Учреждение образования

«Белорусский государственный технологический университет»

**Кафедра информатики и веб-дизайна**

**Лабораторная работа №12**

Юзабилити-тестирование (ч.2)

Выполнил:

Студент 2 курса 1 группы ФИТ

Шкабров Данила Сергеевич

**2021 г.**

**Цель работы:**

1. Приобретение умений по проведению юзабилити-тестирования
2. Выбор методов тестирования
3. Приобретение практических навыков по построению тестовых сценариев, составления анкет и анализу полученных результатов

**Краткие теоретические сведения**

**Методики тестирования**

Для выявления проблем удобства использования приложения, в том числе на ранних этапах планирования и разработки ПО, используется методику двойной проверки:

• Изучение опыта взаимодействия пользователя с приложением **через имитацию поведения пользователей**;

• Проверка соответствия принципам обеспечения удобства пользования и корректного визуального представления в контексте функциональных требований посредством **экспертной оценки**.

**Экспертная оценка** приложения осуществляется в соответствии с целями проекта, функциональными и нефункциональными требованиями к ПО.

Процедуры экспертной оценки включают в себя:

* Выявление и изучение возможных сценариев использования и путей пользователя (User Journeys) в контексте бизнес-целей и функционала приложения.
* Анализ информационной архитектуры приложения.
* Анализ интерфейса и элементов интерфейса.
* Анализ функционального соответствия.

Крупный недостаток юзабилити-тестирования – высокая стоимость. Более быстрым и дешевым способом проверки качества интерфейса является экспертная оценка. Она позволяет обнаружить порядка 80% проблемных мест.

Эксперт (или несколько) проводят аудит системы.

Виды экспертной оценки:

• проверка по контрольному списку

• эвристическая оценка

• мысленная прогонка по интерфейсу.

*Проверка по контрольному списку* ближе всего к формальному тестированию качества.

Составляется список произвольных требований т. н. чек-листы (см. Прил. 1), после чего интерфейс проверяется на соответствие этим требованиям.

Четкий контрольный список может использоваться кем угодно, что дает возможность вынести проверку интерфейса из деятельности юзабилити-специалиста, передав ее отделу контроля качества.

*Эвристическая оценка* была разработана Якобом Нильсеном и Рольфом Моличем, которые надеялись с ее помощью сократить продолжительность проведения проверки по контрольному списку. При эвристической оценке вместо десятков и сотен конкретных требований интерфейс проверяется на соответствие всего нескольким общим принципам.

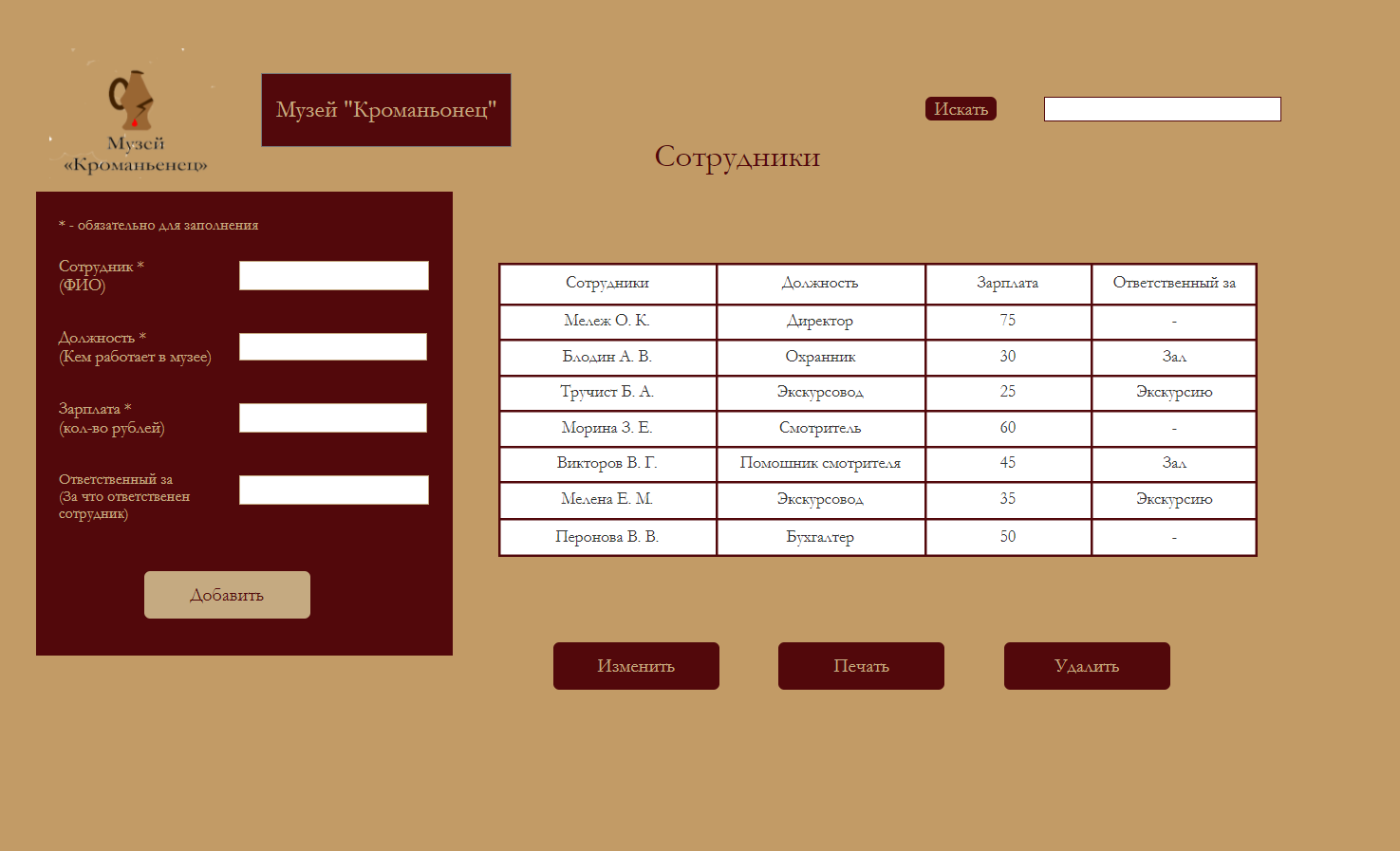
*Мысленная прогонка* формализует метод, по которому интерфейс оценивается. Если исходить из того, что интерфейс предназначен для использования функций, можно проверить, как эти функции вызываются и используются. Если просто проговорить словами, как работают интерфейсы всех функций, становится понятно, какие из них неоправданно подавлены, а какие работают недостаточно хорошо. Конечно, для этого тоже необходим опыт эксперта.

Проведем **экспертное тестирование** проекта с целью анализа информационной структуры приложения, интерфейса, а также функционального соответствия.

**Требования к конкретным элементам управления**

**Кнопки**

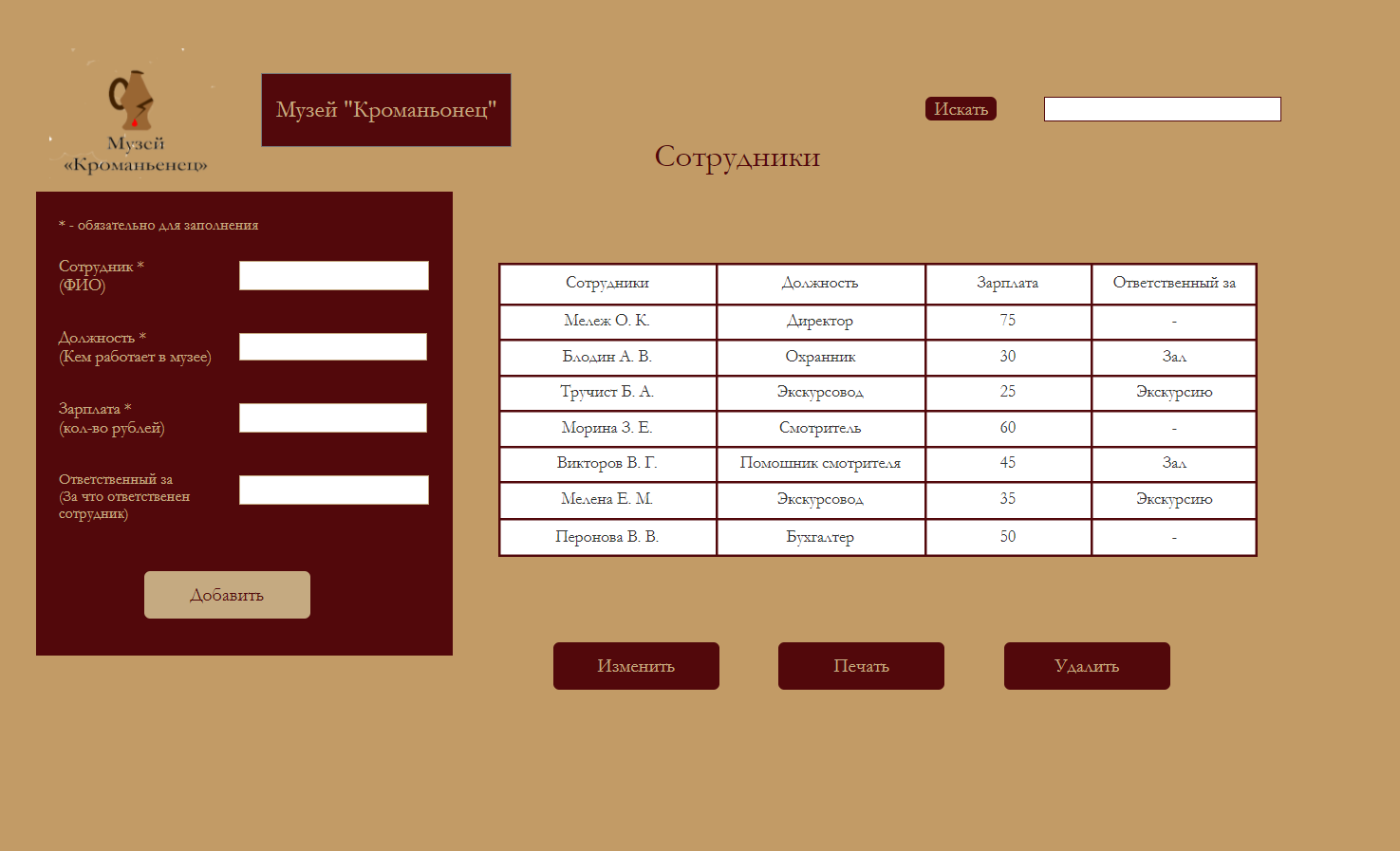
* Все кнопки, запускающие действия, имеют текст в инфинитивной форме глагола (пример: искать), а не другую часть речи либо форму глагола (пример: готово). Давать кнопке текст «ОК» можно, только если какой-либо глагол не вмещается. (Удобно использовать инфинитив, так как это самая краткая форма глагола).



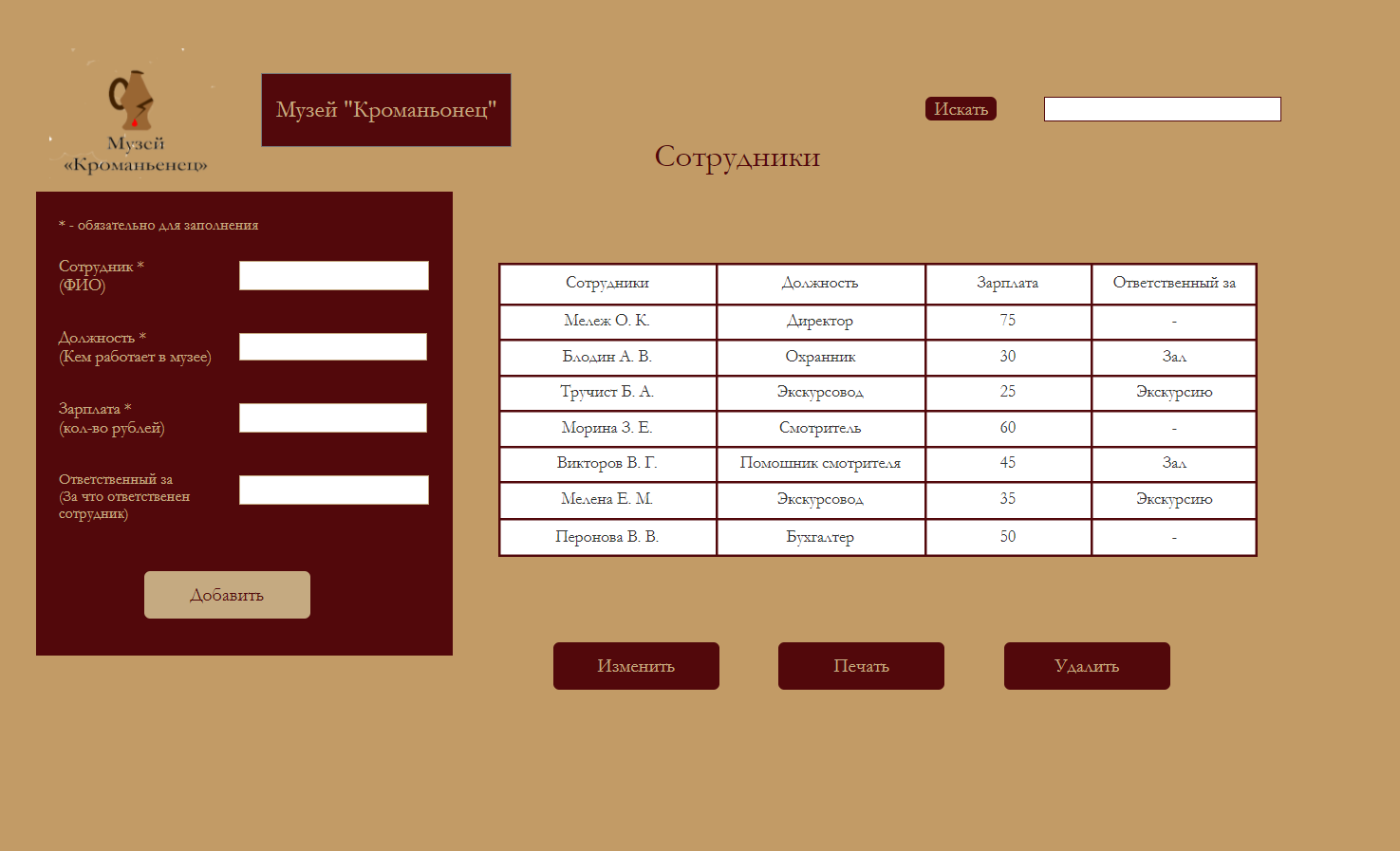
* Кликабельный размер кнопок совпадает с их видимым или логическим размером. (Чтобы показать кнопку, более важную, например в главной странице кнопки страниц больше, чем для цветной слепоты).



* Между кнопками, стоящими рядом, должно быть пустое пространство, щелчок по которому не отрабатывается. (Чтобы пользователь при переносе не нажал на не нужную кнопку возле).



* Нет разных состояний кнопок, которые выглядят одинаково. (Двух одинаковых кнопок нет с разными состояниями, по размерам совпадают, но они разделены пространством между собой и подписаны).



**Поля ввода**

* В полях ввода уже стоят наиболее вероятные значения. (Для удобства и быстрого заполнения поля информацией).



* Длина полей не меньше, и, по возможности, не больше, длины вводимых в них данных (поля специально рассчитаны с запасом для вводимых данных).



* Если поле предназначено для ввода заметного количества текста, оно многострочное. (поле для общей информации больше (в длину) для отображения большего текста).

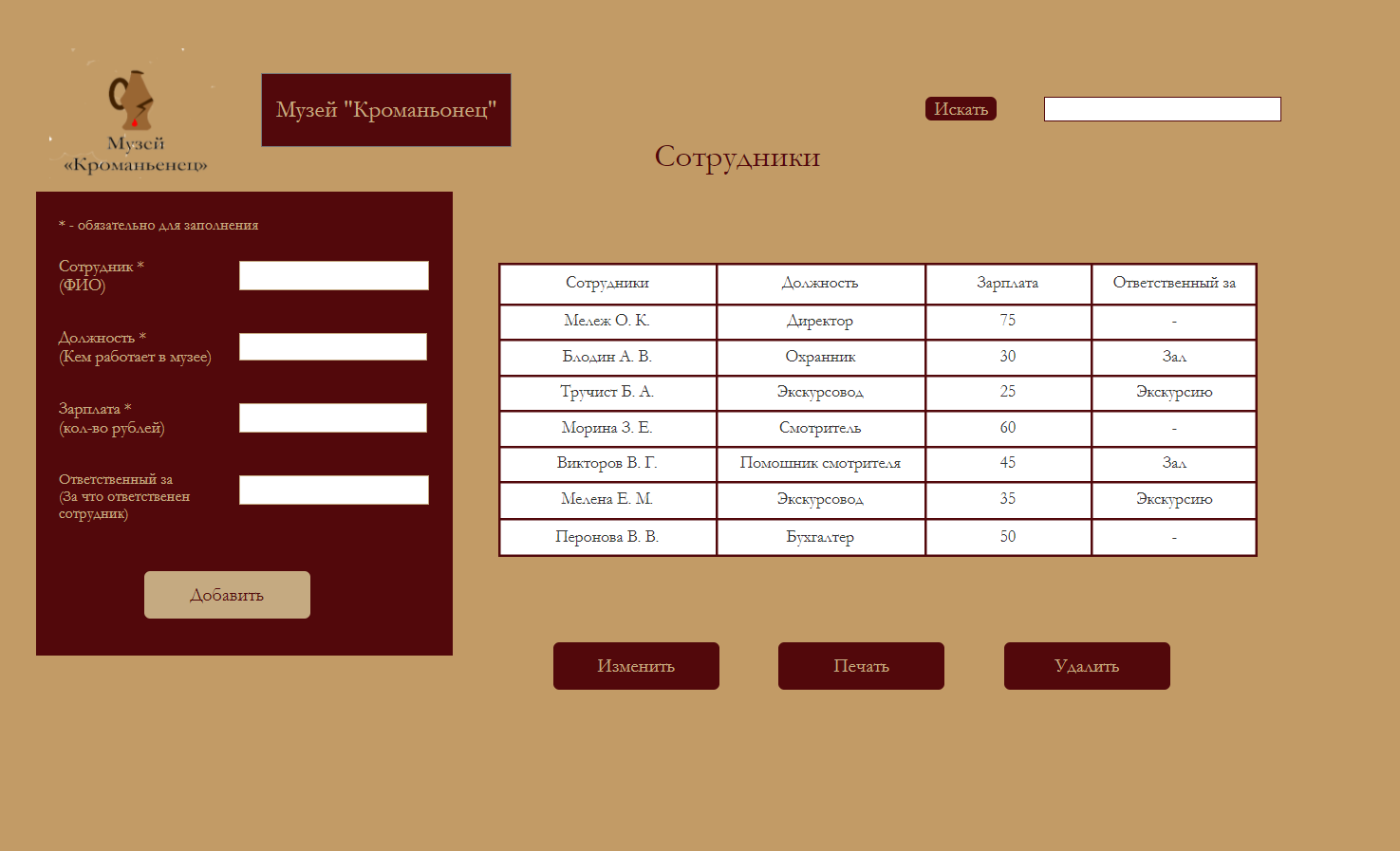


* Многострочные поля имеют максимально возможную высоту; нет резервов для их увеличения. (Этого поля хватает для заполнения информации и не нужно его увеличивать).

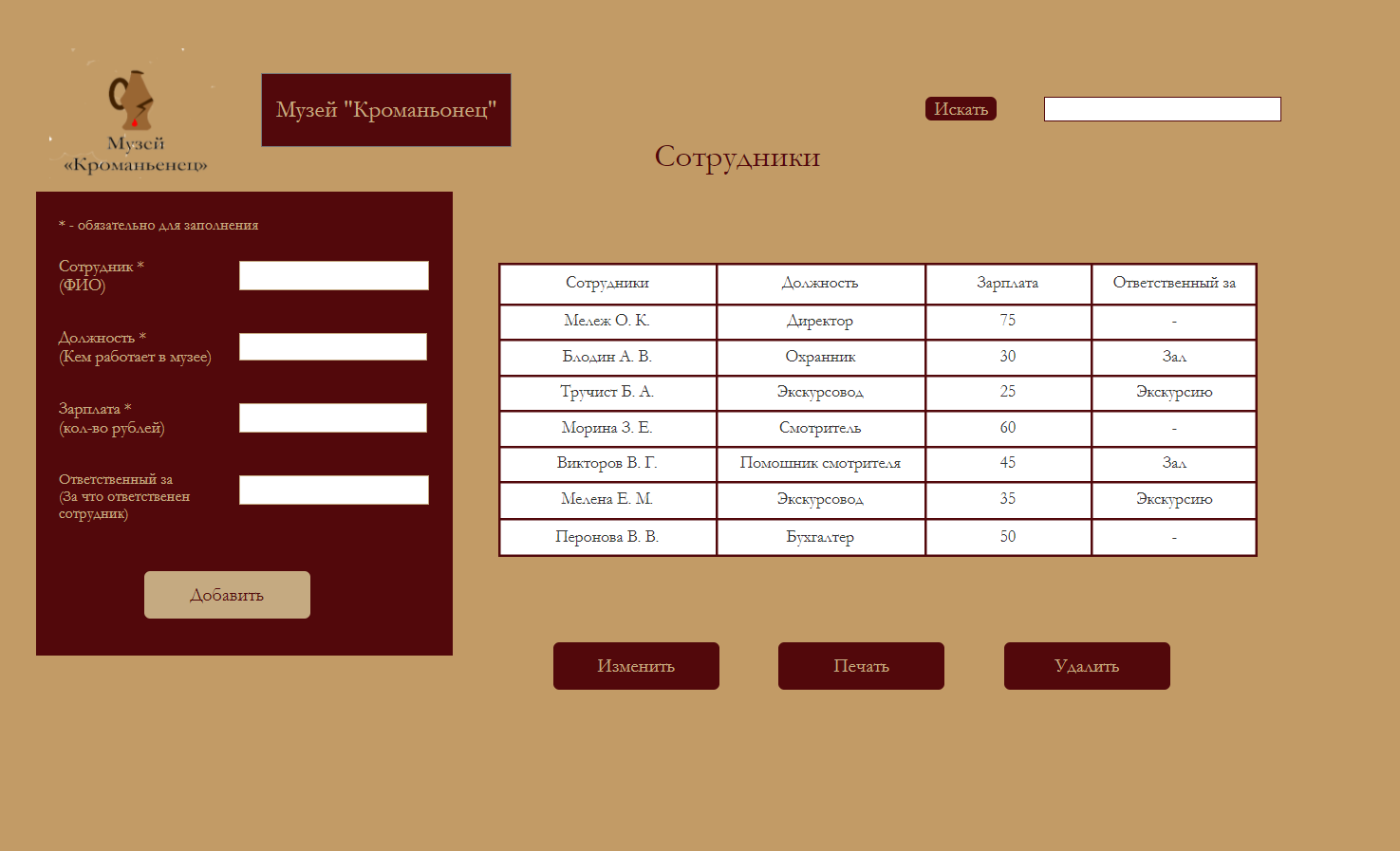


**Таблицы**

* Если таблица содержит более 50 элементов, используется фильтр или режим поиска. (для более быстрого и удобного поиска нужной информации).

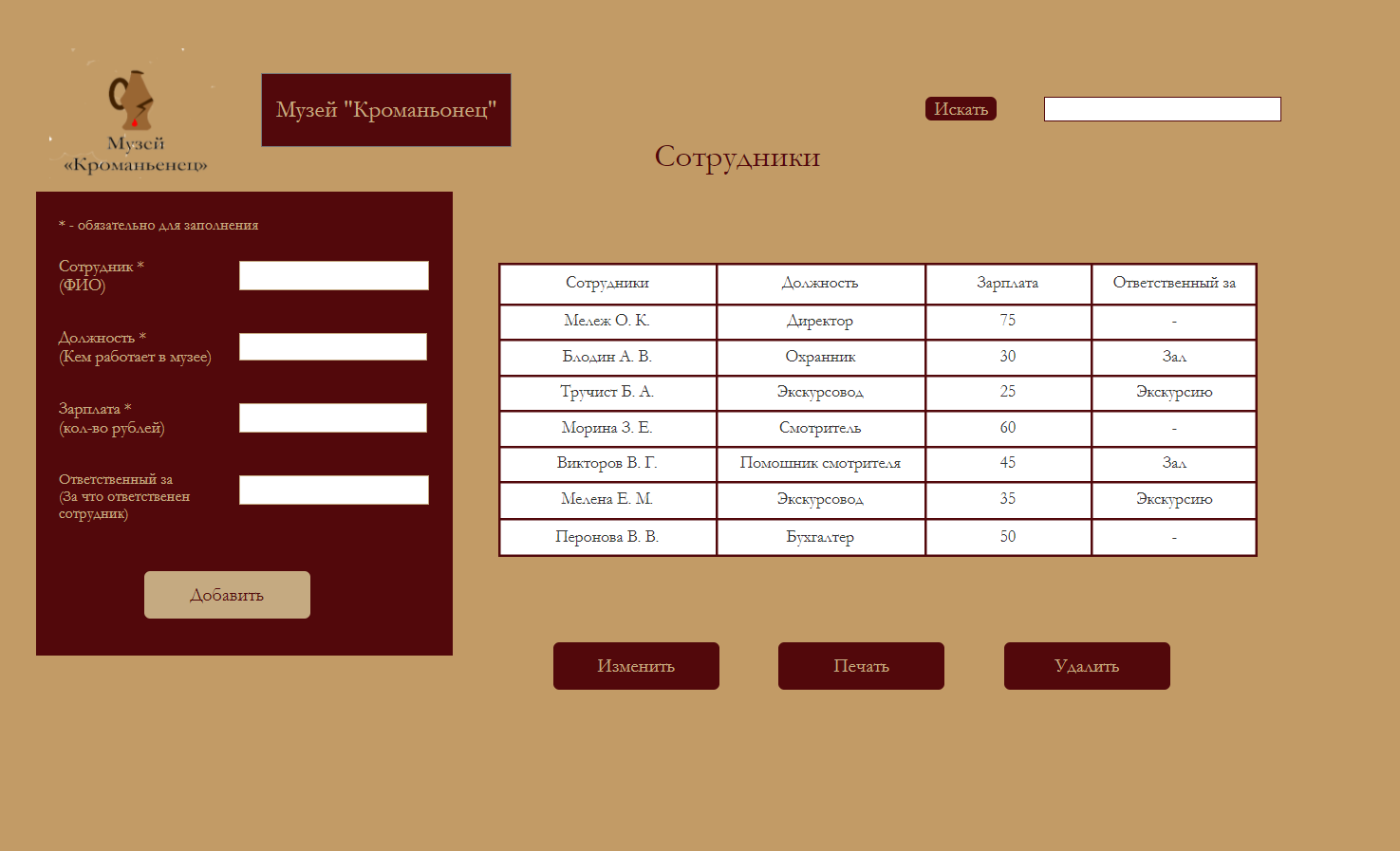


* Нет часто используемых коротких таблиц (менее пяти элементов). (используется одна простая таблица, со всеми подробностями).



* Ширина списков не меньше ширины входящих в них элементов.

(Ширина списков специально рассчитывалась под вводящийся в него максимум).



**Системные сообщения и отработка ошибок**

* В формах ввода проверка корректности вводимых значений выполняется прямо во время ввода; если вводятся некорректные данные, система сразу сообщает об этом пользователю, не дожидаясь момента, когда пользователь завершит ввод данных во всей форме. (Чтобы пользователь сразу же увидел, что он вводит не корректно информацию).



* Сообщения о некорректности введенных данных показываются рядом с элементом управления, данные в котором некорректны. (Для того, чтобы пользователь узнал конкретную проблему).



* Если окно с текстом о некорректности введенных данных всплывающее — оно не должно заслонять собой поле ввода, где содержится текст с ошибкой. (Пользователь должен видеть поле и информацию о некорректности чтобы он мог разобраться что он сделал не так и исправить свою ошибку).



**Меню**

* Первая буква в названии пунктов меню - заглавная. (Сайт должен показывать граммотность).



* Все пункты меню первого уровня активизируют раскрывающиеся меню. (Чтобы было проще переходить в подпункты).

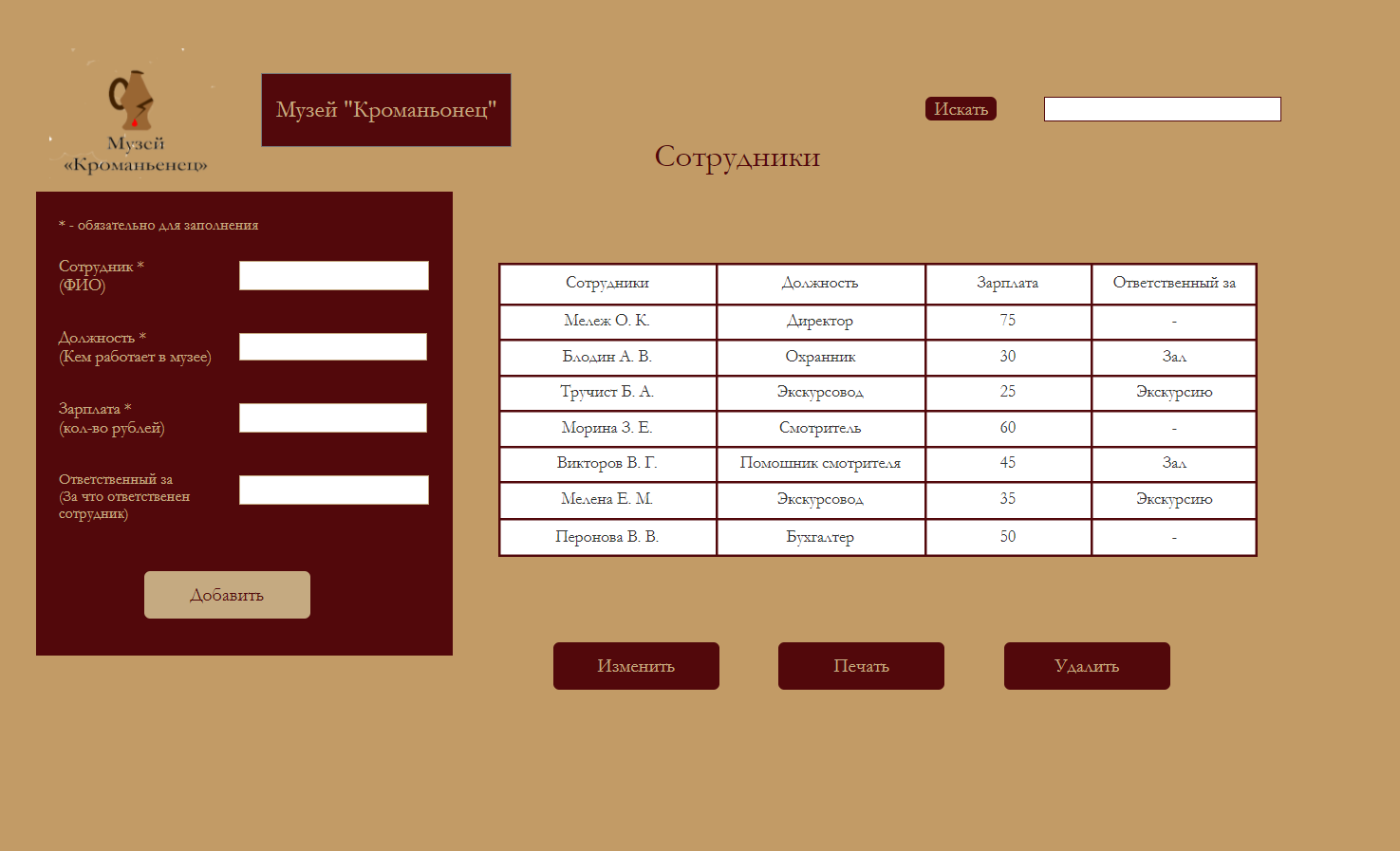


* Используются не более двух подуровней меню. (Слишком подробное оглавление ненужно).

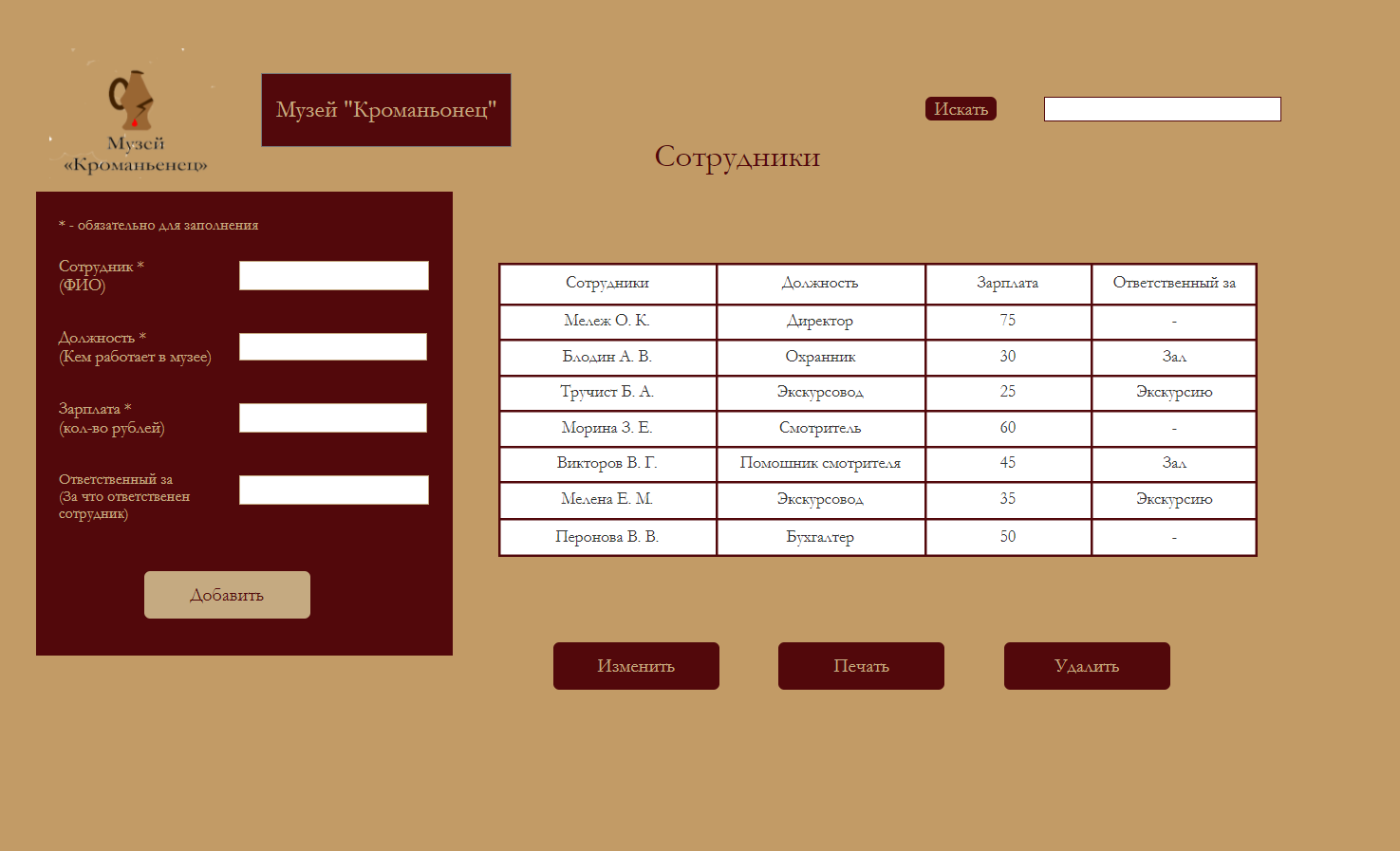


**Формы ввода**

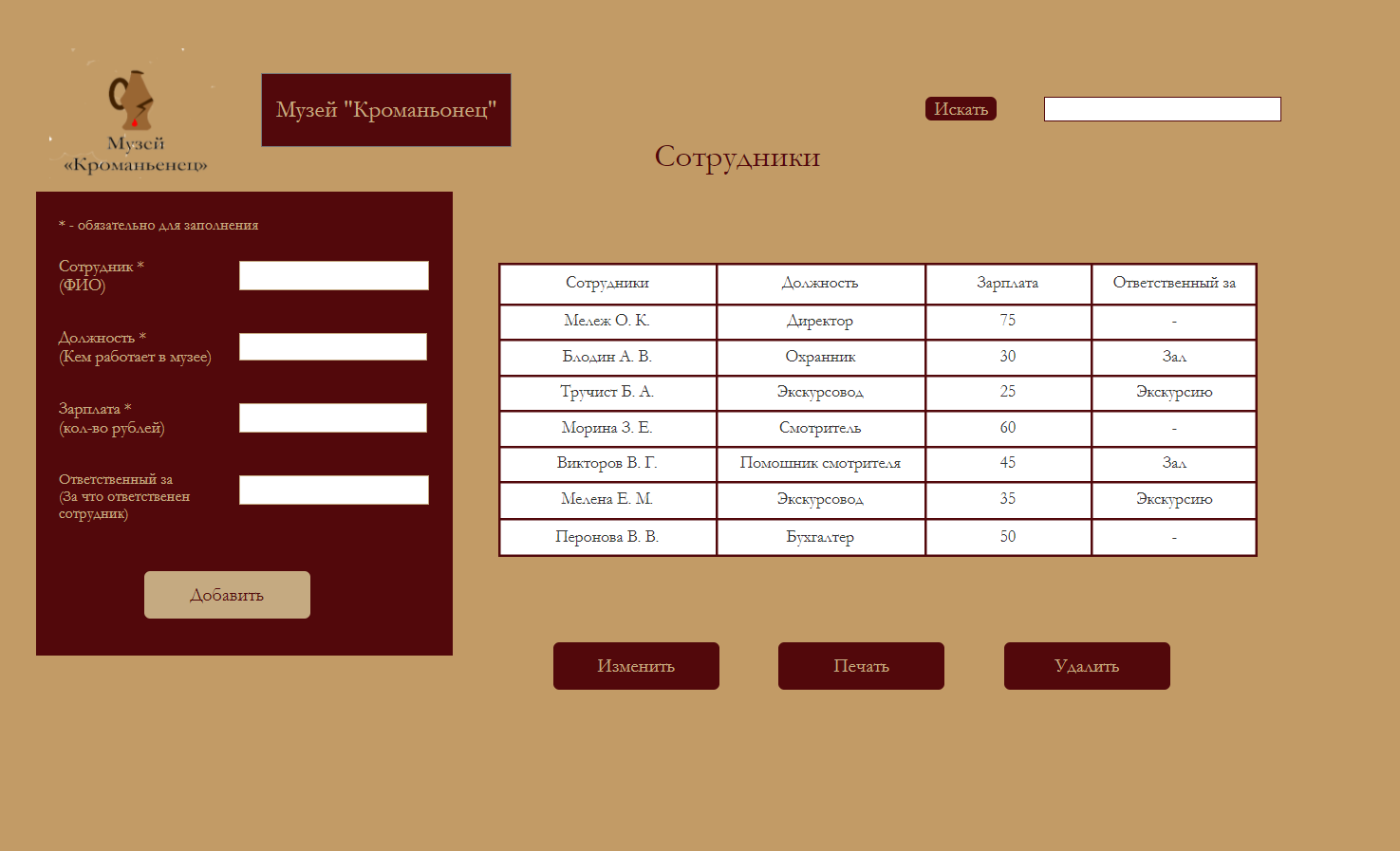
* У формы ввода есть название. (Пользователь должен знать для чего это форма).



* Все поля, обязательные для заполнения, помечены, и есть соответствующее пояснение. (Чтобы пользователь знал, что ему точно нужно заполнить).

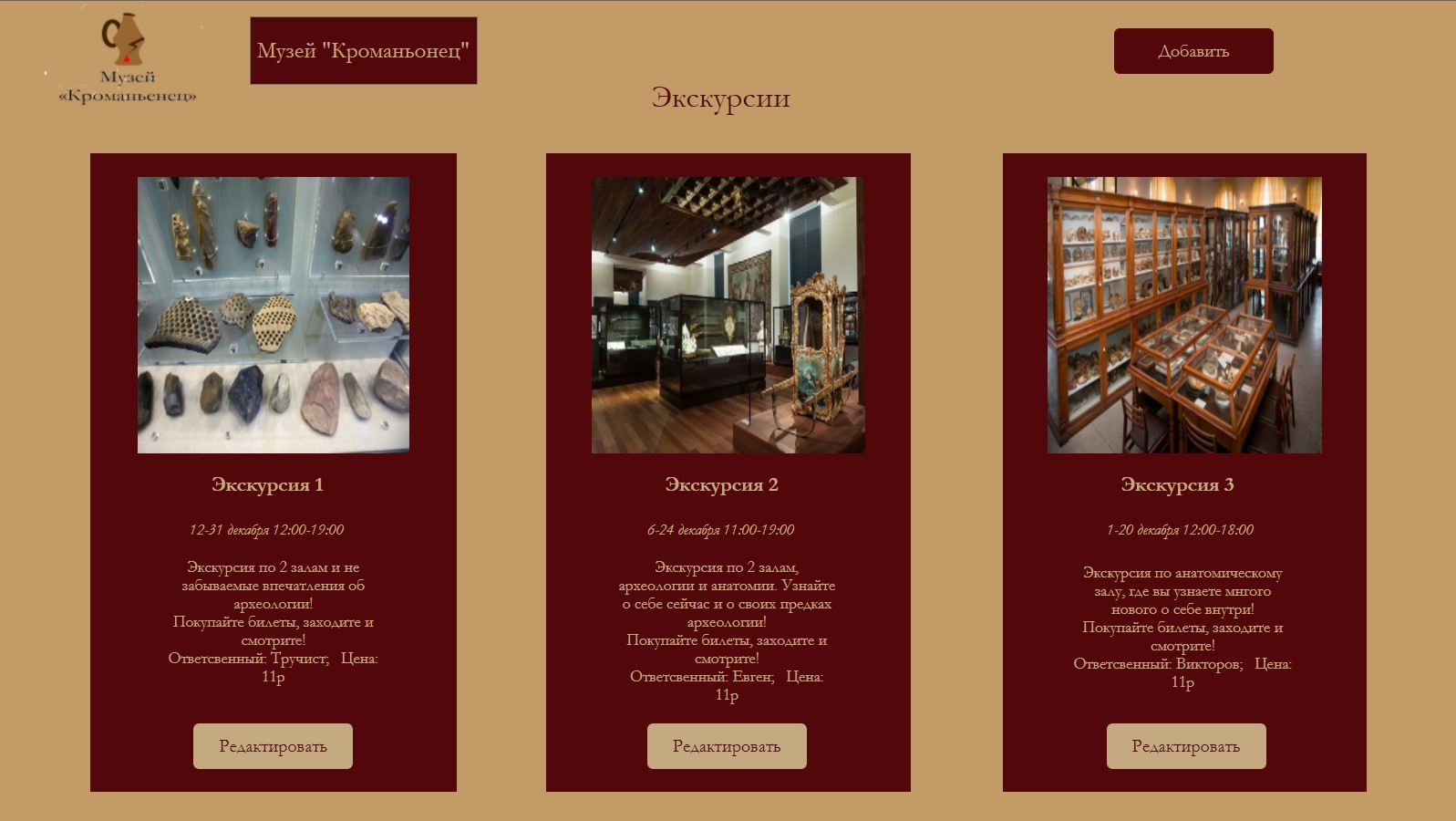


* Во всех формах, служащих для сбора информации, есть описание целей сбора данных, объясняется, что с этими данными будет сделано и что не будет. (Для пояснения того, что нужно вводить).

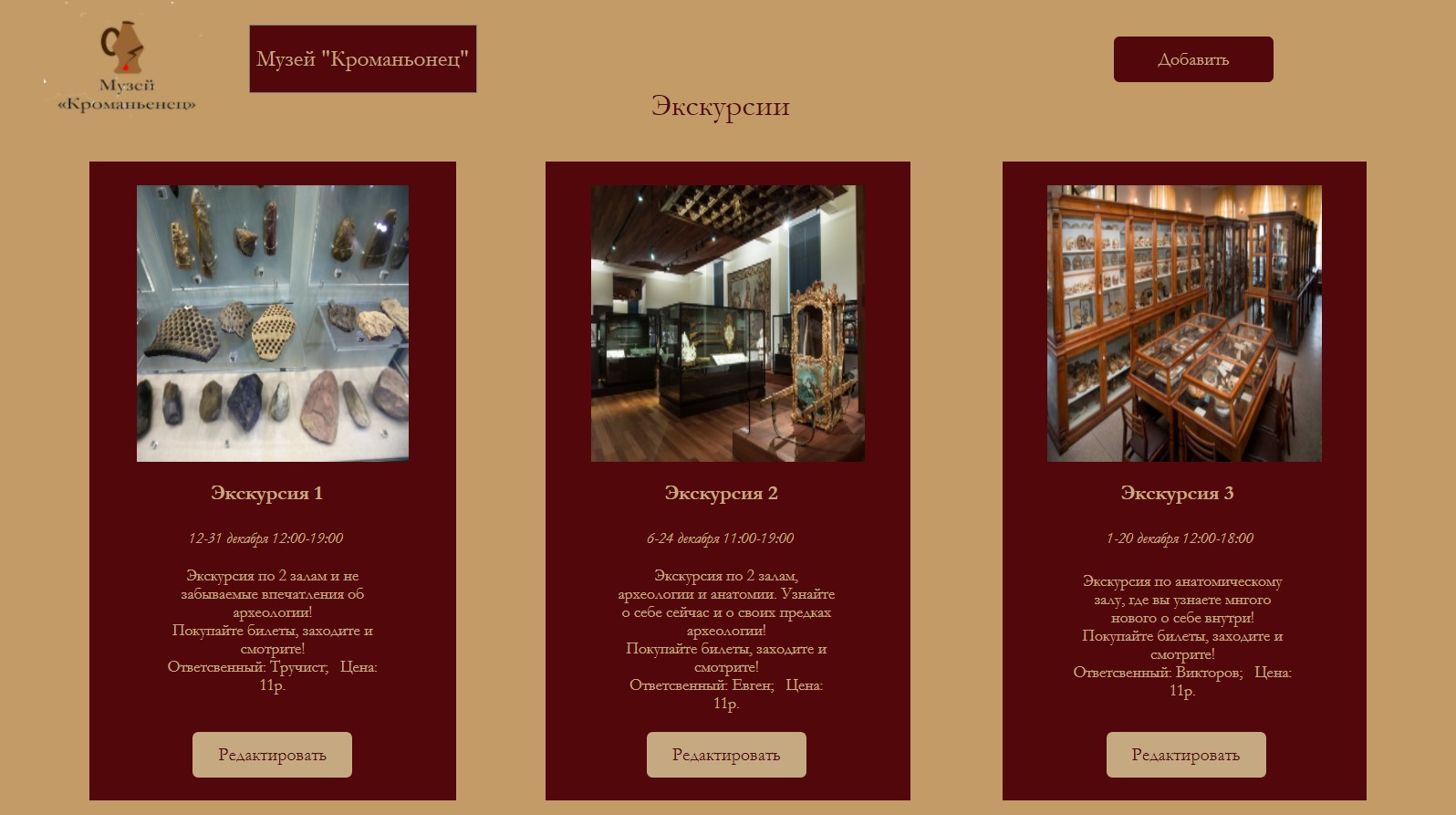


**Текст**

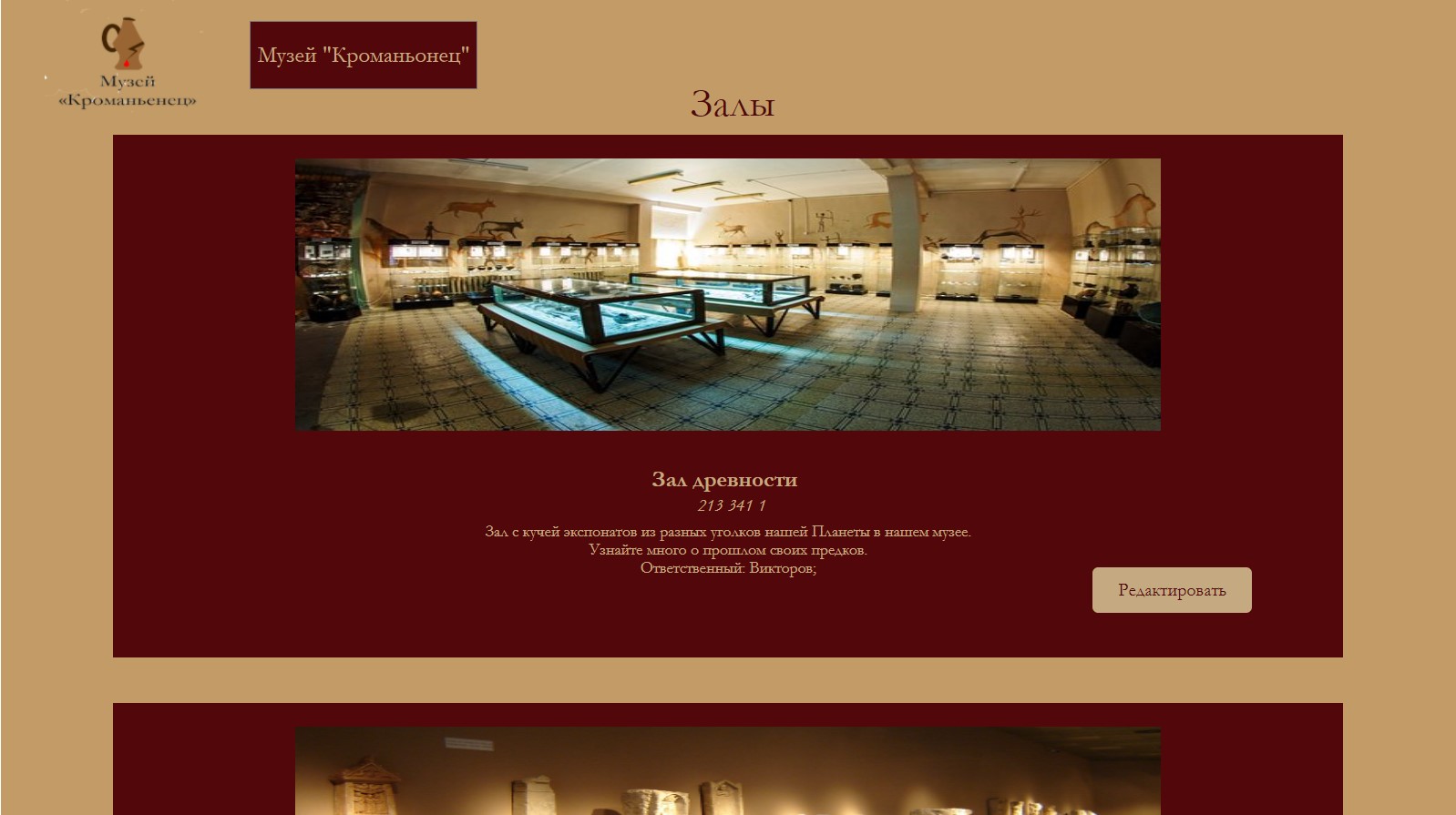
* В интерфейсе отсутствуют жаргонизмы. (Здесь не нужны жаргонизмы, это сайт для музея, где правильность языка подчеркивает строгость музея).



* Каждый элемент списка содержит на конце точку или начинается с прописной буквы по след. правилу: «Текст всех элементов начинается со строчной буквы. Все элементы оканчиваются по последней букве слова без каких-либо знаков препинания, кроме последнего, который оканчивается точкой. Исключение: если хоть один элемент списка содержит более одного предложения, все элементы начинаются с заглавной буквы и заканчиваются точкой.». (В обычном тексте для показа грамотности нужно писать по правилам).

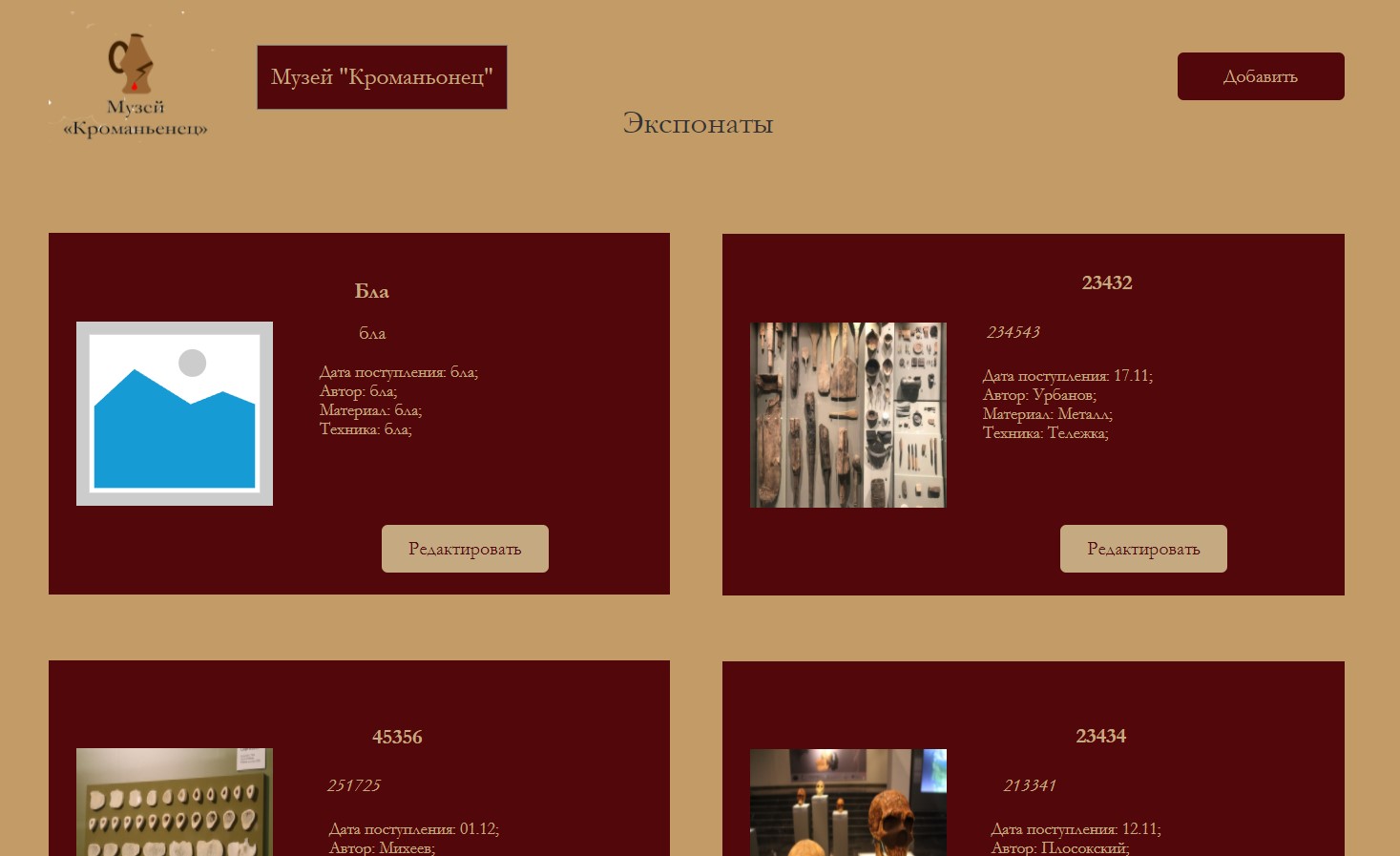
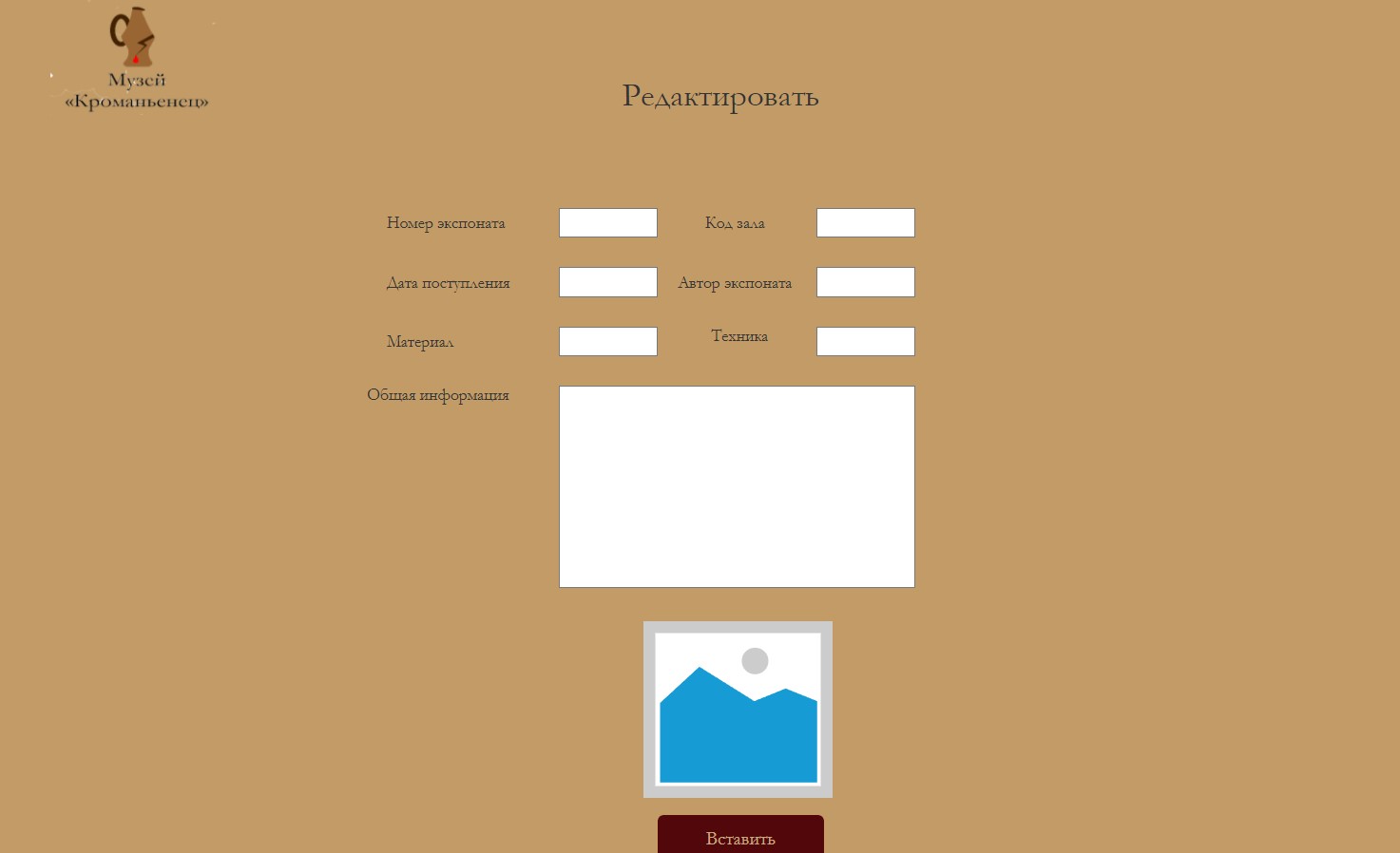
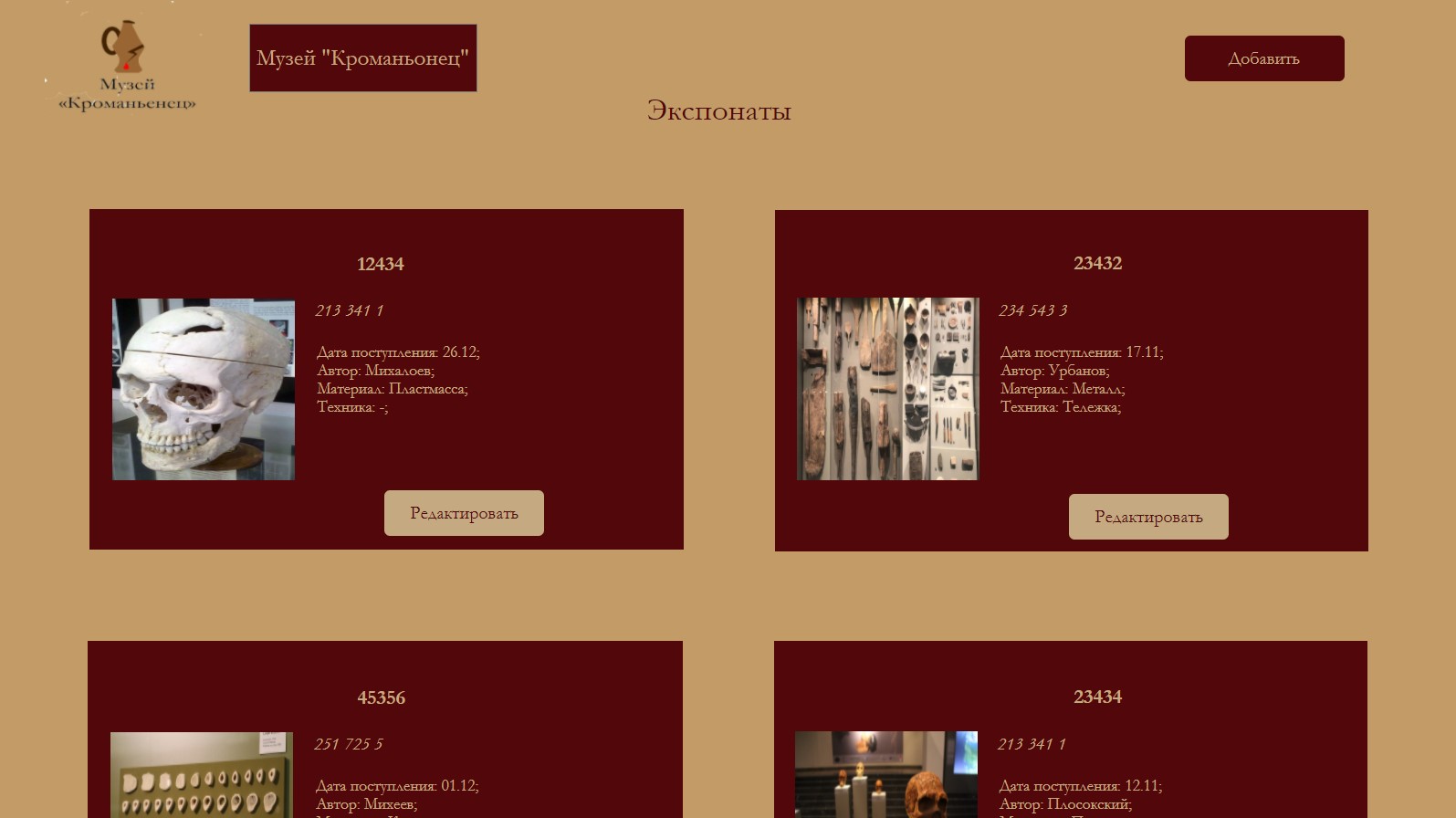


* Для улучшения удобочитаемости длинные числа разбиваются неразрывным пробелом по три цифры: 1 234 567. (Цифры проще называть тройками или двойками, чем сразу 1 число сразу).



**Эвристическая оценка Якоба Нильсена и Рольфа Молича:**

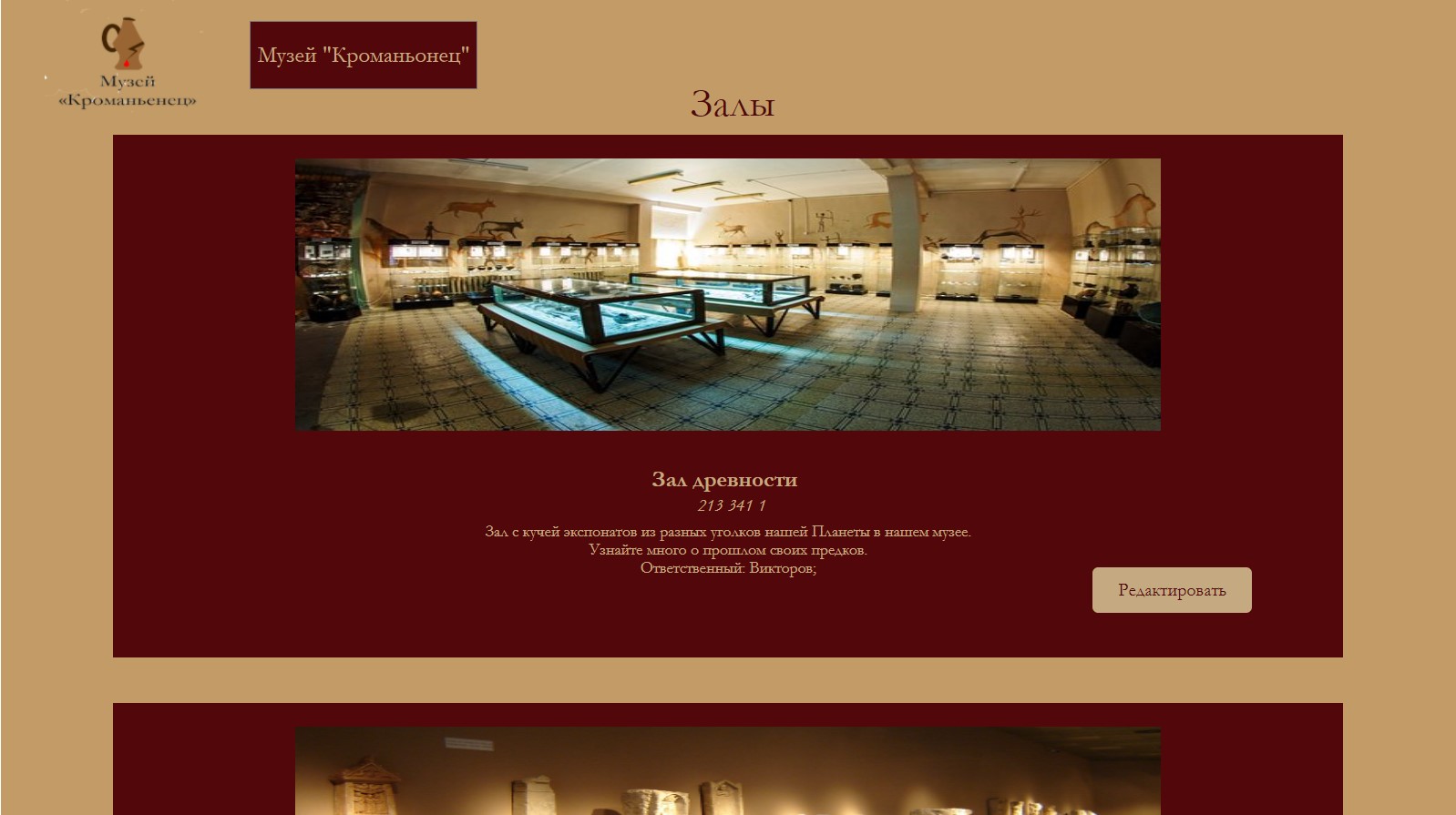
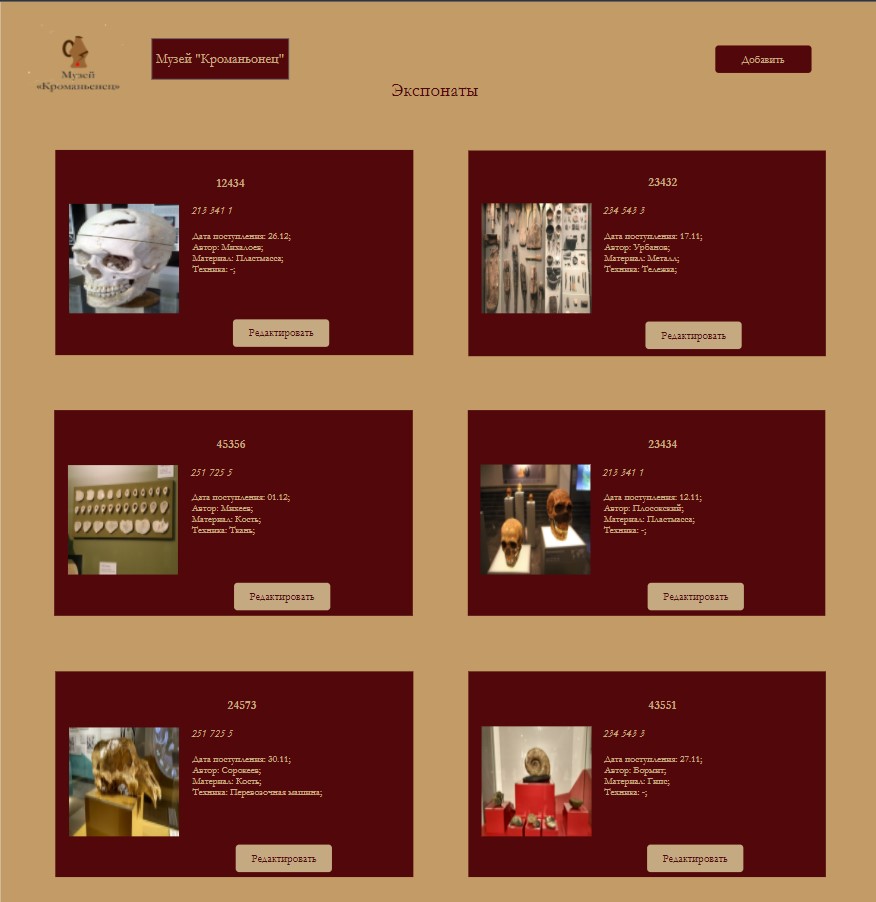
1. В любой момент времени система показывает, что с ней происходит. Да



1. Система использует термины, понятия и метафоры, присутствующие в реальном мире, а не обусловленные компьютером. Да



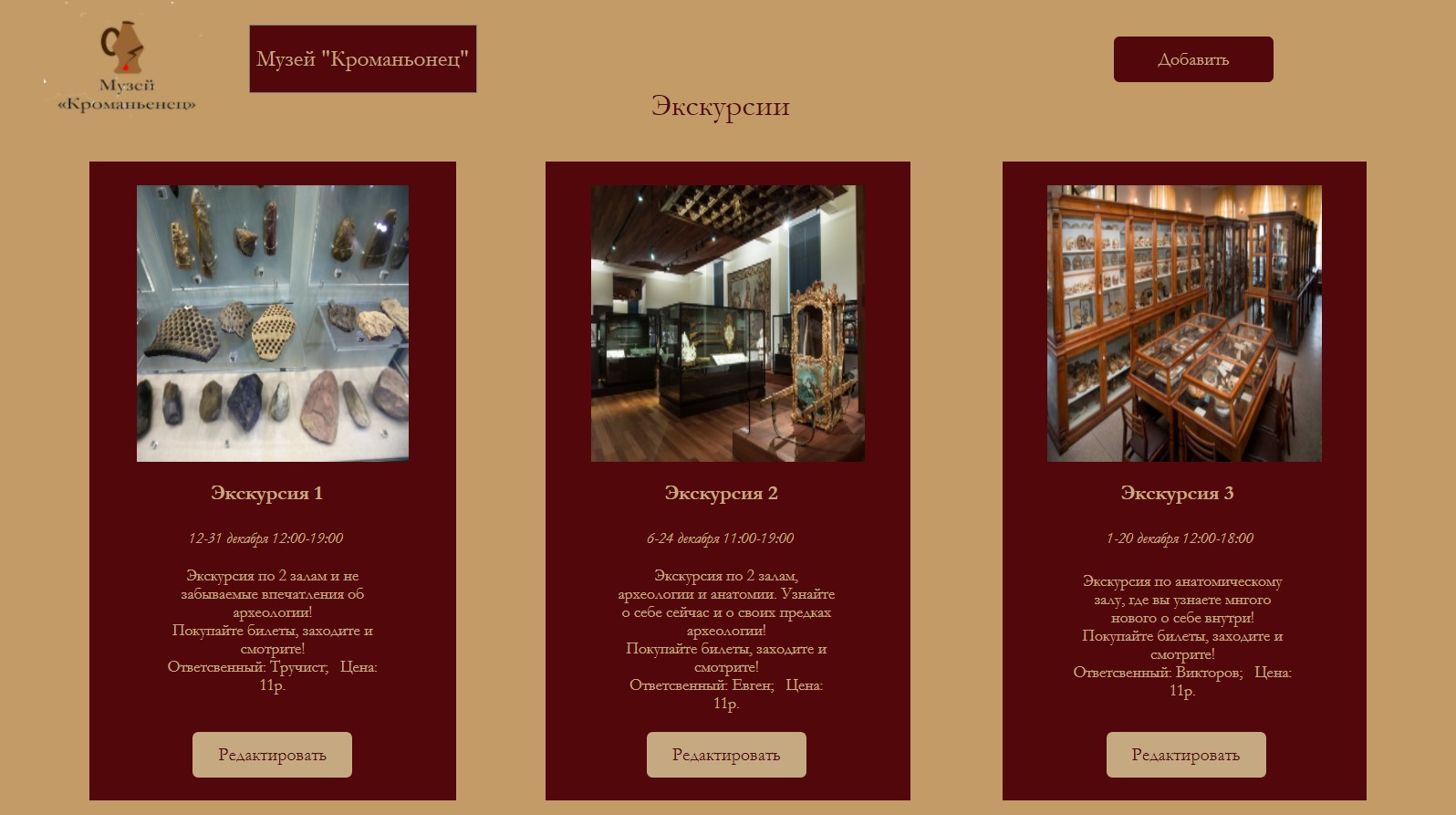
1. В любой момент пользователь контролирует систему, а не наоборот. Любую команду можно отменить или повторить. Нет
2. В любой момент времени система выглядит и функционирует единообразным и стандартным способом. Да



1. Интерфейс системы препятствует появлению человеческих ошибок. Да



1. В любой момент времени интерфейс показывает объекты и команды сам, не требуя от пользователя вспоминать их. Нет
2. В интерфейсе есть методы ускорения работы, предназначенные для опытных пользователей и не мешающие пользователям неопытным. Нет
3. Интерфейс эстетичен и в любой момент времени не содержит ненужной сейчас информации. Да



1. Интерфейс помогает пользователям обнаруживать и исправлять проблемы, включая человеческие ошибки. Нет
2. Справка доступна в любой момент времени. Она достаточна, но не избыточна; к ней легко обращаться; она не абстрактна, а нацелена на решение конкретных задач пользователя; в ней описываются конкретные шаги по решению проблем. Нет

**Выявленные проблемы интерфейсов**

По результатам тестирования нарушений в данном интерфейсе выявлено не было, проблем в информационной структуре или функциональном соответствии элементов обнаружено не было. Таким образом, можно сказать, что интерфейс спроектирован грамотно и при его разработке учитывались все принципы юзабилити.

**Предложения по модификации интерфейса**

Предложений по модификации интерфейса нет, так как сайт прост и работает с пользователями (выявление ошибок, есть подсказки и т. д.) и удовлетворяет минимуму оценкам и требованиям. На этом все.

**Вывод:** в ходе лабораторной работыя приобрёл умения по проведению экспертного тестирования, изучил и осуществил выбор методов тестирования, а также приобрёл практические навыки по построению чек-листов и контрольных списков с самыми важными пунктами интерфейса.