**Тестовое задание**

Создать два базовых класса: первый - описывающий сотрудника с почасовой оплатой, второй - сотрудника с месячной ставкой. Вычисление оплаты труда для сотрудников с почасовой оплатой будет иметь следующую формулу: количество часов \* сумму за один час.  Для сотрудников со ставкой метод расчета возвращает месячную ставку. Реализовать следующую функциональность:

1. Выборка сотрудников из xml файла (формат продумать самостоятельно)
2. Выборка сотрудников из БД (Спроектировать самостоятельно)
3. Вывести список сотрудников со следующей информацией: ФИО, дата рождения, отдел, должность, тип сотрудника (ставка или почасовая оплата), оплата за месяц (рассчитывается по формуле в зависимости от типа сотрудника).
4. Реализовать возможность вывода (20, 40, 50, 100) сотрудников на страницу с возможностью выбора к показу количества сотрудников. (По умолчанию 20).
5. Добавить возможность пагинации по сотрудникам. (Пользователь перешел на страницу с номер 15 это означает, что все страницы пагинации показываться не должны, а только несколько с левой и справой стороны, к примеру, по 5 ссылок, это означает, что если пользователь находится на первой странице, то отображаем 10 ссылок справой стороны и т.д.)
6. Добавить навигацию по отделам, переход по отделу ведет на страницу вывода сотрудников выбранного отдела.

**Принцип формирования URL:**

Возможные варианты URL для ЧПУ относительно домена:

/employes/ - Вывод списка сотрудников

/employes/3/ - третья страница пагинации списка сотрудников

/employes/[department]/ - Вывод списка сотрудников по отделу

/employes/[department]/3/ - Вывод списка сотрудников по отделу третья страница пагинации

Где [department] – это идентификатор отдела

Такие данные как количество вывода сотрудников на страницу можно вынести в GET-параметр.

**Будет плюсом**

1. Выполнение действий пользователя без перезагрузки страницы.
2. Реализация страницы для отображения 404 http статуса.
3. Покрытие кода unit-тестами

**Условия выполнения**

1. Без использования различных фреймверков и CMS и дополнительных библиотек. Без использования различных css и js библиотек, в том числе без JQuery.
2. Механизм работы должен быть построен на основании ЧПУ, например для фильтрации по отделам и должностям достаточно указать url следующего формата(your domain.com/department/position/)
3. Для реализации механизма ЧПУ запрещено создавать множество пакетов.
4. Система должна быть защищена от стороннего вмешательства с помощью SQL инъекций и применены другие меры безопасности на уровне php.

**Что ожидается**

1. Чистота кода и ее читабельность.
2. Соответствие стандартам кодирования.
3. Обоснованное применение принципов ООП.
4. Использование только обоснованных паттернов. Избыточность или появление лишнего кода не допустимо.
5. Проектирование базы данных. (Нормализация)
6. Валидный HTML5 по семантическим нормам.

**Этапы выполнения тестового задания**

1. Прочитать и вникнуть в ТЗ
2. Спроектировать базу данных (Нормализировать).
3. Создать файл script.sql для создания базы данных и работы с ней. Должны находится все необходимые SQL запросы для создания всех необходимых таблиц.  Если есть исходные первоначальные данные, без которых система будет вести себя не корректно или запрашивать их добавление так же они должны присутствовать в виде SQL запросов в данном файле. К таким данным не относятся записи сотрудников в БД. Система может корректно работать как с 10-тью сотрудниками, так и в случае если таблица будет пустой.
4. Проектирование иерархии классов и интерфейсов.
5. Заложение бизнес-логики
6. Верстка и frontend, (Верстка как таковая оцениваться не будет, главное не отступать от корректного использования DOM)
7. Тестирование и отладка

**Прием задания**

1. Схема базы данных
2. Файл script.sql
3. Исходный код
4. **Ссылка на работающий пример** (разместить выполненное задание на бесплатном хостинге, например, timeweb.ru–выбираете любой тариф, регистрируетесь и получаете 10 дней бесплатного теста)
5. Какая часть функциональности была выполнена?
6. При сдаче задания указать количество часов потраченного на выполнение