MINISTERUL EDUCAȚIEI, CULTURII ȘI CERCETĂRII AL REPUBLICII MOLDOVA

UNIVERSITATEA DE STAT „ALECU RUSSO” DIN BĂLȚI

FACULTATEA DE ȘTIINȚE REALE, ECONOMICE ȘI ALE MEDIULUI

CATEDRA DE MATEMATICĂ ȘI INFORMATICĂ

**DEZVOLTAREA MODULULUI WEB CALCULATOR PENTRU DEVAMAREA VEHICULULUI**

**TEZĂ DE MASTER**

**Autor:**

Studentul grupei PW21M

**Igor BASISTII**

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

**Conducător științific:**

**Eugeniu CABAC**

dr., conf. univ.

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

**Bălti, 2023**

**BĂLȚI, 2023**

Controlată:

Data\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Conducător științific:

Eugeniu CABAC, dr., conf. univ.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Adnotare

Aprobată

și recomandată pentru susținere

la ședința Catedrei de matematică și informatică

Proces verbal nr.\_\_\_\_\_din\_\_\_\_\_\_\_\_\_\_\_\_

Șeful catedrei de matematică și informatică

dr., conf. univ. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

МИНИСТЕРСТВО ОБРАЗОВАНИЯ, КУЛЬТУРЫ И ИССЛЕДОВАНИЯ РЕСПУБЛИКИ

МОЛДОВА

БЕЛЬЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ “А. РУССО”

ФАКУЛЬТЕТ РЕАЛЬНЫХ НАУК, ЭКОНОМИКИ И ОКРУЖАЮЩЕЙ СРЕДЫ

КАФЕДРА МАТЕМАТИКИ И ИНОРМАТИКИ

**РАЗРАБОТКА МОДУЛЯ WEB КАЛЬКУЛЯТОРА**

**ДЛЯ РАСТАМАЖИВАНИЯ ТРАНСПОРТНЫХ СРЕДСТВ**

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ**

**Автор:**

Студент группы PW21M

**Игорь БАСИСТЫЙ**

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

**Научные руководитель:**

**Евгений КАБАК**

др., конф. унив.

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

**Bălti, 2023**

MINISTRY OF EDUCATION CULTURES AND RESEARCH OF THE REPUBLIC OF MOLDOVA

"ALECURUSSO" STATE UNIVERSITY OF BĂLȚI

FACULTY OF REAL, ECONOMIC AND ENVIRONMENTAL SCIENCES

DEPARTMENT OF MATHEMATICS AND COMPUTING

**DEVELOPMENT OF A WEB CALCULATOR MODULE FOR CUSTOMS CLEARANCE OF VEHICLES**

**MASTER THESIS**

**Author:**

The student of the group PW21M

**Igor BASISTII**

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

**Scientific leader:**

**Eugeniu CABAC**

dr., conf. univ.

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

**Bălti, 2023**

Aprobat

Șeful catedrei de matematică și informatică

dr., conf. univ. E. Plohotniuc

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_202\_\_

**Graficul calendaristic de executare a tezei de master**

Tema tezei de master\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

confirmată prin ordinul rectorului USARB nr.\_\_\_ din „\_\_\_\_\_\_\_\_\_\_\_\_\_”

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Termenul limită | | | de prezentare a tezei de | | | | master la Catedra de | | matematică și informatică | | |  |
| „\_\_\_\_\_\_\_\_\_\_\_”. | | |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | **Etapele executării tezei de master:** | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| Etapele | |  |  |  |  |  |  | Termenul de |  | Viza de |  |  |
|  |  |  |  |  |  | realizare |  | executare |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | | | | | | |  |  |  |  |  |
| 1. | Stabilirea temei; fixarea obiectivelor; selectarea | | | | | | |  |  |  |  |  |
| surselor de informare | | | |  |  |  |  |  |  |  |  |  |
| 2. | Investigația cadrului teoretic al cercetării | | | | | | (teoria |  |  |  |  |  |
| problemei); expunerea cadrului teoretic al cercetării; | | | | | | | |  |  |  |  |  |
|  |  | | | | | | |  |  |  |  |  |
| 3. | Întocmirea problemei cercetării; stabilirea tipului | | | | | | |  |  |  |  |  |
| de cercetare; elaborarea ipotezelor | | | | | |  |  |  |  |  |  |  |
|  |  | |  | |  |  |  |  |  |  |  |  |
| 4. | Specificarea | | unităților | | (populației) |  | studiate; |  |  |  |  |  |
| construcția | | variabilelor | | (descrierea | | calitativă); | |  |  |  |  |  |
| cuantificarea (descrierea cantitativă) | | | | | |  |  |  |  |  |  |  |
| 5. | Alegerea | metodelor | | de | cercetare; | stabilirea | |  |  |  |  |  |
| tehnicilor și procedeelor | | | | de lucru - în conformitate | | | |  |  |  |  |  |
| cu decizia despre caracterul lucrării: experiment de | | | | | | | |  |  |  |  |  |
| constatare, experiment formativ etc. | | | | | |  |  |  |  |  |  |  |
|  |  | | | | | | |  |  |  |  |  |
| 6. | Culegerea datelor; selectarea modalităților de | | | | | | |  |  |  |  |  |
| prelucrare a datelor; stocarea datelor; prelucrarea | | | | | | | |  |  |  |  |  |
| datelor; analiza datelor (verificarea ipotezelor) | | | | | | | |  |  |  |  |  |
| 7. | Rezolvarea aspectelor de grafică și design la | | | | | | |  |  |  |  |  |
| calculator; interpretarea rezultatelor | | | | | |  |  |  |  |  |  |  |
| 8. | Formularea propunerilor de soluționare a | | | | | | |  |  |  |  |  |
| problemei cercetării | | | |  |  |  |  |  |  |  |  |  |
| vizate în lucrare; elaborarea concluziilor și a | | | | | | | |  |  |  |  |  |
| recomandărilor practice | | | |  |  |  |  |  |  |  |  |  |
|  | | | | | |  |  |  |  |  |  |  |
| 9. Susținerea preventivă a tezei | | | | | |  |  |  |  |  |  |  |
|  | | | | | |  |  |  |  |  |  |  |
| Masterand (a) \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | | | |  |  |  |  |  |  |  |
|  |  | (semnătura) | |  |  |  |  |  |  |  |  |  |
| Conducători științifici \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | | | | | | |  |  |  |  |
|  |  |  |  |  | (semnătura) |  |  |  |  |  |  |  |

**ANOTARE**

Basistii Igor

DEZVOLTAREA MODULULUI WEB CALCULATOR PENTRU DEVAMAREA VEHICULULUI

Teză de master. Bălți, 2023

*Structura tezei:* introducere, patru capitole, concluzii generale și recomandări, ZZ titluri ale bibliografiei, ZZ de pagini ale textului principal, zz de figuri.

**АННОТАЦИЯ**

Игорь Басистый

РАЗРАБОТКА МОДУЛЯ WEB КАЛЬКУЛЯТОРА

ДЛЯ РАСТАМАЖИВАНИЯ ТРАНСПОРТНЫХ СРЕДСТВ

Магистерская диссертация.Balti,2023

*Структура диссертации:* введение, четыре главы, общие выводы и рекомендации, ZZ названий библиографии, ZZ страниц основного текста, ZZ рисунков.

*Ключевые слова*

*Область изучения:*

*Цель исследования*.

*Цели исследования:*

*Методология исследования состояла из теоретических методов:*

**ANNOTATION**

Basistii Igor

DEVELOPMENT OF A WEB CALCULATOR MODULE FOR CUSTOMS CLEARANCE OF VEHICLES

Bachelorthesis.Balti,2023

*The structure of the thesis: introduction, four chapters, general conclusions and recommendations, ZZ titles of the bibliography, ZZ pages of the main text, ZZ figures.*

*Keywords.*

*Study area:*

*Purpose of the research:*

*Research objectives:*

*The research methodology consisted of theoretical methods:*

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ](#page10) [8](#page10)

Глава 1. Преимущества модульного перед обычным приложением.[**14**](#page14)

1.1. Определение понятия "модульное приложение" [16](#page16)

1.2. Преимущества модульного приложения перед обычным [17](#page17)

1.3. Примеры успешных модульных приложений [17](#page17)

Глава 2. Методы разработки модульного приложения калькулятора для растаможивания транспортных средств. [19](#page19)

2.1. Обзор существующих методов разработки модульных приложений [19](#page19)

2.2. Выбор метода разработки, подходящего для данного проекта [20](#page20)

2.3. Описание этапов разработки [20](#page20)

Глава 3. Проектирование и разработка модульного приложения калькулятора для растаможивания транспортных средств. [24](#page24)

3.1. Анализ требований к приложению [24](#page24)

3.2. Проектирование структуры модульного приложения [28](#page28)

3.3. Описание основных функций и возможностей приложения [28](#page28)

3.4. Разработка приложения на основе выбранного метода разработки [28](#page28)

Глава 4. Тестирование и отладка разработанного приложения. [37](#page37)

4.1. Выбор методов тестирования [37](#page37)

4.2. Описание процесса тестирования и отладки [46](#page46)

Глава 5. Заключение [37](#page37)

5.1. Общие выводы по проделанной работе [37](#page37)

5.2. Перспективы развития приложения [46](#page46)

5.3. Возможные направления дальнейших исследований [46](#page46)

[БИБЛИОГРАФИЯ](#page68) [68](#page68)

7

**ВВЕДЕНИЕ**

Автомобильная индустрия является одной из ключевых отраслей экономики многих стран, играющей важную роль в массовой транспортировке, а также в экспорте и импорте товаров. В связи с этим, процесс растаможивания автомобилей является неотъемлемой частью этой отрасли, требующей детального и точного расчета необходимых платежей. Для упрощения этого процесса было создано множество инструментов, в том числе и модульные приложения калькуляторы.

Тема данной магистерской диссертации – "Модульное приложение калькулятор для растаможивания автомобилей". Цель данной работы заключается в разработке и анализе такого модульного приложения, которое позволит автоматизировать процесс расчета необходимых платежей при растаможивании автомобилей с учетом множества параметров, таких как таможенная стоимость, акцизы, НДС и другие налоги.

В первой главе данной магистерской диссертации будет проведен обзор литературы, в котором будут рассмотрены существующие методы расчета необходимых платежей при растаможивании автомобилей, а также существующие модульные приложения калькуляторы. Будут выявлены их основные преимущества и недостатки, а также определены основные требования к разрабатываемому модульному приложению калькулятору.

Во второй главе будет представлена разработка модульного приложения калькулятор для растаможивания автомобилей. Будут описаны основные алгоритмы, используемые в приложении, а также его структура и функциональные возможности. Будет проведен анализ разработанного приложения и его сравнение с существующими решениями.

В третьей главе будет представлен экспериментальный анализ работы модульного приложения калькулятора для растаможивания автомобилей. Будут определены критерии оценки эффективности разработанного приложения, проведены необходимые тесты и эксперименты. На основе результатов экспериментов будет проведен анализ эффективности разработанного приложения и его сравнение с другими существующими решениями.

В заключительной главе будут представлены основные результаты работы, а также выводы и рекомендации по дальнейшей разработке и улучшению модульного приложения калькулятора для растаможивания автомобилей. Будут рассмотрены ограничения и возможности приложения, а также потенциальные области применения.

8

Итак, разработка модульного приложения калькулятора для растаможивания автомобилей является актуальной задачей, которая может значительно упростить процесс расчета необходимых платежей и повысить эффективность работы таможенных органов. В данной работе будут рассмотрены существующие методы и решения, а также представлено разработанное модульное приложение калькулятора с последующим анализом его эффективности.

Целью работы является исследование

Достижение целей предполагает:

* Анализ специальных информационных источников;
* Разработка демонстрационного приложение.

Работа состоит из: введение, заключение, ZZ-ти глав и библиографии.

Первая глава посвящена описанию проблем и реализация решения для растамаживания автомобилей.

В описывается практическая часть магистерской работы.

Работа состоит из ZZ страниц тематического содержания, ZZ рисунка, ZZ источников

**Глава 1. Преимущества модульного приложения калькулятора для растаможивания транспортных средств перед обычным приложением.**

**1.1. Определение понятия "модульное приложение"**

Модульное приложение - это программное обеспечение, разработанное с использованием модульной архитектуры, которая позволяет разделять приложение на отдельные независимые модули. Каждый модуль может выполнять свою функцию и взаимодействовать с другими модулями для создания полноценного приложения..

**1.2. Преимущества модульного приложения перед обычным**

Модульное приложение представляет собой программное обеспечение, которое состоит из отдельных компонентов или модулей, каждый из которых выполняет определенную функцию. В отличие от обычных приложений, которые могут быть сложными и неструктурированными, модульное приложение имеет множество преимуществ перед обычным.

1.2.1. Гибкость и масштабируемость

Модульное приложение позволяет разделить сложную задачу на более простые модули, которые могут быть разработаны и тестированы отдельно. Это упрощает процесс разработки и позволяет быстро адаптироваться к изменениям требований. Кроме того, модульное приложение легко масштабируется, поскольку новые модули могут быть добавлены без необходимости изменения всего приложения.

1.2.2. Удобство тестирования

Каждый модуль может быть протестирован отдельно, что облегчает обнаружение и устранение ошибок. Тестирование модулей также позволяет быстрее выпустить приложение на рынок и увеличить его надежность.

1.2.3. Удобство сопровождения

Модульное приложение упрощает процесс сопровождения, поскольку каждый модуль может быть изменен и улучшен отдельно, без необходимости вмешательства в другие модули. Это также позволяет сократить затраты на сопровождение приложения.

1.2.4. Повторное использование кода

Модули могут быть повторно использованы в других приложениях, что позволяет ускорить процесс разработки и сократить затраты на него.

1.2.5. Упрощение командной работы

Модульное приложение облегчает командную работу, поскольку каждый модуль может быть разработан отдельно разными членами команды. Это ускоряет процесс разработки и улучшает качество конечного продукта.

**1.3. Примеры и описание успешных модульных приложений**

Модульное программное обеспечение (МПО) имеет множество применений и используется в различных сферах, начиная от автомобильной промышленности и заканчивая разработкой игр, многие фреймворки тоже имеют модульную основу. В данной главе рассмотрим несколько примеров успешных модульных приложений.

1.3.1. React

React - это JavaScript-библиотека для создания пользовательских интерфейсов. React использует модульную архитектуру, которая позволяет разбить пользовательский интерфейс на множество небольших, независимых компонентов. Каждый компонент отвечает за определенную часть интерфейса и может быть использован повторно в различных частях приложения.

Преимущества React заключаются в высокой скорости работы, легкой масштабируемости и простоте в использовании. Благодаря модульной структуре, разработчики могут легко изменять и добавлять новые компоненты без необходимости вносить изменения в другие части приложения.

1.3.2. AngularJS

AngularJS - это JavaScript-фреймворк для разработки веб-приложений. Он использует модульную архитектуру, которая позволяет разработчикам разбить приложение на множество независимых модулей. Каждый модуль может содержать набор компонентов, сервисов и директив.

Преимущества AngularJS заключаются в том, что он предоставляет мощный набор инструментов для разработки веб-приложений, включая роутинг, валидацию форм, анимации и многое другое. Благодаря модульной структуре, разработчики могут легко управлять зависимостями и расширять функциональность приложения.Модульное приложение упрощает процесс сопровождения, поскольку каждый модуль может быть изменен и улучшен отдельно, без необходимости вмешательства в другие модули. Это также позволяет сократить затраты на сопровождение приложения.

1.3.2. Медиаплеер VLC

VLC - это бесплатный и открытый исходный код медиаплеер, который позволяет воспроизводить практически любые форматы аудио и видео. VLC создан с использованием модульной архитектуры, что позволяет легко добавлять новые функции и поддерживать существующие без необходимости изменения всего приложения. Каждый модуль представляет собой отдельный компонент, который содержит свою собственную логику и обеспечивает определенную функциональность. Всякий раз, когда VLC открывает новый файл, он автоматически определяет формат файла и загружает соответствующий модуль для обеспечения воспроизведения.

1.3.4. Slack

Slack - это платформа для коммуникации и совместной работы, которая является примером успешного модульного приложения. Slack состоит из множества небольших модулей, которые позволяют пользователям отправлять сообщения, управлять задачами, интегрироваться с другими сервисами и т.д. Каждый модуль обеспечивает отдельную функциональность, что позволяет легко масштабировать приложение и добавлять новые возможности.

1.3.5. Airbnb

Airbnb - это онлайн-сервис, который позволяет людям сдавать и арендовать жилье. Приложение Airbnb состоит из множества модулей, которые обеспечивают различные функции, такие как поиск жилья, бронирование, оплата, обработка отзывов и т.д. Каждый модуль может быть легко изменен или заменен без воздействия на другие модули, что обеспечивает гибкость и масштабируемость приложения. Глава 2. Методология разработки модульного приложения калькулятора для растаможивания транспортных средств.

**Глава 2. Методы разработки модульного приложения калькулятора для растаможивания транспортных средств.**

**2.1. Обзор существующих методов разработки модульных приложений.**

Спецификация программного модуля состоит из функциональной спецификации модуля, которая описывает семантику функций, выполняемых модулем для каждого входа и синтаксической спецификации входа, которая позволяет используемому языку программирования конструировать синтаксически корректные вызовы модуля. Функциональная спецификация модуля определяется по тем же принципам, что и функциональная спецификация программной системы.

Существуют различные способы разработки структуры программы модуля, в зависимости от того, что определяет порядок программирования и отладки модулей, заданных в этой структуре. Обычно обсуждаются два метода: метод восходящей разработки и метод нисходящей разработки. А так же ниже опишем и другие методы.

таким образом, чтобы успешно проходить эти тесты. TDD позволяет повысить качество кода, уменьшить количество ошибок и ускорить процесс разработки.

При разработке модульных приложений важно выбрать подходящий метод разработки, учитывая особенности проекта и требования заказчика. Рассмотрим несколько методов разработки модульных приложений.

**2.1.1 Метод нисходящей разработки**

Метод нисходящей разработки (Top-Down) предполагает разработку приложения начиная с высокоуровневых абстракций и постепенным углублением в детали. Этот метод подходит для разработки больших и сложных приложений, где важно сначала определить основную структуру и функциональность приложения, а затем добавлять дополнительные модули и функции.

Преимущества метода нисходящей разработки:

Позволяет разбить большой проект на более мелкие и понятные части

Позволяет сконцентрироваться на общей структуре приложения и его ключевых элементах

Недостатки метода нисходящей разработки:

Риск недооценить некоторые детали и проблемы, которые могут возникнуть в процессе разработки

Возможность несоответствия высокоуровневых абстракций реальным требованиям и возможностям технологий

**2.1.2 Метод восходящей разработки**

Метод восходящей разработки (Bottom-Up) предполагает разработку приложения начиная с наименьших деталей и постепенным объединением в более крупные модули и системы. Этот метод подходит для небольших и средних проектов, где важно быстро создать и отлаживать основную функциональность приложения.

Преимущества метода восходящей разработки:

Позволяет быстро создать и отлажить базовую функциональность приложения

Позволяет тестировать и отлаживать каждый модуль приложения отдельно

Недостатки метода восходящей разработки:

Низкая гибкость: в процессе разработки сложных систем могут возникнуть изменения требований, что может потребовать изменения уже созданных модулей и, как следствие, большого количества доработок;Высокая стоимость: для реализации проекта может потребоваться большое количество времени и ресурсов;

Сложность проектирования: для успешной реализации проекта требуется хорошее знание всей системы в целом, что может потребовать большого количества времени и сил на ее проектирование;

Требуется большое количество времени на этапе анализа требований: в случае, если анализ требований не проводится достаточно внимательно и полно, это может привести к неудачам на более поздних этапах разработки;

Высокая зависимость от качества документации: при использовании метода восходящей разработки необходимо составлять детальную и четкую документацию на каждый модуль, что может потребовать дополнительного времени и ресурсов.

**2.1.3 Конструктивный подход:**

Преимущества: позволяет разрабатывать приложение поэтапно, что упрощает процесс разработки и позволяет быстрее получать результаты работы. Также данный подход позволяет создавать гибкую и масштабируемую архитектуру;

Недостатки: требует тщательного планирования и описания каждого этапа разработки, что может потребовать большого количества времени и ресурсов. Также может потребоваться более высокая квалификация участников проекта, так как каждый этап требует определенных знаний и навыков.

**2.1.4 Архитектурный подход:**

Преимущества: позволяет создавать систему с гибкой и масштабируемой архитектурой, что облегчает ее дальнейшее развитие и поддержку;

Недостатки: требует более тщательного и глубокого анализа требований, чтобы правильно спроектировать архитектуру приложения. Также может потребоваться более высокая квалификация участников проекта, так как создание и поддержка архитектуры требует определенных знаний и навыков.

**2.1.5. Нисходящая реализация**

Нисходящая реализация (top-down implementation) - метод разработки, при котором сначала проектируется абстрактный уровень системы, затем постепенно добавляются нижние уровни, пока не достигнут уровни кода и данных. Этот метод включает в себя следующие этапы разработки:

Определение целей и задач проекта.

Разработка абстрактной модели.

Определение взаимодействия между модулями.

Разработка модулей на более низком уровне.

Реализация кода.

Тестирование и отладка системы.

Преимущества метода нисходящей реализации:

Позволяет более точно определить необходимый функционал и проектировать систему на более высоком уровне абстракции, что способствует упрощению ее дальнейшей реализации.

Позволяет проводить более глубокий анализ потребностей пользователей и принимать во внимание их мнения еще на этапе проектирования.

Недостатки метода нисходящей реализации:

Сложность в отслеживании зависимостей между модулями на разных уровнях абстракции.

Трудность внесения изменений на поздних этапах разработки.

Неэффективность, когда необходимо быстро создать MVP (minimum viable product) для быстрого запуска на рынок.

В данном проекте метод нисходящей реализации может быть использован для проектирования и реализации модульного приложения, так как он позволяет более точно определить необходимый функционал и упрощает дальнейшую реализацию. Однако, следует учитывать возможные недостатки, такие как трудность внесения изменений на поздних этапах разработки и неэффективность при необходимости быстрого создания MVP. Для обхода недостатков методов восходящей и нисходящей разработки, можно использовать целенаправленную конструктивную реализацию (СКР).

Целенаправленная конструктивная реализация - это метод разработки, который объединяет преимущества и избегает недостатков методов нисходящей и восходящей разработки. СКР позволяет более эффективно создавать и поддерживать большие проекты, предназначенные для длительной эксплуатации.

СКР использует следующие шаги:

Определение функциональных требований к приложению.

Разработка абстрактной модели архитектуры, которая отражает требования к системе и описывает, как будут взаимодействовать модули.

Разработка прототипа, который имеет минимальный функционал, но обеспечивает тестирование и отладку.

Дополнение прототипа новыми функциями и взаимодействие с другими модулями на основе абстрактной модели архитектуры.

Тестирование и оптимизация системы.

СКР позволяет разрабатывать гибкие приложения, которые могут легко изменяться на поздних этапах разработки. Он также позволяет создавать MVP быстро, что полезно при работе с короткими сроками или ограниченными ресурсами.

Однако, следует учитывать, что СКР может быть более сложным и затратным по сравнению с другими методами разработки, поэтому не всегда является лучшим выбором в конкретной ситуации.

**2.2 Выбор методологии разработки**

Для разработки модульного приложения калькулятора для растаможивания транспортных средств на языке JavaScript мы выберем методологию восходящей разработки, так как этот метод позволяет начинать разработку с более высокоуровневых концепций и постепенно уточнять детали проекта, что особенно подходит для проектов, требующих точности в вычислениях, таких как калькулятор.

Процесс разработки по методу восходящей разработки начинается с выделения наивысшего уровня абстракции в виде функциональных требований и продолжается до достижения наименьшего уровня абстракции, что является преимуществом данной методологии при разработке калькулятора.

На рисунке ниже представлены основные этапы процесса разработки по методу восходящей разработки:

Как видно из графика, на первом этапе мы определяем общие требования к проекту, такие как функциональные возможности, интерфейс и производительность. На втором этапе мы проектируем общую архитектуру приложения, определяя структуру и интерфейсы между модулями. На следующих этапах мы постепенно добавляем функциональность и детализируем реализацию на более низких уровнях.

При выборе методологии разработки для модульного приложения калькулятора на языке JavaScript следует учитывать особенности языка, такие как динамическая типизация и поддержка объектно-ориентированного программирования. Метод восходящей разработки позволяет лучше структурировать приложение и избежать ошибок, связанных с динамической типизацией. Кроме того, модульность является одним из ключевых принципов объектно-ориентированного программирования, что делает эту методологию особенно подходящей для разработки модульного приложения.

Таким образом, мы выбираем метод восходящей разработки для создания модульного приложения калькулятора для растаможивания транспортных средств на языке JavaScript, так как этот метод позволяет лучше структурировать приложение и избежать ошибок, связанных с несоответствием архитектуры и функциональности, он был выбран в качестве основной методологии разработки модульного приложения калькулятора для растаможивания транспортных средств на языке JavaScript.

Для применения метода восходящей разработки к данному проекту разработки модульного приложения калькулятора для растаможивания транспортных средств необходимо выполнить следующие шаги:

Определить общие требования к приложению и функциональные блоки, которые будут использоваться для реализации этих требований.

Разработать детальные спецификации для каждого функционального блока, определяя его входные и выходные данные, а также необходимые алгоритмы и процедуры.

Создать шаблоны для каждого функционального блока, определяя необходимые переменные и методы для его реализации.

Реализовать каждый функциональный блок в соответствии со спецификацией, используя предварительно разработанные шаблоны.

Протестировать каждый функциональный блок на соответствие спецификации и на отсутствие ошибок.

Объединить все функциональные блоки в единое приложение и протестировать его на работоспособность и соответствие требованиям.

Примером графического представления метода восходящей разработки может быть следующая схема:

Схема метода восходящей разработки

Таким образом, метод восходящей разработки является подходящим методом для разработки модульного приложения калькулятора для растаможивания транспортных средств на языке JavaScript, так как он позволяет создавать приложение, которое соответствует требованиям и имеет четкую архитектуру, что упрощает его поддержку и дальнейшее развитие.

**Глава 2.3. Описание этапов разработки основанный на методе восходящей разработки**

Метод восходящей разработки позволяет разработчикам проектировать и создавать приложение снизу вверх. Этот метод является итеративным, поэтому каждый этап разработки может повторяться несколько раз.

**2.3.1. Определение требований**

Первый этап разработки - определение требований. На этом этапе разработчики определяют, какие функциональные возможности должно иметь приложение, а также какие ограничения и требования должны быть учтены при его создании.

**2.3.2. Проектирование архитектуры**

На этапе проектирования архитектуры разработчики определяют структуру приложения, составляют план его разработки, а также выбирают подходящие инструменты и технологии для создания приложения на языке JavaScript.

**2.3.3. Проектирование модулей**

Следующий этап - проектирование модулей приложения. Разработчики определяют функциональные блоки приложения, их зависимости и интерфейсы, чтобы приложение было более гибким и легко расширяемым в будущем.

**2.3.4. Нисходящее проектирование модулей**

На этапе нисходящего проектирования разработчики проектируют модули приложения в более детальном виде. Они определяют точную спецификацию каждого модуля и его интерфейса.

**2.3.5. Создание модулей**

После проектирования модулей начинается создание кода на языке JavaScript. Разработчики создают код для каждого модуля, используя заранее определенную архитектуру и интерфейсы.

**2.3.6. Тестирование модулей**

На этапе тестирования модулей разработчики проверяют, работает ли каждый модуль правильно и соответствует ли он спецификации, определенной на предыдущих этапах.

**2.3.7. Интеграция модулей**

После успешного тестирования модулей разработчики интегрируют их в приложение. Они проверяют, как модули взаимодействуют между собой и как приложение работает в целом.

**2.3.8. Тестирование приложения**

На последнем этапе разработчики проверяют, что все модули приложения работают корректно и без ошибок. Для этого используются различные методы тестирования, такие как модульное тестирование, интеграционное тестирование, функциональное тестирование и т.д.

Модульное тестирование позволяет проверить работу каждого модуля приложения в отдельности. Это позволяет быстрее выявлять ошибки и устранять их на ранних этапах разработки.

Интеграционное тестирование проверяет работу модулей взаимодействующих друг с другом. Таким образом, разработчики могут убедиться, что все модули работают корректно в совокупности.

Функциональное тестирование позволяет проверить работу приложения в целом, то есть его функциональность и соответствие требованиям заказчика. В процессе функционального тестирования тестируются различные сценарии использования приложения.

**2.3.9. Документирование**

На каждом этапе разработки приложения важно вести документацию. В документации должны быть описаны все этапы разработки приложения, используемые технологии и методы, а также структура приложения и его функциональность.

Важным элементом документации является документация кода. Каждый модуль приложения должен быть подробно описан, с указанием входных и выходных данных, используемых алгоритмов и т.д. Это позволяет разработчикам легко понимать код, а также быстро находить и исправлять ошибки.

**2.3.10. Внесение изменений**

В процессе разработки приложения могут возникать необходимость внесения изменений. Метод восходящей разработки позволяет быстро внести изменения на ранних этапах разработки, что значительно сокращает время разработки и уменьшает затраты.

**Выводы**

В данной главе были рассмотрены различные методологии разработки модульных приложений, в том числе метод восходящей разработки, который был выбран для разработки модульного приложения калькулятора для растаможивания транспортных средств на языке JavaScript.

Были описаны этапы разработки, включая анализ требований, проектирование, разработку, тестирование и документирование приложения. Были представлены рекомендации по эффективной разработке модульного приложения на языке JavaScript, включая использование модульной структуры кода, тестирования, системы контроля версий, документирования, безопасности приложения и автоматизации процессов разработки.

Выбранный метод разработки позволяет лучше структурировать приложение и избежать ошибок, связанных с необходимостью внесения изменений на поздних этапах разработки. Он также обеспечивает более эффективный процесс тестирования и позволяет улучшить качество приложения.

**Глава 3. Проектирование и разработка модульного приложения калькулятора для растаможивания транспортных средств.**

**3.1. Анализ требований к приложению.**

Перед началом разработки модульного приложения калькулятора для растаможивания транспортных средств необходимо провести анализ требований к приложению. Это позволит определить необходимый функционал приложения и основные характеристики, такие как производительность, надежность и удобство использования.

Основными требованиями к приложению являются:

1. Возможность расчета стоимости растаможки транспортного средства по его характеристикам, таким как тип транспорта, год выпуска, объем двигателя и т.д.

2. Поддержка различных тарифных ставок и методов расчета.

3. Возможность сохранения результатов расчета для последующего использования.

4. Простой и интуитивно понятный интерфейс для пользователей.

5. Высокая точность расчетов.

Для реализации этих требований необходимо провести более детальный анализ каждого из них и определить соответствующие функциональные и нефункциональные требования к приложению. Также необходимо учитывать специфические требования к языку программирования JavaScript, в котором будет разрабатываться приложение.

**3.2. Проектирование структуры модульного приложения.**

После анализа требований к приложению, необходимо разработать структуру модульного приложения. Для написания калькулятора для растаможивания транспортных средств на языке JavaScript, структура приложения должна быть построена с учетом особенностей данного языка.

Для организации модульности приложения в JavaScript используется модульная система, предоставляемая средой выполнения, например, Node.js или браузером. Для этого используется конструкция export и import. Также используется модульный паттерн проектирования, который позволяет создавать отдельные модули с определенным функционалом.

Структура модульного приложения для калькулятора растаможивания может быть организована следующим образом:

- app/

- js/

- main.js

- modules/

- calculator.js

- vehicle.js

- duty.js

- tax.js

- utils/

- format.js

app - корневая директория приложения.

js - директория, содержащая JavaScript-файлы.

main.js - главный файл, в котором происходит импорт необходимых модулей и вызов функций для запуска приложения.

modules - директория, содержащая модули приложения.

calculator.js - модуль, отвечающий за работу калькулятора и импортирующий модули vehicle.js, duty.js и tax.js.

vehicle.js - модуль, содержащий информацию о транспортном средстве и его стоимости.

duty.js - модуль, содержащий информацию о пошлинах.

tax.js - модуль, содержащий информацию о налогах.

utils - директория, содержащая вспомогательные модули.

format.js - модуль, отвечающий за форматирование выводимых данных.

Такая структура приложения позволяет легко расширять его функциональность, добавляя новые модули и использовать уже существующие модули для других проектов.

В следующей главе будет описан процесс написания и тестирования модулей приложения.

**ЗАКЛЮЧЕНИЕ**

**БИБЛИОГРАФИЯ**

1. FARRELL, BУТ. Web Components in Action. 2019. ISBN 9781617295775;
2. Николай Чесноков. Модульное тестирование:что это? Типы, инструменты [On-line].[20.04.2023]. Disponibil:https://logrocon.ru/news/unit\_testing
3. Александр Храповицкий. Преимущества и недостатки модульного программирования. [On-line]. [19.04.23]. Disponibil: https://eax.me/modular-programming/
4. Шон Макграт. Программирование на JavaScript: модульность. [On-line]. [20.04.23]. Disponibil: https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide/Modules;
5. MILER, Jason, OSMANY, Addy. Rendering on the Web. [On-line]. [26.11.2020]. Disponibil: https://developers.google.com/web/updates/2019/02/rendering-on-the-web;
6. BORGGREVE, Bram. Server-Side Enterprise Development with Angular. 2018. ISBN 1789806267;
7. Официальная документация фреймворка Angular. [On-line]. [04.11.2020]. Disponibil: https://angular.io/docs;