

Проект

по

Мултимедийни технологии

на тема

„Microinvest“

**Изготвили:** Преслав Пламенов Анев, 61663,

Ростислав Венциславов Големанов, 61667

СУ“Св. Климент Охридски“,

Факултет по математика и информатика, Специалност: Софтуерно инженерство, Четвърти курс.

Дата: 30.06.2017

1. Описание на “Microinvest”.

MicroInvest е автоматизирана система, която използва счетоводен софтуер и може да бъде използвана за водене на отчет за транзакции и генериране на фактури и също така предоставя мултимедийни елементи като графики, които биват „оживени“ от бизнес логика.

* 1. Функционални и нефункционални изисквания

В тази секция ще представим функционалните и нефункционалните изисквания, които трябваше да дефинираме, при изпълнението на проекта.

* + 1. Функционални изисквания
* Системата трябва да предоставя възможност за създаване на акаунт, с псевдоним, парола, имейл.
* Системата трябва да предоставя възможност за създаване на титулярни потребители, такива които ще се прикрепят към компании и счетоводни документи.
* Системата трябва да предоставя възможност за добавяне на компании.
* Системата трябва да предоставя възможност за преглед на всички налични потребители с техните атрибути и полета.
* Системата трябва да предоставя възможност за преглед на всички налични компании с техните атрибути и полета.
* Системата трябва да предлага няколко начина за плащане(по банков път, наложен платеж и т.н)
* Системата трябва да изобразява графиките базирани на данни от базата данни за не повече от една секунда.
  + 1. Нефункционални изисквания
* Базата данни трябва да разполага с достатъчно място за всички документи, компании и т.н, които ще съхранява.
* Изисквания по безопасност са свързани с надеждността и сигурността на сървъра и базата данни.
* Изисквания за производителност, създаване на нов документ, например фактура, с всички полета попълни не трябва да отнема повече от 2 секунди.
* Изисквания за производителност, зареждането на екрана за прегледа на всички потребители не трябва да отнема повече от 3 секунди.
  1. Use-cases в Microinvest.

|  |  |  |
| --- | --- | --- |
| 1. Кратко описание на потребителските случаи (Use cases) | | |
| **Име на потребителския случай** | **Кратко описание (Brief Descriptions)** | **Кратко описание на актьорите (Actor Brief Descriptions)** |
| **1. Запознаване със системата** | Потребителят преглежда наличните страници в уеб приложението. | Нерегистриран потребител, Регистриран потребител |
| **2. Създаване на потребител** | Регистриран потребител създава потребител, който може да асоциира с компания и също така да прикрепи счетоводен документ. | Нерегистриран потребител, Регистриран потребител |
| **3. Преглед на потребители** | Потребителят има възможност да разгледа всички създадени потребители, кратка информация за всеки. | Нерегистриран потребител, Регистриран потребител. |
| **4. Създаване на счетоводен документ(фактура)** | Потребителят има възможност да създаде фактура като опише стока, цена, компания и още допълнителна информация. | Нерегистриран потребител, Регистриран потребител. |
| **5. Добавяне на компания(титуляр)** | Потребителят има възможност да добави компания към фактурата и да я асоциира с вече съществуващ потребител и транзакция. | Нерегистриран потребител, Регистриран потребител. |
| **6. Преглед на графики, диаграми** | Потребителят преглежда наличните графики, диаграми(мултимедия). | Нерегистриран потребител, Регистриран потребител |

2. Описание на използваните технологии и библиотечни модули с референции.

2.1 Използвани технологии за реализация на front-end частта от системата.

* [React.Js](https://facebook.github.io/react/)

React.js e javascript библиотека за изграждане на потребителски интерфейс.

За интерфейсът иползван в Microinvest е използван React.js. Всички екрани или изгледи за изработени чрез React.js. Възползвали сме се от компонентно-базирания характер на React.js, за да постигнем интерактвиен и комплексен потребителски интерфейс.

* [Redux](http://redux.js.org/docs/introduction/)

Използвали сме също Redux, който е много удобна технология за използване в комбина с React.js. Включихме я в проекта, тъй като внася организираност, подреденост, контролиране на потокa от информация в един обект(store) и също така би било полезно когато системата ни започне да се разраства(scalling).

* HTML 5, CSS 3, JS, jQuery, Bootstrap

Използвали сме стандартните технологии за изграждане на потребителски интерфейс(таблици, навигация, картинки, контейнери, поведение, дизайн, изглед и т.н)

* [Node.js/Node-restful](https://github.com/baugarten/node-restful)

Тази технология е използвана за реализация на комуникацията между уеб приложението и сървърната част. Поддържа основните http заявки :

GET /resources

GET /resources/:id

POST /resources

PUT /resources/:id

DELETE /resources/:id

Също така е много удобна за ползване с Mongoose, който сме използвали за базата данни и Express.

* [Express](https://expressjs.com/)

Express е лека и удобна Node.js платформа за изграждане на уеб приложения. Точно затова сме я включили в проекта тъй като и двете части на нашата система използват express, за да бъдат стартирани. Позволява ни, без особено много конфигурация да получим работещи и достъпни инстанция на клиентската и сървърна част.

2.2 Използвани технологии за реализация на back-end частта от системата.

* [Node.js/Node-restful](https://github.com/baugarten/node-restful)

Тази технология е използвана за реализация на комуникацията между уеб приложението и сървърната част. Поддържа основните http заявки :

GET /resources

GET /resources/:id

POST /resources

PUT /resources/:id

DELETE /resources/:id

Също така е много удобна за ползване с Mongoose, който сме използвали за базата данни и Express.

* [Express](https://expressjs.com/)

Express е лека и удобна Node.js платформа за изграждане на уеб приложения. Точно затова сме я включили в проекта тъй като и двете части на нашата система използват express, за да бъдат стартирани. Позволява ни, без особено много конфигурация да получим работещи и достъпни инстанция на клиентската и сървърна част.

* [Mongoose](http://mongoosejs.com/)

Mongoose е удобен и елегантен начин за оформяне на моделите, които влизат и се пазят от базата данни. Позволява ни лесно да добавим валидация, кастване към типове изграждане на заявки. Предоставя възможността да оформим обектите си чрез схема, която после mongoDB използва в колекциите си.

* [MongoDB](https://www.mongodb.com/)

Изполвали сме MongoDB, тъй като и двамата имаме опит с нея, както и поради факта, че е изключително удобна за манипулация от страна на Node.js – базирани системи, каквато е Microinvest. Също така е подходяща за разрастване(scalling).

3. Описание на архитектурата и реализация на системата.

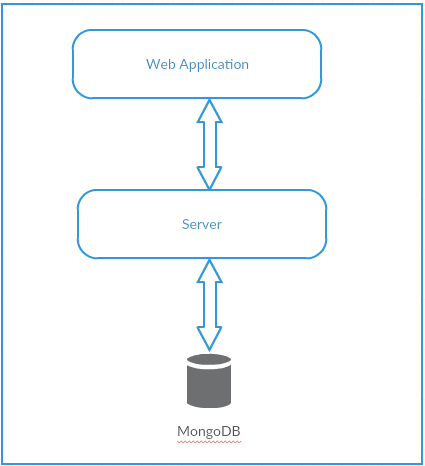
3.1 Описание на архитектурата

Архитектурата на Microinvest, се състои от две автономни

Системи и база данни:

**Клиентска част** – предоставяща уеб интерфейс, с който потребителите взаимодействат и изпращат потребителски вход към сървъра за обработка;

**Сървърна част**- приема заявките идващи от клиентското приложение, обработва ги и запазва в базата данни.



3.2 Реализация на системата

3.2.1 Реализация на Клиентската част

Клиентската част е реализирана основно с помощта на React + Redux, като за самите интерфейс

елементи са използвани html, css за изгледа, дизайна, разположението, javascript за поведението на елементите(event handlers, functions, etc.)

3.2.2 Реализация на сървърната част

Сървърната част представлява, класове които обработват заявките от клиентската част, като също така има модели, схеми, по които обектите които ще се запазват в базата данни са дефинирани.

Комуникацията между двете части на системата е реализирана чрез стандартни http заявки, като в Microinvest, конкретно се използват get – за получаване на обекти от базата данни, post – за създаване на обекти, delete – за изтриване на обекти.

4. Описание на архитектурно значимите интерфейси.

5. Описание на начин на инсталиране на разработената система.

6. Заключение.

Докато разработвахме Microinvest, не срещнахме особени затрдунения. Може би, най-сложната част беше успешното настройване на express, който да „вдигне“ клиентската част и сървърната част едновременно. Също така процесът по конфигуриране на Heroku хостингът, който сме използвали, за да може Microinvest, да бъде достъпен 24/7 отне повече време от колкото очаквахме.

Някои от нещата, които със сигурност ще включим в системата в бъдеще са:

* Въвеждане в системата на акаунти, регистриране, логин на потребители, аутентикация;
* Въвеждане на допълнителни бизнес обекти(фактури, компании, документи и т.н);
* Въвеждане на повече мултимедия(клипове, видеа).
* Версия на системата за мобилни приложения.

7. Използвани материали.

* React.js – Javascript библиотека за изграждане на потребителски интерфейс.

<https://facebook.github.io/react/>

* Redux – контейнер на информация за javascript приложения

<https://github.com/reactjs/redux>

* Node-restful – библиотека осигуряваща REST API, работеща с express.

<https://github.com/baugarten/node-restful>

* Mongoose – обектно моделиране на mongodb обекти.

<http://mongoosejs.com/>

* Creating a basic site with Node.js and Express – Подробна

информация за създаване на уеб приложение с помощта на Node.js & Express.

<https://shapeshed.com/creating-a-basic-site-with-node-and-express/>

* Easily develop Node.js and MongoDB Apps with Mongoose – Подробна информация за създаване на сървърна част и база данни с помощта на MongoDB & Express.

<https://scotch.io/tutorials/using-mongoosejs-in-node-js-and-mongodb-applications>