**Курсов проект**

**Дисциплина: Проектиране и интегриране на софтуерни системи**

**Фаза 2: Анализ на изискванията и проектиране на системата**

**Версия 1.x**

|  |  |  |
| --- | --- | --- |
| **Фак. №** | **Име на студент** | **Секция от документа** |
| 61917 | **Христо Христов** |  |
| 61954 | **Милен Дончев** |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Ноември, 2018**

**Съдържание**

[**ВЪВЕДЕНИЕ**](#_30j0zll) **4**

[Цел](#_1fob9te) 4

[Обхват](#_3znysh7) 4

[Резюме](#_2et92p0) 4

[Дефиниции и акроними](#_tyjcwt) 4

[**АНАЛИЗ НА ИЗИСКВАНИЯТА**](#_4z9om3rqw4sy) **5**

[Функционални изисквания](#_wf6y8eqsfl6d) 5

[Нефункционални изисквания](#_p1gr47k6n50) 6

[Изисквания към потребителския интерфейс](#_tfaf00lhey03) 6

[Производителност](#_v1wqx84i64ak) 6

[Наличност](#_g2emkln1f691) 6

[Сигурност](#_o15r5325tjo3) 7

[Скалируемост](#_q7secybyhh39) 7

[**СИСТЕМНА АРХИТЕКТУРА**](#_3dy6vkm) **8**

[Архитектурно проектиране](#_1t3h5sf) 8

[Клиенти](#_qzhy2fmqteqz) 8

[Уеб сървър](#_le0p20h5craf) 8

[База данни](#_yrocm2ux8qwo) 8

[Файлова система](#_ai1yq4tchrvd) 8

[Описание на декомпозицията](#_4d34og8) 8

[Обосновка на проектирането](#_2s8eyo1) 9

[**ДАННОВО ПРОЕКТИРАНЕ**](#_17dp8vu) **10**

[Описание на данните](#_3rdcrjn) 10

[Речник на данните](#_26in1rg) 12

[**КОМПОНЕНТНО ПРОЕКТИРАНЕ**](#_lnxbz9) **13**

[**ПРОЕКТИРАНЕ НА ПОТРЕБИТЕЛСКИЯ ИНТЕРФЕЙС**](#_35nkun2) **14**

[Обобщение на потребителския интерфейс](#_1ksv4uv) 14

[Екранни изображения](#_44sinio) 15

[**МАТРИЦА НА ИЗИСКВАНИЯТА**](#_2jxsxqh) **17**

[**ПРИЛОЖЕНИЯ**](#_z337ya) **18**

# ВЪВЕДЕНИЕ

## Цел

*Опишете на целта на документа.*

Настоящият документ описва дизайна на системата ParallelType. Той ще служи за създаване на системата от нулата на по- късен етап и описва нейните функционалности, както и някои имплементационни детайли. Създаден е за курса “Проектиране и интегриране на софтуерни системи”, зимен семестър 2018.

## Обхват

*Опишете обхвата на софтуерната система и изяснете целта и ползите от нея.*

Системата ще е достъпна за употреба от всеки, който има достъп до уеб браузър и интернет. Съответно, достъпът ще е ограничен за определени файлове в зависимост от акаунта, чрез който потребителят я достъпва.

Целта ѝ е да позволява на няколко потребители да работят върху даден документ едновременно, като всеки файл може и да има текуща и предишни версии, които могат лесно да се възстановяват.

## Резюме

*Създайте резюме на документа и опишете структурата му.*

Документът ще служи за създаване на системата от нулата. Структурата му е описана в съдържанието.

## Дефиниции и акроними

*Дефинирайте всички термини, понятия и акроними, използвани в документа.*

HTML- Hypertext Markup Language - декларативен език за описване на информация(текст).

# АНАЛИЗ НА ИЗИСКВАНИЯТА

*Опишете системните изисквания по начин, който подпомага процеса на проектиране, както и процеса на тестване на системата. Дефинирайте уникален номер и приоритет за всяко изискване.*

* *Пример за дефиниране на приоритети:*
* *Приоритет 1 – изискването е „задължително“*
* *Приоритет 2 – изискването е „необходимо“ за подобряване на софтуера*
* *Приоритет 3 – изискването е „препоръчително“*

*При описание на изискванията използвайте термините, с които сте дефинирали приоритетите. Например „Статусът на поръчката* ***задължително*** *приема стойност 0 или 1.“ Или „****Препоръчително*** *е потребителят да бъде уведомяван при изтичане на срока за плащане 3 дни предварително.“*

## Функционални изисквания

Примерен формат на таблица за описание на функционални изисквания.

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Изискване | Коментар | Приоритет |
| FR\_01 | Системата задължително трябва да записва файловете в БД така, че те да са достъпни само за дадени акаунти | Част от метаданните на даден файл трябва да са потребителите с достъп до файла, т.е. при промяната на правата се налага файлът да се запише наново. | 1 |
| FR\_02 | Потребителят задължително трябва да може да споделя достъпа до файла с други потребители на системата | Може да има 3 нива на достъп- скрито, read only и read/write. Ако не е споделен с даден потребител, файлът изобщо няма да е видим. | 1 |
| FR\_03 | Необходимо е системата да пази различни версии на файловете. | Това ще се осъществява чрез пазене на различните нови и променени сегменти на файла | 2 |
| FR\_04 | Системата трябва да поддържа едновременно писане в реално време на два или повече потребители, както и показването на променения текст в реално време | Може да се обновява всеки път, когато потребителят запише, или автоматично на даден интервал от време/ въведени символи. | 1 |
| FR\_05 | Препоръчително е системата да предлага статистика за добавения брой символи от всеки потребител | Това ще са част от метаданните за всеки потребител, който някога е участвал в проекта. | 3 |
| FR\_06 | Системата задължително да дава възможност за регистрация на потребители | Ще се изисква потребителско име, електронна поща и парола. Всеки акаунт ще има собствено ID, чрез което ще се представя в системата | 1 |
| FR\_07 | Препоръчително е потребителите да могат да си променят потребителското име и паролата | Това е безопасна операция, понеже системата използва ID-тата на потребителите, а не техните потребителски имена | 3 |
| FR\_08 | Задължително е потребителите да могат да свалят документа в текстов формат и HTML | HTML - за да може да се виждат различните стилове които са приложени в документа (описани в FR-10)  Текстов файл - само текст без стилизиране | 1 |
| FR\_09 | Задължително е системата да предоставя възможността за възстановяване на парола | Понеже системата изисква потребителят да е logged in за да ползва документите, това е критично изискване в случай, че забрави паролата си | 1 |
| FR\_10 | Редакторът да позволява различни промени по формата и изобразяването на текста | Тоест да имплементира функционалности от типа на различен размер на буквите, **bold**, *italic* и подобни. | 2 |
| FR\_11 | Системата е препоръчително да предлага опцията за добавяне на етикети към файл | Това ще направи възможна имплементацията на опростена търсачка | 3 |

## Нефункционални изисквания

*Изброените нефункционални изисквания са примерни и могат да бъдат модифицирани и/или разширени.*

### Изисквания към потребителския интерфейс

*Опишете характеристиките на потребителския интерфейс – формати, шаблони, структура на менюто, съобщения за грешка и др. съобщения, функционални клавиши и др.*

Интерфейсът трябва да е опростен, но и да предоставя достъп до пълната функционалност на системата.

### Производителност

Системата да може да поддържа едновременната работа на 10000 души. Производителността не би трябвало да се влияе от броя и размера на файловете, над които се работи, освен поради кеширане на сървъра.

### Наличност

Системата да е налична в 99.9% от времето, навсякъде по света с интернет връзка. Лесно подновяване на услугата при проблеми от техническо и друго естество.

*Базата данни е препоръчително да е на два различни сървъра (отделни машини). В случай на проблем с връзката на едната БД, да се пренасочат към другата. Също така ще има и два отделни файлови storage-а, където да се дублират промените. При срив на едната система, може лесно да се възстанови бекъп и да се продължи работата, докато през това време се отстраняват проблемите на първата система.*

### Сигурност

Задължително е паролата да бъде криптирана и след това хеширана. Възможност за възстановяване на парола чрез е-mail.

*По този начин при евентуален пробив, паролата да не може да бъде възстановена. Задължително е да не се пази в plain text, понеже много потребители използват еднакви пароли за пощи, социални мрежи и др. Потребителите ще имат възможност да си възстановяват паролата чрез e-mail на който ще бъде изпратен link със token, откъдето могат да си напишат нова парола. При възстановяването на парола е задължително да бъде проверено дали има потребител с такъв email за да може да се изпрати. За да се избегне последователна промяна на email и след това заявка за промяна на паролата, при промяната на профила ще се изисква текущата парола.*

### Скалируемост

При увеличаване на използването няма да се наблюдава забавяне.

*Всеки потребител ще работи върху дадено парче (директория/определени редове от релационната БД) от цялата система. По този начин се избягват изчакващи операции върху определени ресурси.*

# СИСТЕМНА АРХИТЕКТУРА

## Архитектурно проектиране

*Опишете модулната структура на системата и взаимовръзките между отделните модули от гледна точка на функционалността ѝ. Идентифицирайте подсистемите от високо ниво заедно с техните роли и отговорности. Опишете как подсистемите си взаимодействат. Създайте диаграма, представяща модулната архитектура и я коментирайте.*

### Клиенти

Уеб браузър който през javascript изпраща данни по HTTP към уеб сървъра. Document editor-a ще поддържа въвеждане на текст, редактиране и изтриване. Административната част ще поддържа управлението на потребителите които имат достъп до даден файл. Останалата част от системата ще поддържа преглед на файлове, статистики, потребители. Преди да се достъпва цялата функционалност, ще има форми за регистриране и логване.

### Уеб сървър

Приложение написано на ASP.NET Web API, които ще приема заявките от клиентите. Сървъра ще поддържа authentication и authorization с cookie. Ще получава заявките от клиентите и ще извършва съответните операции върху данните.

### База данни

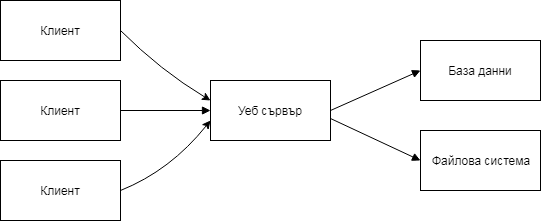
Базата данни ще бъде достъпвана от уеб сървъра. В нея ще се съдържат пътищата към файловете.

### Файлова система

Там ще се съдържат самите файлове. Ще бъде достъпна от уеб сървъра, който ще добавя, редактира и изтрива файлове.

## Описание на декомпозицията

*Декомпозирайте подсистемите, идентифицирани при архитектурното проектиране. Осигурете функционално или обектно-ориентирани описание на системните компоненти. При функционалното описание използвайте диаграми на данновия поток (data flow diagrams) от високо ниво и структурни декомпозиционни диаграми. При обектно-ориентирани описание осигурете модел на подсистемите, обектни диаграми, обобщаващи йерархични диаграми (generalization  hierarchy  diagrams), агрегиращи йерархични диаграми (aggregation  hierarchy diagrams), интерфейсни спецификации и др.*

**

## Обосновка на проектирането

*Обосновете взетите архитектурни решения в Секция 2.1.*

Архитектурата трябва да е максимално опростена, за да може едновременно да се поддържа лесно и съответно да върши необходимата работа.

# ДАННОВО ПРОЕКТИРАНЕ

## Описание на данните

*Опишете как информационният домейн на системата се трансформира в даннови структури. Изяснете как данните се съхраняват, организират и обработват.*

Данните ще се съхраняват в релационна база данни (например SQL Server). Ще бъдат обработвани от система (използваща уеб услуги). Данновите структури са организирани по следният начин:

Таблици: **Files**, **Users**, **PermissionTypes**, **UsersFiles** (връзка между първите 3), **FilesVersions**, **Statistics**

Таблица **Users** (потребители на системата)

Могат да се регистрират с username и password, или вход с Google и Facebook

- UserId - уникален идентификатор

- Username

- Password (налична когато се е регистрирал през системата)

- Email - в случая че си забрави паролата

- ExternalLogin (Google, Facebook)

- Друга информация свързана с потребителите

Таблица **Files** (данните за файловете)

- създател: CreatorId (има права да изтрие документа и историята му, както и да определя кой какъв достъп има до файла)

- име: за визуализиране в браузъра (може да бъде променяно)

- guid: за вътрешно запазване на сървъра(не се вижда през потребителски интерфейс)

- видимо за четене от всички: true/false

- видимо за писане от всички: true/false

Таблица за тип достъп до файл **PermissionTypes**

- Read, Write, ReadAndWrite

Таблица за връзка между потребители и файлове (**UsersFiles**):

- потребител (UserId)

- файл (FileId)

- тип достъп: (PermissionTypeId)

Когато полето за видимо за всички е false, то данните за кой потребител какъв документ може да достъпва ще бъдат попълвани в тази таблица. Съответно ако се разреши достъпа за четене от всички, то ще бъдат изтрити редовете където са свързани с четене на този файл.

Понеже не може и да има определени потребители които да го достъпват и всички едновременно.

Аналогично и за писане.

Таблица **FilesVersions**

- файл (FileId)

- директория (път до директорията на документа, името на файла е guid + version)

- версия (пореден номер)

- промени (информация за въведен/изтрит текст и позиция във файла)

Два начина на подход за запазването на данните за файловете.

Списък със промени (текст и позиция) - Когато промените станат прекалено много, за да се визуализира последната версия на файла ще отнема много изчисления.

При всяка промяна - нов файл. Когато промените станат прекалено много един и същ файл от 5 MB, ще има много версии на файловата система и ще заема много памет.

**Уравновесен вариант**:

В таблицата **FilesVersions** ще има колонка за нови промени по дадена версия на документ.

Тоест когато версия 1 достигне до 20 промени (ckeckpoint), файлът ще бъде записан във файловата система и ще бъде добавен ред със новата версия и колонката "промени" ще бъде празна. По този начин версията се "завършва".

При опит за виждане на по стара версия, ще се открие между кои 2 записа се намира промяната и ще бъдат приложени съответните промени бързо и ефективно.

Таблица **Statistics**

- UserId (потребител)

- AddedSymbols (добавени символи)

- RemovedSymbols (премахнати символи)

## Речник на данните

*Опишете същността и типовете на данновите елементи на системата. Ако сте използвали функционално описание в Секция 2.2, то дефинирайте всички функции, заедно с техните параметри. Ако сте използвали обектно-ориентирано описание, то дефинирайте обектите, техните атрибути, методи и параметри на методи.*

# КОМПОНЕНТНО ПРОЕКТИРАНЕ

*Опишете по систематичен начин, ако е релевантно, съществените компоненти на системата. Приложете алгоритми и/или псевдокод.*

PushChange(int fileId, string text)

{

if change == 20

insert into FileVersion (file, “”)

else

update last version (file, change + text)

}

AddRole(fileId, userId, role)

{

if currentUserId == file.creatorId && not exists in FilesUsers(fileId, userId, role)

{

insert into FilesUsers(fileId, userId, role)

}

}

DeleteFile(file)

{

if currentUserId == file.creatorId

{

file = getFile(fileId);

delete from FilesUsers(fileId)

delete from FileVersions(fileId)

delete from Files(fileId)

// Премахва от файловата система всички версии на файла

System.IO.Remove(file.paths)

}

}

# ПРОЕКТИРАНЕ НА ПОТРЕБИТЕЛСКИЯ ИНТЕРФЕЙС

## Обобщение на потребителския интерфейс

*Опишете функционалността на системата от потребителска гледна точка. Изяснете как потребителят ще използва функционалността на системата и механизмите за обратна връзка към него.*

Интерфейсът ще е опростен и разделен на три вертикални части.

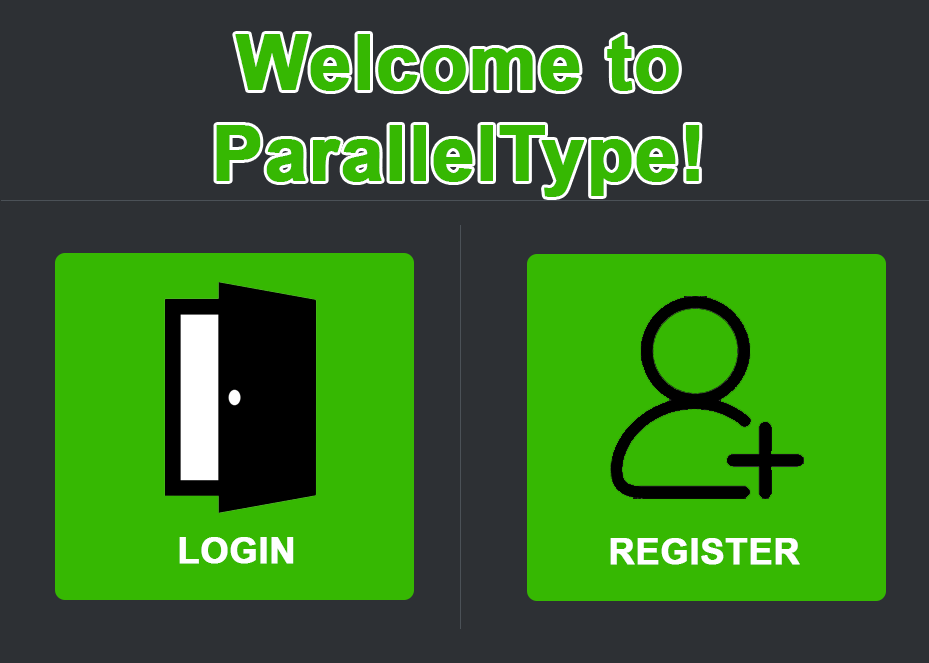
Лявата третина ще е с индекси на предишни файлове за бърз достъп, както и търсачка, която търси файловете по зададени тагове.

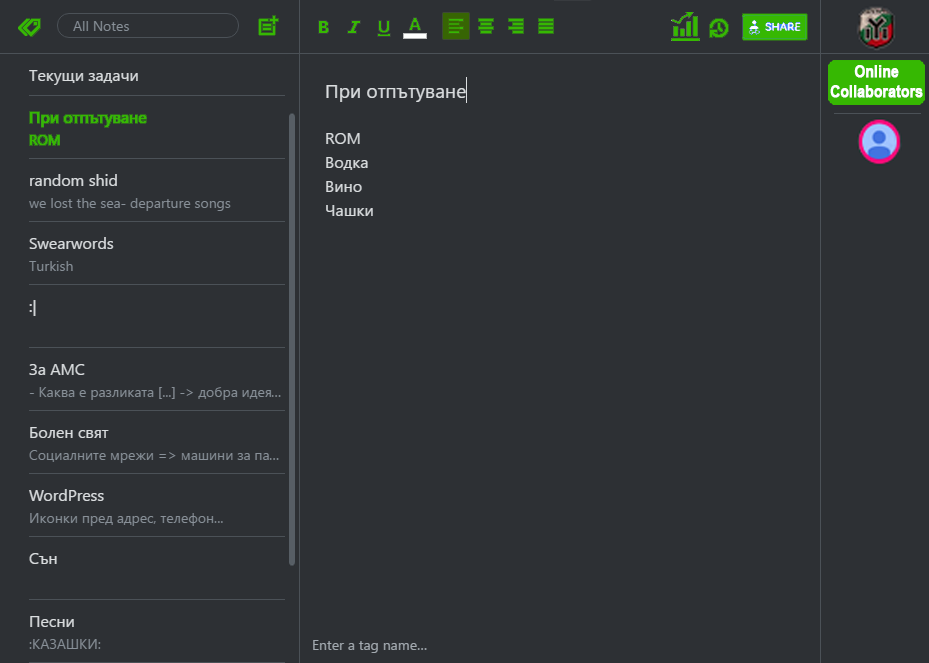
Втората третина ще има textarea по средата. Над нея ще има меню с бутони за манипулиране на документа. Отдясно на тези бутони ще са бутоните за извеждане статистика на файла, предишни версии и споделяне. На дъното е опционалното поле за добавяне на таг към файла.

Третата третина ще е за информация свързана с потребителите- Най- горе вдясно ще е профилът на текущия потребител, а под него- профилите на хората, които са онлайн и имат достъп до файла.

Ако потребителят не е вписан, ще има опростен начален екран за login или регистрация.

## Екранни изображения

*Приложете изображения, показващи потребителския интерфейс. Не се изискват реални екранни изображение (screenshots) от системата, но ако проектът е в по-напреднала фаза, могат да се приложат*

**

# МАТРИЦА НА ИЗИСКВАНИЯТА

*Създайте матрица на съответствието между изискванията към системата, дефинирани в спецификацията на изискванията, и компонентите и данновите структури, описани в настоящия документ. Използвайте табличен формат, за да покажете кои системни компоненти удовлетворяват функционалните изисквания към системата. Реферирайте функционалните изисквания по кодовете, зададени в спецификацията на изискванията (FR1.01, FR1.02, ...).*

# ПРИЛОЖЕНИЯ

*Тази секция не е задължителна и се използва при необходимост.*

*Приложенията се включват директно или се реферират. Те подпомагат разбирането на системното проектиране, изложено в настоящия документ.*

Facebook/ Google+ login