Частное учреждение образования
«Колледж бизнеса и права»

ОТЧЕТ
ПО ПРЕДДИПЛОМНОЙ ПРАКТИКЕ

ОП Т.817022.401

Руководитель практики
от предприятия (Д.Д. Напрушкин)

Руководитель практики
от колледжа (И.М. Рагунович)

Учащийся (Р.Ю. Подольсикй)

2021

Содержание

Введение

Задачей преддипломной практики является разработка веб-приложения по автоматизации отдела кадров СТО, которое облегчит просмотр и редоктирование списка сотрудников.

Данное приложение будет служить для решения следующих задач: авторизация, просмотр данных, редактирование данных, мобильную версию, выдачу справок.

Для достижения цели преддипломной практики, необходимо решить следующие задачи:

– выполнить объектно-ориентированный анализ и проектирование системы, результатом которой будет модель системы;

– определить вычислительную систему, необходимую для создания программного средства;

– по модели выполнить проектирование задачи;

– разработать программное средство;

– описать созданное программное средство;

– выбрать методику испытаний;

– описать процесс тестирования;

– привести примеры области применения.

Решение поставленных задач отражено в пояснительной записке, которая состоит из шести разделов и содержит необходимую и достаточную информацию по организации и использованию данного программного средства.

В первом разделе «Объектно-ориентированный анализ и проектирование системы» раскрывается организационная сущность задачи, описывается предметная область и круг задач, которые должны быть автоматизированы. Описывается задача, перечисляются основные функции программы. Строится информационная модель, отражающая сущности задачи, их свойства и взаимосвязи. Описываются новые возможности программы, а также ее отличия от предыдущих версий.

Во втором разделе «Вычислительная система» перечисляются требования к аппаратному обеспечению и конфигурации компьютера, приводится характеристика операционной системы, выбор и обоснование среды разработки приложения.

В третьем разделе «Проектирование задачи» проводится объектно-ориентированный анализ задачи, описываются требования к приложению, концептуальный прототип, организация данных, функции и элементы управления, проводится проектирование справочной системы приложения.

В четвертом разделе «Описание программного средства» представлены общие сведения о программном средстве и его функциональном назначении, описываются входные и выходные данные.

В пятом разделе «Методика испытаний» описываются требования к техническим средствам для проведения испытаний, требования к характеристикам программы применительно к условиям эксплуатации, требования к информационной и программной совместимости. Предоставляются результаты функционального тестирования.

Шестой раздел «Применение» предназначен для описания сведений о назначении программного средства и области его применения. В этом разделе приводится структура справочной системы, а также методика ее использования.

В заключении описывается выполнение поставленной задачи, степень соответствия проектных решений задания, причины несоответствия, если таковые имеются.

В приложении А содержится текст программы.

Графическая часть представлена пятью диаграммами вариантов использования, классов, последовательности, деятельности и компонентов.

1 Объектно-ориентированный анализ и проектирование системы

1.1 Сущность задачи

Предметной областью решаемой задачи является автоматизация отдела кадров.

Исходя из предметной области при тестировании процедур/функций Oracle будут достигнуты следующие цели:

- авторизация;
- просмотр данных;
- редактирование данных;
- выдача справок.

Главная цель применения программного средства – обеспечение удобства в работе с отделом кадров.

Аналоги данной программы предоставляют возможность просматривать и редактировать определённый отдел кадров и не дают неизвестным пользователям возможности изменения данных.

Целью разработанной программы является создание такого веб-приложения, которое будет влключать в себя функции представленных программ и дать возможность пользователям данной программы просматривать и редактировать отдел кадров.

1.2 Проектирование модели

Цель моделирования данных состоит в обеспечении разработчика информационной системы концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных.

В рамках унифицированного языка моделирования (UML) все представления о модели сложной системы фиксируются в виде специальных графических конструкций – диаграмм. В терминах языка UML определены следующие виды диаграмм: диаграмма вариантов использования, диаграмма классов, диаграмма деятельности, диаграмма последовательности, диаграмма компонентов.

Суть диаграммы вариантов использования состоит в том, что проектируемая система представляется в виде множества сущностей или актёров, взаимодействующих с системой с помощью, так называемых, вариантов использования.

Данная программа имеет следующие основные функции:

- авторизация;
- просмотр данных;
- редактирование данных.

К вспомогательным функциям, расширяющим возможности системы, относятся следующий функции:

- мобильная версия;
- выдача справок.

Диаграмма вариантов использования представлена в графической части на листе 1.

Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать, в частности, различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений. На данной диаграмме не указывается информация о временных аспектах функционирования системы. С этой точки зрения диаграмма классов является дальнейшим развитием концептуальной модели проектируемой системы.

Диаграмма классов для проектируемой системы представлена в графической части на листе 2.

При моделировании поведения проектируемой или анализируемой системы возникает необходимость детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Для моделирования процесса выполнения операций в языке UML используются так называемые диаграммы деятельности. Каждое состояние на диаграмме деятельности соответствует выполнению некоторой элементарной операции, переход в следующее состояние срабатывает только при завершении этой операции. Графически диаграмма деятельности представляется в форме графа, вершинами которого являются состояния действия, а дугами - переходы от одного состояния действия к другому.

Основная цель использования диаграмм деятельности - визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения.

Диаграмма деятельности для функции добавления информации о новом клиенте представлена в графической части на листе 3.

Для моделирования взаимодействия объектов в UML используются соответствующие диаграммы взаимодействия. Если рассматривать взаимодействия объектов во времени, тогда для представления временных особенностей передачи и приема сообщений между объектами используется диаграмма последовательности.

Временной аспект поведения имеет существенное значение при моделировании синхронных процессов, описывающих взаимодействия объектов. Именно для этой цели и используются диаграммы последовательности, в которых ключевым моментом является динамика взаимодействия объектов во времени. При этом диаграмма последовательности имеет как бы два измерения: одно - слева направо в виде вертикальных линий, каждая из которых изображает линию жизни отдельного объекта, участвующего во взаимодействии; второе - вертикальная временная ось, направленная сверху вниз, на которой начальному моменту времени соответствует самая верхняя часть диаграммы.

Диаграмма последовательности для функции составления ассортиментного перечня представлена в графической части на листе 4.

Рассмотренные ранее диаграммы отражали концептуальные аспекты построения модели системы и относились к логическому уровню представления. Особенность логического представления заключается в том, что оно оперирует понятиями, которые не имеют самостоятельного материального воплощения. Другими словами, различные элементы логического представления, такие как классы, ассоциации, состояния, сообщения, не существуют материально или физически. Они лишь отражают наше понимание структуры физической системы или аспекты ее поведения.

Основное назначение логического представления состоит в анализе структурных и функциональных отношений между элементами модели системы. Однако для создания конкретной физической системы необходимо, некоторым образом, реализовать все элементы логического представления в конкретные материальные сущности. Для описания таких реальных сущностей предназначен другой аспект модельного представления, а именно физическое представление модели.

Диаграмма компонентов описывает объекты реального мира – компоненты программного обеспечения. Эта диаграмма позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами.

Вид диаграммы компонентов для данной проектируемой системы представлен в графической части на листе 5.

2 Вычислительная система

2.1 Требования к аппаратным и операционным ресурсам

Конфигурация компьютера, на котором будет разрабатываться программное приложение, содержит следующие характеристики:

- процессор Intel(R) Pentium(R) CPU N3530 @ 2,16 ГГц 2,16 ГГц;
- оперативная память 8,00 Гбайт;
- видеокарта NVidia® GeForse M820;
- твердотельный накопитель 500 Гбайт.

Для работы с программой необходимо наличие браузера, клавиатуры, мыши и монитора.

2.2 Инструменты разработки

Инструментами разработки будут являться:

- операционная система Windows 10 Pro;
- среда программирования VS Code Insiders;
- язык программирования JavaScript;
- сервис для разработки диаграмм UMLet;
- программа для создания справочной системы Dr. Explain.

Программное приложение будет разрабатываться и тестироваться под управлением операционной системы Windows 10 Pro. Данная операционная система является оптимальным решением для предприятия любого размера. Данная версия операционной системы Windows сочетает в себе преимущества Windows eXtreme Programming (XP) Professional (например, средства безопасности, управляемость и надежность) с лучшими качествами WindowsVista. Это делает Windows 10 Pro наиболее подходящей операционной системой для настольных компьютеров, применяемых в корпоративной среде [15].

Исходя из результатов объектно-ориентированного анализа и проектирования (ООАП), итогов исследования предметной области, можно сделать заключение, что наиболее подходящей средой разработки программного приложения будет среда программирования VS Code Insiders [16].

Что касается языка программирования, то язык JavaScript используется в разработке приложений и браузерах с целью придания им интерактивности и «живости». Язык актуален в первую очередь потому, что область применения этого языка удивительно обширна и ничем не ограничена. Среди программ, которые используют JavaScript, присутствуют и тестовые редакторы, и приложения, и прикладное ПО [17].

UMLet – это надежная и удобная утилита для разработки моделей различного рода процессов и проектирования приложений на основе языка UML. С ее помощью вы сможете быстро строить UML диаграммы. Это инструмент рисования, а не инструмент моделирования, поскольку нет каталога повторно используемых объектов моделирования [12].

Dr.Explain – это приложение для быстрого создания файлов справки (help-файлов), справочных систем, on-line руководств пользователя, пособий и технической документации к программному обеспечению и техническим системам [10].

3 Проектирование модели

3.1 Требования к приложению

Разрабатываемое приложение должно иметь понятный и удобный в использовании интерфейс, чтобы взаимодействие между программой и пользователем было максимально упрощённым. Программное приложение должно обеспечить оперативный выбор необходимой информации по различным критериям, для этого будут разработаны процедуры обработки информации.

Данное приложение не будет требовать никаких специальных средств защиты, либо ограничений прав доступа к данным.

Все входные данные должны проверяться на ошибки. При совершении пользователям ошибки, ему будет предоставлено диалоговое окно с разъяснением ошибки.

Кроме этого, окна в приложении должны быть выполнены в едином стиле, сдержанной цветовой гамме, иметь стандартные элементы управления данными, не должны быть перегружены информацией, и в свое время понятны простому пользователю.

3.2 Концептуальный прототип

Концептуальный прототип состоит из описания внешнего пользовательского интерфейса, а именно, элементов управления.

При создании данного приложения важную роль играют страницы, так как они являются основным диалоговым средством работы пользователя. Разрабатываемое приложение будет содержать страницу авторизации, одну главную страницу.

При проектировании концептуального прототипа предполагается, что при запуске программы первой будет загружаться страница авторизации, макет которой представлен на рисунке 3.1.

На форме авторизации будут находиться такие компоненты как: текстовые поля и кнопки. После авторизации пользователь попадает на основную Страницу. Основная страница представления на рисунке 3.2.

3.3 Организация данных

Основное назначение логического представления состоит в анализе структурных и функциональных отношений между элементами модели системы. Однако для создания конкретной физической системы необходимо, некоторым образом, реализовать все элементы логического представления в конкретные материальные сущности. Для описания таких реальных сущностей предназначен другой аспект модельного представления, а именно физическое представление модели.

Организация данных подразумевает создание модели данных, главными элементами которой являются сущности и их связи.

Структура базы данных разрабатываемого программного средства включает три таблицы. Структура данных таблиц, и их краткое описание приводится в таблицах 3.1 и 3.2.

Таблица «Пользователь» хранит информацию о пользователях. Структура приведена в таблице 3.1.

3.4 Функции и элементы управления

Для организации в приложении графического пользовательского интерфейса используются элементы управления в виде кнопок, выпадающих списков, таблиц и страниц.

Пользователю предоставляется возможность просматривать, вводить/изменять и выводить данные. При нажатии на активный элемент управления, событие выполняет соответствующее действие.

Для выполнения вышеперечисленных функций необходимо запустить приложение. Откроется страница авторизации.

На основании диаграммы вариантов использования были реализованы следующие функции:

- авторизация;
- просмотр и редактирование данных;
- выдача справок.

3.5 Проектирование справочной системы приложения

Для удобной работы с приложением требуется обеспечить пользователя справочной системой, в которой будут приведены приемы работы с приложением, включающие данные о том, что произойдёт после нажатия на определенную кнопку или при выборе вкладки.

Справочная система необходима для ознакомления с программой. В ней должна присутствовать информация, которая поможет в решении проблемы с приложением, а также может напомнить, как пользоваться программным средством.

Система справки данного приложения будет содержать следующие разделы:

- «Условные обозначения»;
- «Основные понятия и элементы интерфейса»;
- «Работа с программой»;
- «Выход из программы».

Справка будет вызываться клавишей F1 на главной форме.

Она будет разработанна с помощью Microsoft Office Word.

4 Описание программного средства

4.1 Общие сведения

Веб-приложение «mern» представляет собой исполняемый сайт, который предназначен для удалённой работы с отделом кадров.

Данная программа позволяет облегчить работу на предприятии.

Программное средство создано в среде разработки VS Code Insiders на языке программирования JavaScript. Оно может работать на операционных системах Windows 7, 8, 8.1, 10 при установленном браузере. К проекту подключена база данных «Workers.mdf» созданная на платформе MongoDB. Программа не требовательна к системным ресурсам, также проста в использовании и не требует специальных навыков при работе. Программа требует 200 Мбайт свободного пространства на диске.

Предварительная инсталляция происходит путем запуска исполняемого файла «index.html».

4.2 Функциональное назначение

Программа предназначена для пользователей пк.

Программа предназначена для авторизации отдела кадров СТО.

Данная программа значительно облегчает работу с отделом кадров на предприятии.

4.3 Входные и выходные данные

К входным данным являются:

- браузер;
- команды от сервера.

К выходным данным являются:

- команды от клиента о выбраном сотруднике, для просмотра и редактирования данных.

5 Методика испытаний

5.1 Технические требования

Минимальные системные требования для оптимальной работы программного средства является персональный компьютер (ПК) под управлением операционной системы Microsoft Windows 10 со следующими характеристиками:

- процессор 2000 МГц и выше;
- оперативная память 16 Гбайт;
- свободное место на диске 300 Мбайт;
- видеокарта с объемом памяти 4000 Мбайт.

Компьютер должен работать под управлением операционной системы, начиная с Windows 7 и выше. Наиболее удобной операционной системой для проведения испытаний является Windows 10, так как она ориентирована на максимальное использование всех возможностей персонального компьютера (ПК), сетевых ресурсов и обеспечение комфортных условий работы.

5.2 Функциональное тестирование

В процессе написания программного средства необходимо производить тестирование на правильность работы приложения. Одной из основных задач тестирования является устранение ошибок, происходящих при вводе данных.

Функциональное тестирование – это тестирование функций приложения на соответствие требованиям. Оценка производится в соответствии с ожидаемыми и полученными результатами (на основании функциональной спецификации), при условии, что функции отрабатывали на различных значениях входных данных.

Тестирование программы будет производиться последовательно, переходя из одной части программы в другую. Во время теста будут проверяться все действия с программой, навигация пунктам меню, которые может произвести пользователь. После чего, все собранные и найденные ошибки будут исправлены.

Для тестирования функций программы необходимо запустить сначала программу,а уже после этого переходить к тестированию.

6 Применение

6.1 Назначение программы

Веб приложения «mern» представляет собой исполняемый сайт, который предназначен для удалённой работы с отделом кадров.

Данная программа позволяет облегчить работу на предприятии.

Программное средство создано в среде разработки VS Code Insiders на языке программирования JavaScript. Оно может работать на операционных системах Windows 7, 8, 8.1, 10 при установленном браузере. К проекту подключена база данных «Workers.mdf» созданная на платформе MongoDB. Программа не требовательна к системным ресурсам, также проста в использовании и не требует специальных навыков при работе. Программа требует 200 Мбайт свободного пространства на диске.

Предварительная инсталляция происходит путем запуска исполняемого файла «index.html».

6.2 Условия применения

Для применения данного программного средства необходимы следующие технические требования:

- процессор Intel® Core™ i5-2430M, 2.40 ГГц;
- оперативная память 8,00 Гбайт;
- видеокарта Intel® HD Graphics 3000 (2 Гбайт GDDR3);
- твердотельный накопитель 240 Гбайт.
- операционная система Windows 7 и выше;
- наличие клавиатуры, мыши и монитора.

Программа адаптирована под все компьютеры. Для переноса программы на другой компьютер достаточно будет перенести папку с программой.

6.3 Справочная система

Справочная система программного средства представляет собой отдельное окно «help», которые открывается при нажатии F1. В справочной системе даны ответы на типичные вопросы, возникающие при работе с приложением, что должно помочь при освоении программного средства. Инструкция применения была разработанна с помощью Microsoft Office Word.

Заключение

Данная программа будет служить решением для автоматизации отдела кадров СТО.

Для достижения цели проекта были решены следующие задачи:

– определена вычислительная система, необходимая для создания программного средства;

– по модели выполнено проектирование задачи;

– разработаны основные функции программного средства;

– описаны созданные разработанные функции;

– выбрана методика испытаний;

– описан процесс тестирования;

– приведены примеры области применения.

Программа готова к использованию.

Основное преимущество программного средства в том, что программа по сравнению с аналогами проста в использовании, имеет современный и удобный интерфейс, позволяет добавлять данные и легко манипулировать ими.

Для разработки данного программного средства были применены и закреплены знания по уже изученному материалу, были отработаны навыки владения методами надёжного программирования и эффективности разработки программного обеспечения.

Данная программа может быть дополнена и модернизирована.

Список информационных источников

1 Багласова, Т.Г. Методические указания по выполнению дипломного проекта для учащихся по специальности 2-40 01 01 «Программное обеспечение технологий» / Т.Г. Багласова. – Минск : КБП, 2017. – 30 с.

2 Багласова, Т.Г. Методические указания по оформлению курсовых и дипломных проектов / Т.Г. Багласова, К.О. Якимович. – Минск : КБП, 2013. – 29 с.

3 Бондарь, А.Г. Microsoft SQL Server 2012 / А.Г. Бондарь. – СПб. : БХВ-Петербург, 2013. – 608 с.

4 Орлов, С. А. Технологии разработки программного обеспечения: Учебник для вузов. 4-е изд. / С. А. Орлов, Б. Я. Цилькер. – СПб. : Питер, 2012. – 608 с.

5 Михнюк, Т.Ф. Охрана труда / Т.Ф. Михнюк. – Минск : ИВЦ Минфина, 2009. – 365 с.

6 Экономика предприятия. Практикум / Э. В. Крум [и др.] ; под ред. Э. В. Крум. – Минск : Издательство Гревцова, 2009. – 355 с.

7 Общие требования к тестовым документам : ГОСТ 2.105-95. – Введ. 01.01.1996. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 1995. – 84 с.

8 Программа и методика испытаний. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.301-2000. – Введ. 01.09.2001. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 14 с.

9 Текст программы. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.401-2000. – Введ. 01.09.2001. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 16 с.

10 Dr. Explain [Электронный ресурс]. – Режим доступа : https://www.drexplain.ru/– Дата доступа 04.05.2020.

11 Microsoft Office [Электронный ресурс] / Microsoft Word: правка документов и общий доступ. – Google, 2020. – Режим доступа : https://www.office.com – Дата доступа : 20.05.2020.

12 UMLet – Free UML Tools for fast UML diagrams [Электронный ресурс] / UMLet 14.3. – Режим доступа: www.umlet.com. – Дата доступа : 04.05.2020.

13 Выпуски и поддерживаемые функции SQL Server 2017 [Электронный ресурс] / Microsoft, Документация по SQL. – Режим доступа : https://docs.microsoft.com/ru-ru/sql/sql-server/editions-and-components-of-sql-server-2017?view=sql-server-2017. – Дата доступа : 04.06.2020.

14 Обзор и установка SQL Server Management Studio 17 [Электронный ресурс] / Заметки IT специалиста. Блог о компьютерах и программировании для начинающих. –Заметки IT специалиста, 2019. – Режим доступа : https://info-comp.ru/softprodobes/595-review-and-install-ssms-17.html. – Дата доступа 04.05.2020.

15 Обзор обновлений и новых функций Windows 10 [Электронный ресурс]. – Microsoft, 2019. – Режим доступа : https://www.microsoft.com/ru-ru/windows/features. – Дата доступа : 22.05.2020.

16 Общие сведения о Visual Studio [Электронный ресурс] / Microsoft, Документация по Visual Studio. – Режим доступа : https://docs.microsoft.com/ru-ru/visualstudio/ide/visual-studio-ide. – Дата доступа 04.05.2020.

17 Руководство по программированию на C# [Электронный ресурс]. – Microsoft, 2019. – Режим доступа : http://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/. – Дата доступа : 25.04.2020.

Приложение А
(обязательное)
Текст программы

```
const express = require('express')
const config = require('config')
const mongoose = require('mongoose')

const app = express()

app.use('/api/auth', require('./routes/auth.routes'))

const PORT = config.get('port') || 5000

async function start() {
  try {
    await mongoose.connect(config.get('mongoUri'), {
      useNewUrlParser: true,
      useUnifiedTopology: true,
      useCreateIndex: true
    })
    app.listen(PORT, () => console.log(`App has been started on port ${PORT}...`))
  } catch (e) {
    console.log('Server Error', e.message)
    process.exit(1)
  }
}

start()
const {Router} = require('express')
const bcrypt = require('bcryptjs')
const config = require('config')
const jwt = require('jsonwebtoken')
const {check, validationResult} = require('express-validator')
const User = require('../models/User')
const router = Router()

// /api/auth/register
router.post('/register',
[
  check('email', 'Некорректный email.').isEmail(),
  check('pssword', 'Минимальная длина пароля 6 символов').isLength({ min: 6 })
],
async (req, res) => {
try {
    const errors = validationResult(req)

    if(!errors.isEmpty()) {
      return res.status(400).json({
        errors: errors.array(),
        message: 'Некорректные данные при регистрации.'
      })
    }
```

```
    const {email, password} = req.body

    const candidate = await User.findOne({ email })

    if(candidate) {
      return res.status(400).json({ message: 'Такой пользователь уже зарегистрирован.' })
    }

    const hashedPassword = await bcrypt.hash(password, 12)
    const User = new User({ email, password: hashedPassword })

    await user.save()

    res.status(201).json({ message: 'Пользователь успешно зарегестрирован.' })

  } catch (e) {
   res.status(500).json({message: 'Что-то пошло не так, попробуйте снова.'})
  }
  })

  // /api/auth/login
  router.post('/login',
  [
    check('email', 'Введите email').normalizeEmail().isEmail(),
    check('password', 'Введите пароль').exists()
  ],
  async (req, res) => {
  try {
    const errors = validationResult(req)

     if(!errors.isEmpty()) {
      return res.status(400).json({
        errors: errors.array(),
        message: 'Некорректные данные при входе.'
      })
    }

    const {email, password} = req.body

    const user = await User.findOne({ email })

    if(!user) {
      return res.status(400).json({ message: 'Пользователь с данным email не зарегистрирован.' })
    }

    const isMatch = await bcrypt.compare(password, user.password)

    if(!isMatch) {
      return res.status(400).json({ message: 'Вы ввели неверный пароль.' })
    }

    const token = jwt.sign(
      { userId: user.id },
```

```
      config.get('jwtSecret'),
      { expiresIn: '1h' }
    )

    res.json({ token, userId: user.id })

} catch (e) {
  res.status(500).json({message: 'Что-то пошло не так, попробуйте снова.'})
}
})

module.exports = router
const {Schema, model, Types} = require('mongoose')

const schema = new Schema({
  email: {type: String, required: true, unique: true},
  password: {type: String, required: true},
  links: [{ type: Types.ObjectId, ref: 'Link' }]
})

module.exports = model('User', schema)
{
  "port": 5000,
  "jwtSecret": "rostislav mern",
  "mongoUri":
"mongodb+srv://rostislav:qwerty1234@cluster0.1jah8.mongodb.net/app?retryWrites=true&w=majorit
y"
}
{
  "name": "diplom",
  "version": "1.0.0",
  "description": "mern stack",
  "main": "app.js",
  "scripts": {
    "start": "node app.js",
    "server": "nodemon app.js",
    "client": "npm run start --prefix client",
    "dev": "concurrently \"npm run server\" \"npm run client\""
  },
  "keywords": [
    "mern",
    "react",
    "node"
  ],
  "author": "Rostislav Podolskiy <rostypod@gmail.com>",
  "license": "ISC",
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "config": "^3.3.3",
    "express": "^4.17.1",
    "express-validator": "^6.9.2",
    "jsonwebtoken": "^8.5.1",
    "mongoose": "^5.11.15"
  },
```

```
  "devDependencies": {
    "concurrently": "^5.3.0",
    "nodemon": "^2.0.7"
  }
}

import React from 'react'
import 'materialize-css'

function App() {
  return (
    <div className="container">
      <h1>Hello</h1>
    </div>
  )
}

export default App;
@import "~materialize-css/dist/css/materialize.min.css";
{
  "name": "client",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.11.9",
    "@testing-library/react": "^11.2.5",
    "@testing-library/user-event": "^12.6.3",
    "materialize-css": "^1.0.0-rc.2",
    "react": "^17.0.1",
    "react-dom": "^17.0.1",
    "react-router-dom": "^5.2.0",
    "react-scripts": "4.0.2",
    "web-vitals": "^1.1.0"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
```

```
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```