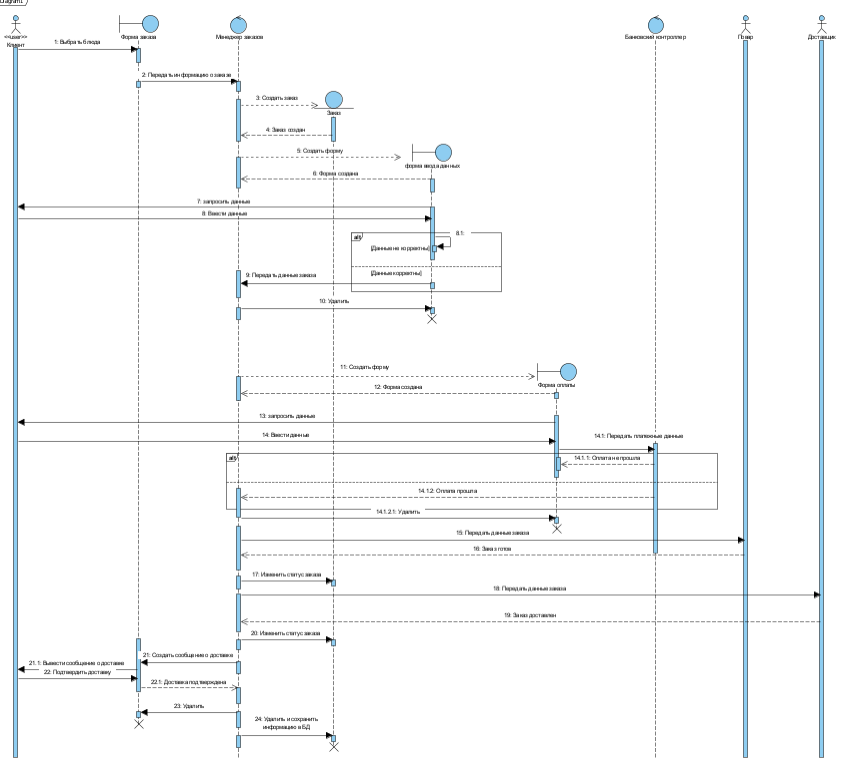


Рисунок 2. Диаграмма взаимодействий

**Лабораторная работа №4**

Перед началом разработки дизайна экранных форм необходимо разработать макеты, на которых будет только формальная расстановка кнопок, основных блоков.

Также указывается связь между экранами (при нажатии на кнопки-переходы).

Для разработки макетов можно использовать Figma (https://www.figma.com), PowerPoint или другое ПО.

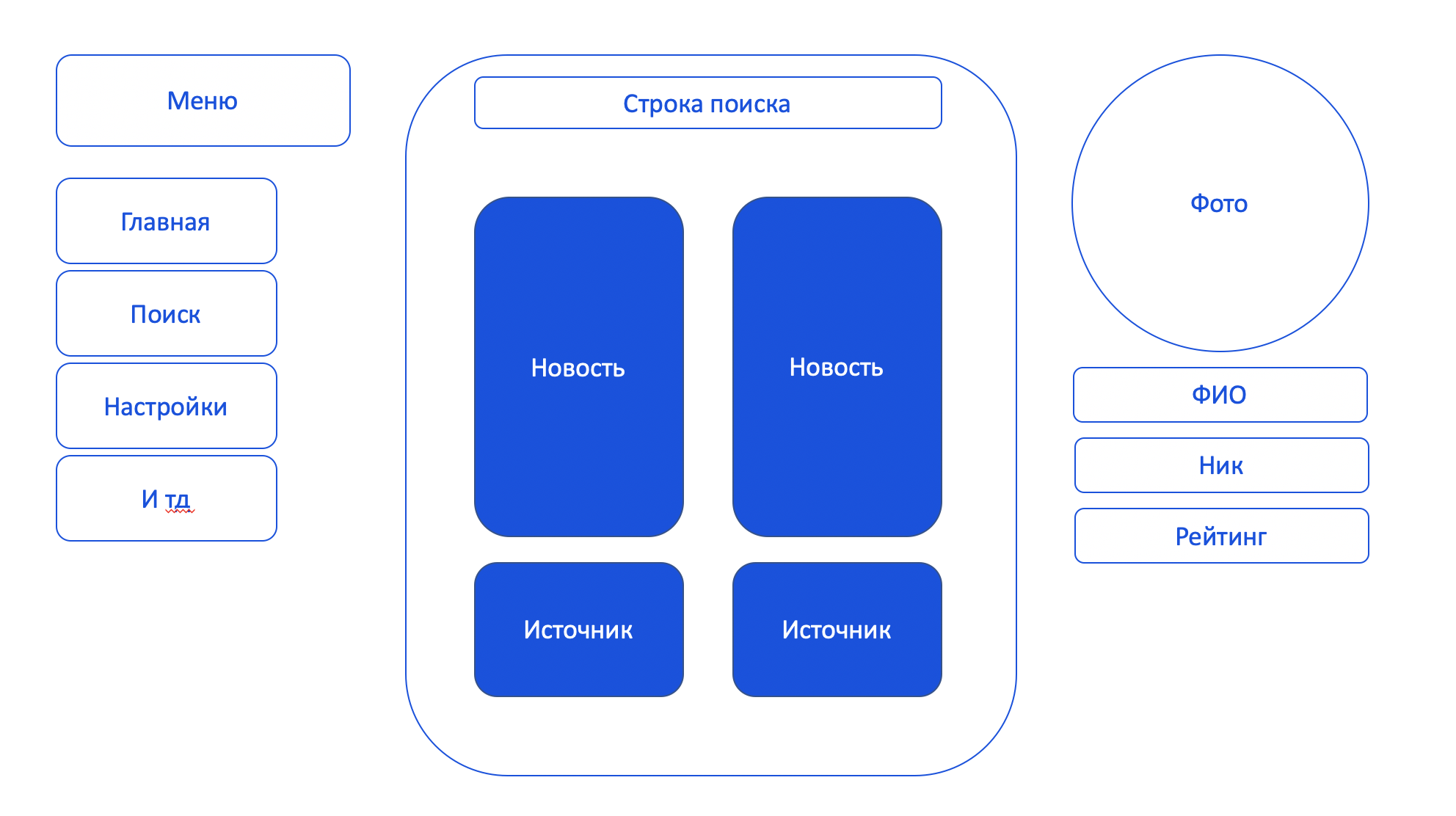


Рисунок 3. Пример макета экранной формы

**Лабораторная работа №5-6**

Разработка дизайна экранных форм (https://www.figma.com).

Использование цветового круга для определения дизайна проекта: **https://colorscheme.ru**





Рисунок 4. Примеры экранных форм

**Лабораторная работа №7**

«The model of Goals, Objects, Methods, and Selection rules» (GOMS) — это метод исследования интерфейса, разработанный Кардом, Мораном и Ньюэллом в 80-х годах. GOMS позволяет предсказать, сколько времени потребуется опытному (именно опытному) пользователю на выполнение конкретной операции при использовании конкретного интерфейса.

**Жесты и время по модели GOMS:**

**H (перенос руки на мышь) = 0,4 сек**

**К (нажатие клавиши клавиатуры или мыши) = 0,2 сек**

**Р (перенос курсора к позиции на экране) = 1,1 сек**

**М (обдумывание следующего шага) = 1,35 сек**

**R (ожидание ответа системы) — время зависит от быстродействия конкретной системы и не участвует в расчётах.**

Правила расстановки операторов:

* **Правило 0. Начальная расстановка операторов M**

Операторы M надо ставить перед всеми операторами K (нажатие клавиши), а также перед всеми операторами P (указание с помощью мыши), предназначенными для выбора команд (например, указание на выпадающий список); но перед операторами P, предназначенными для указания на аргументы этих команд (например, конкретный пункт в выпавшем списке), ставить оператор M не надо.

* **Правило 1. Удаление ожидаемых операторов M**

Если оператор, следующий за оператором M, является полностью ожидаемым с точки зрения оператора, предшествующего M, то этот оператор M может быть удален. Например, если вы перемещаете мышь чтобы нажать кнопку по достижении цели, то в соответствии с этим правилом следует удалить оператор M, устанавливаемый по правилу 0.

* **Правило 2. Удаление операторов M внутри когнитивных единиц**

Если строка вида M K M K M K… принадлежит когнитивной единице, то следует удалить все операторы M, кроме первого. Когнитивной единицей является непрерывная последовательность вводимых символов, например «4564.23» или «Константин Константинопольский».

* **Правило 3. Удаление операторов M перед последовательными разделителями**

Если оператор K означает разделитель, стоящий в конце когнитивной единицы (например, тире между двумя днями «понедельник — четверг»), то следует удалить оператор M, стоящий перед ним.

* **Правило 4. Удаление операторов M, которые являются прерывателями команд**

Если оператор K является разделителем, стоящим после постоянной строки (например, точка в конце предложения, которая каждый раз вводится в неизменном виде), то следует удалить оператор M, стоящий перед ним. Добавление разделителя станет привычным действием, и поэтому разделитель станет частью строки и не будет требовать специального оператора M. Но если оператор K является разделителем для строки аргументов или любой другой изменяемой строки, то оператор M следует сохранить перед ним.

* **Правило 5. Удаление перекрывающих операторов M**

Любую часть оператора M, которая перекрывает оператор R, означающий задержку, связанную с ожиданием ответакомпьютера, учитывать не следует.

**Что почитать:**

1. <https://habr.com/ru/articles/512712/>