***НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ***

*CSCB024 Практика по програмиране и интернет технологии*

**Курсова работа:**

Система за управление на задачите

**Изготвил:**

Йордан Милков Костадинов, F76814

учебна година 2017/2018

Летен семестър

1. **Предназначение:**

Системата има за цел да улесни разпределението, изпълнението и проследяването на разнообразни задачи от ежедневието като отстраняване на повреди, изготвяне на списъци за пазар, грижа за дома и други дейности. Уеб сайтът дава възможност за обединяване на едно място сходни задачи, като се разпределят по групи. Така потребителите на групите лесно могат да се ориентират в своите задачи и а останалите могат да следят прогреса.

1. **Обща информация:**

Концепцията на системата разделя потребителите и задачите в групи с различна тематика. Всяка една от задачите е уникална и принадлежи на някоя от групите. Потребителите могат да членуват в една или повече групи, като това им дава достъп до всички задачи. Всяка група и задачите в нея са невидими за останалите потребители. Страниците и групите могат да бъдат променяни от всеки с достъп до тях.

Системата е имплементирана като модерен сайт, подходящ за визуализация на всички видове устройства и браузъри. Състои се от 3 компонента – клиентски интерфейс, сървърна част и база от данни. За разработката на клиентската част е използван Aurelia фреймуърк, който е базиран на програмния език TypeScript. Предимството му е, че той е обектно-ориентиран и се компилира до JavaScript. Комуникацията със сървъра се осъществява чрез PHP заявки, които се изпълняват отдалечено. Сървърът е отговорен за достъпа и модифицирането на базата данни. Системата е инсталирана на Raspbian операционна система с инсталиран Apache Server. Той осигурява зараждането на страниците, както и изпълнението PHP и SQL скриптовете. То има и интегрирана база данни, базирана на езика MySQL за реализиране на релативна база данни. Уеб приложението е достъпно онлайн на адрес: <http://78.90.20.71:27010/> .

1. **Потребителски интерфейс:**
   1. *Страница за вход:*

При отваряне на сайта се появява страница за изчакване докато не приключи процеса по инициализиране. Когато той приключи, при първо посещение се визуализира страница за вход и регистрация. Тя съдържа полета за потребителско име и парола. При натискане на бутона Вход се прави валидация на полетата. Ако данните са неверни, на потребителя се визуализира съобщение за грешка с описание на проблема, а системата остава на страницата за вход. Ако данните са валидни, връзката е успешна, а потребителят е вписан в системата. Тогава сайтът преминава към началния изглед. За улеснение на потребителя, временни данни се запазват на машина му и не изискват повторно въвеждане на парола. За нови потребители съществува поле, даващо възможност за избор на потребително име. При натискане на бутона Регистриране, въведено потребителското име се проверява дали е уникално. Ако такъв потребител вече съществува се появява съобщение за грешката. При успешен избор, потребителят е отведен в панела за регистрация.

* 1. *Страница за регистриране:*

Заглавието на страницата за регистриране съдържа вече одобреното потребителско име. Следват полета за публично име, с което потребителят ще се визуализира в групите и задачите, както и имейл, и телефон. В последните две полета потребителят трябва да въведе паролата си и да я повтори за валидация. За излизане от страницата, в края на формата се намират бутоните за приключване и отказ от регистрация. Отказът връща обратно в страницата за вход. При завършване на регистрацията, ако има проблеми, съобщение за грешка спомага за отстраняването им. Когато регистрацията е успешна, потребителят е прехвърлен в началната страница на новия профил.

* 1. *Поле за навигация:*

Когато започне потребителска сесия, в горната част на всички страници се появява полето за навигация, което съдържа няколко менюта. Първото меню отвежда към началната страница на сайта. Следва менюто Профил, което отвежда към страницата за преглед и редактиране на потребителски данни и парола. Последното меню приключва сесията, данните за потребителя се изчистват и се визуализира страницата за вход. Полето за навигация в този случай изчезва отново.

* 1. *Страница за потребител и редактиране:*

Страницата на потребителя съдържа списък с всички видими данни за него. Това са публично и потребителско име, имейл и телефон. Бутон за редактиране на данните е наличен до заглавието. При натискането му, потребителят е отведен във форма, идентична с тази за регистрация, попълнена с наличните данни. Разликите са в това, че въвеждането на пароли не е задължително за промяна на останалите данни, както и че за смяна на парола се изисква и текущата в допълнително поле. Бутона за приключване на редактирането валидира данните, съобщава за грешки, а ако няма, навигира обратно до обновената потребителска страница. При отказ, потребителят е препратен до неговата страница, но всички въведени промени са анулирани.

* 1. *Начална страница:*

В началната страница на потребителя се визуализира лист с всички групи, към които той е включен. За удобство до името на групата е поместено описанието ѝ. Групите са мястото, където се записват задачите. Всяка група има уникален код и име, както и един или повече членове. При избиране чрез клик на група от списъка се отваря изглед за съответната група. До бутона на групата има и бутон за редактиране на група. Той визуализира страница, даваща възможност за промяна на данните и потребителите. В края на списъка с групи се намира бутон, даващ възможност и за добавяне на нови групи от задачи. При клик се отваря страница, почти идентична с тази за редактиране.

* 1. *Страница за добавяне и редактиране на група*

Страницата за добавяне и редактиране на група съдържа полета с всички данни за групата, като предлага интуитивни методи за промяна. Първо е текстово поле за промяна, което съдържа името на групата. Следва аналогично поле с описанието. В списък се визуализират и потребителите, които имат достъп до групата. По подразбиране за нова група като единствен потребител е вписан създаващия я. До всеки един от членовете има бутон за премахването му от групата. При натискането му се поява диалог за потвърждение на операцията. Нови членове се добавят от контрола с автоматични предложения. За да бъде добавен нов член в групата, името му трябва да бъде избрано от списъка на контролата, които съдържа всички регистрирани потребители. След това се натиска бутона за добавяне и новият потребител се появява в списъка по-горе. На края на страницата са бутоните за прилагане и отказ от промените. При отказ от промени, сайтът препраща към началната страница. При прилагане на промените се прави валидация дали групата има непразно заглавие и поне един потребител. При грешка, излиза съобщение с повече информация. Ако данните са валидни, промените се записват и следва автоматично препращане към страницата на групата, която е празна при създаване.

* 1. *Страница на група*

Страница на група съдържа лист с всички активни и затворени задачи. Изгледа е структуриран в табличен вид, а заглавният ред съдържа етикет за всяка колона. При натискането на някой от тях, таблицата се сортира спрямо стойността в избраната колона. Аналогично на списъка на групите, за всяка задача се визуализират името и описанието ѝ. Следва бутон с флаг, който обозначава дали задача е приключена. В последната колона на списъка се намира бутон, позволяващ редактирането на съответната задача. Потребителят може да добавя и нови задачи, като използва бутонът за това в края на списъка. При избор на някоя от задачите в списъка чрез клик, сайтът преминава към страница на задача. При натискане на бутона с флаг се променя статуса на задачата. При избор на бутон за редактиране или добавяне на задача, на потребителя се визуализира страница за редактиране, аналогична на тази на задачата.

* 1. *Страница на задача:*

Страницата на задача показва цялата информация свързана с нея. Всяка задача има уникален номер, заглавие и описание. Задачите имат и статус – Възложена и Приключена. Ако задачата е възложена, до заглавието ѝ е визуализиран бутон с съответната икона, който дава възможност задачата да бъде маркирана като приключена. Ако задачата вече е приключена, всеки от членовете на групата може да я възложи наново с бутона, чиято икона е променена. Следва бутона Редактирай, който дава възможност на членовете на групата да отидат в изгледа за промяна на задача. Под името на задачата е полето с описанието ѝ. Следва списъка с изпълнителите на задачата. Това са потребителите, отговорни за приключване на задачата. Последния елемент на страницата е незадължителен списък с подзадачи за изпълнение. Всеки от елементите в списъка има чекбокс, който показва дали подзадачата е приключена. При клик от потребителя върху елемент, полето със статуса се променя. Нови подзадачи могат да се добавят от формата за писане под списъка. В нея се въвежда кратко описание и при клик на бутона „+“ вдясно, подзадачата се добавя към списъка.

* 1. *Страница за редактиране на задача:*

Страницата за редактиране служи за промяна на полетата на задачата. Той много наподобява изгледа на задачата, като част от полетата са променени. Името и описанието на задачата се визуализират като текст за редактиране. Там потребителя може да смени съответните стойности. Следва списъкът на изпълнителите. На края на реда, след името на всеки от потребителите е добавен бутон, който дава възможност за премахване на изпълнителя след потвърждение през диалог. От поле с автоматични предложения под листа могат се добавят и други изпълнители от групата. В него, аналогично на страницата за редактиране на група, потребителя може да въведе име на друг член от групата. Появява се списък с възможни потребители, от които лесно и сигурно може да се избере и да запълни автоматично полето. Новият изпълнител се добавя към списъка с бутона „+”. След това и той може да бъде премахнат. Ако избраният член на групата вече съществува в списъка или е въведен невалиден потребител, полето се изчиства без допълнителни промени. Следва листът с подзадачите. Описанието на всяка може да бъде редактирано. До нея е добавен бутон, който подобно на списъка с изпълнителите, изтрива подзадачата. Нови подзадачи не могат да бъдат добавяни. Отново има бутони за прилагане и отказ от промените. Когато промените трябва да бъдат запазени, а редактирането да приключи, потребителят избира бутона за запис. Прави се валидация за име на задачата както и за поне един изпълнител. При настъпване на грешка, съобщение информира потребителя за проблема. Ако всичко е наред, визуализира се изгледа на задачата с обновени данни. При натискане на бутона за отказ, редактирането приключва без да запази промените и отново препраща към страницата на задачата.

* 1. *Страница за създаване на нова задача:*

Страница за създаване на нова задача е идентична с тази за редактиране. Подзадачи не могат да се добавят при създаването на задача. За да приключи успешно операцията по добавяне, задачата трябва да има въведен поне 1 потребител в списъка с изпълнители. Отново ако има проблем, съобщение напътства потребителя. При успешно добавяне на задачата, сайта навигира към основната ѝ страница. Ако се избере бутона за отказ, потребителят бива пренасочен към страницата на групата, към която е трябвало да принадлежи новата задача.

1. **Софтуерни спецификации:**
2. *Клиентска част:*

Системата е разработена като уеб-базирано приложение. Предназначена е за използване на широк набор от браузъри на различни операционни системи и платформи. Подходящ за визуализация на устройства с различни резолюции и пропорции на екрана. Тъй като е базиран на Aurelia фреймуърк, сайтът изисква браузър имплементиращ JavaScript команди и HTML5 поддръжка, каквато имат всички широко използвани модерни браузъри – Chrome, Mozila Firefox, Edge, Safari, Opera и други. Приложението изисква постоянен достъп до интернет за връзка със сървъра.

1. *Сървърна част:*

Сървърната част изисква устройство, поддържащо уеб-сървър приложение, поддържащо HTTP заявки, както и изпълняващо команди на скриптовия език PHP. Също така, за да се осъществява комуникацията, устройството трябва да бъде разрешено за откриване в избраната за ползване мрежа.

1. *База данни:*

За реализиране на базата данни е избран релативен модел. Изисква се системата за бази данни да работи с MySQL заявки за достъп и модифициране. Връзката с базата се осъществява от сървърното приложение.

1. **Използвани технологии:**
2. *Клиентска част:*

За интегрирана среда за разработка е използвано Visual Studio Code. Предимствата му са, че той поддържа Type Script като език за разработка. Има вградени команди, които дават достъп до компилатора и създават виртуален сървър, който подпомага разработката. В него е интегрирана и функционалност за автоматични подсказки (autocomplete), както и дава информация за грешки в кода преди компилиране (intellisense).

За реализацията е избран програмният език TypeScript. Той е обектноориентиран език, надмножество на JavaScript. Има значителни предимства пред стандартните процедурни методи за писане на уеб-базирани системи – типизация, наследяване, използване на библиотеки. Езика се компилира до JavaScript, за да може да бъде интерпретиран без проблем от браузъри и други системи.

За настройка на работната среда и инсталиране на JavaScript пакети е използван npm (Node.js Package Manager). Това е софтуерно приложение, което дава възможност за инсталиране на необходимите библиотеки за реализирането на уеб портала. Оперира като конзолно приложение с набор от команди.

След инсталирането му, следва да се инсталира пакета, отговорен за динамичното управление на версиите на пакетите, от които има нужда. Това става като се използвайки командата npm install jspm -g. Това приложение автоматично конфигурира версиите на различните библиотеки спрямо записаната информация в package.json файла в основната директория на системата. Също така това е приложението, което се грижи да инсталира и необходимата документация за автоматичните подсказки и поправка на грешки.

Последния необходим елемент, който трябва да бъде инсталиран, осигурява компилирането и дистрибуцията на системата - gulp.js: npm install gulp -g. Това е библиотека с инструменти, която подпомага създаването на задачи, които автоматизират компилирането и пакетирането на системата. Конкретни команди могат да бъдат открити в директорията build/tasks, сред които build.js за компилиране, bundle.js за пакетиране, watch.js за автоматично презареждане на страницата при промяна на изходния код.

Системата е базирана на шаблона с отворен код aurelia-skeleton-navigation (typescript). Той е създаден изцяло на Aurelia фреймуърк, нов и модерен, разработен на TypeScript и с отворен код. Също така, Aurelia е модулно базиран и предоставя възможност за инсталиране на допълни разширения на функционалността. Силата на този фреймуърк е в свързването на данните от логическия модел с елементите на HTML страницата, като предоставя двустранна връзка. Разширява и възможностите на HTML кода с нови тагове, които се обработват при изпълнение и дават възможност за динамичното му променяне. За да изглежда добре на всякакъв вид устройства, Aurelia разчита на Bootstrap – най-популярният HTML и CSS фреймурк за създаване на риспонсив интерфейс.

Шаблонът Sceleton-Navigation предоставя основната организация на едно уеб базирано приложение изградено на Aurelia. Той съдържа файлът package.json, чрез който използвайки командата jspm install изтегля и конфигурира необходимото за първа компилация на шаблона. При отваряне на папката като Visual Studio Code проект, трябва да се изпълни командата Build. Използвайки gulp, изходният код се компилира на приложението до чист HTML и JavaScript файлове. Watch е командата, автоматично извиква задачата Build при промяна в изходния код. Също така инициализира виртуален сървър за отваряне на сайта, като и се грижи автоматично да презарежда страницата на браузъра, когато приложението е било прекомпилирано.

От основния файл main.ts, в configure функцията започва изпълнението на уеб приложението. Това се случва при първоначално зареждане на страницата или при презареждането ѝ. След това на потребителя се визуализира заявената страница. От шаблона има на разположение меню за навигация. То е имплементирано от инстанция на класа Router, който Aurelia предоставя. Този клас се грижи за конфигурирането и отварянето на подстраниците, като и за историята им в браузъра.

В шаблона се откриват и няколко примерни страници. Всяка страница или сложен компонент от изгледа се състои от две части: HTML документ представляващ темплейт и файл с логиката на компонента. В HTML документа е структурата на страницата, която се визуализира – чрез тагове се създават различните видове контроли, като Aurelia предоставя и нови тагове за специални елементи, както и функция за итериране на данни от модела и динамичното им попълване. Информацията, която се визуализира в контролите, се обработва от логическата част. Чрез CSS всеки от елементите се стилизира, като основните класове са предоставени от библиотеката Bootstrap.

Файлът с логиката на страницата или съставния компонент започва с необходимите включвания на различни библиотеки. Обикновено следва дефиниция на клас, чието име съвпада с името на HTML темплейта. Всяка от отделните се извиква през Router инстанцията, която трябва да е една и съща. Както във всеки обектно ориентиран език, в този момент се извиква конструктора, за да се инициализират данните. Получават се и различни събития, като при дефиниране на съответстващата функция, то те ще бъдат обработени спрямо имплементацията ѝ. Такива са функциите „attached“, която се извиква когато HTML темплейта е закачен за изгледа, и функцията „activate“, която се извиква при активиране на страницата. Това е моментът, в който данните се визуализират в свързаните елементи, обикновено това са член данни на класа. При динамична промяна на данните на класа, фреймуъркът Aurelia се грижи те автоматично да се променят и в потребителския интерфейс.

За реализирането на някои по-сложни компоненти на сайта, са използвани допълнителни разширения на Aurelia – Plug-ins. Такива са плъгините aurelia-dialog, използван за реализиране на диалозите за потвърждение, и aurelia-autocomplete, за контролите с автоматични предложения. За да се използва плъгин, то той първо трябва да бъде инсталиран през npm или изтеглен от github. За да бъде конфигуриран за употреба, той се добавя в регистрацията в Configure функцията в основния файл main.ts.

1. *Сървърна част:*

Сървърната част отговоря за комуникацията между клиентската част и базата данни. За разработването на сървърната част е използван скриптовия език PHP. Това е познат и широко използван език за реализиране на сървър. Уеб приложението използва заявки, чрез които се осъществява комуникацията. Ако заявката е коректна, се изпълнява съответната операция върху базата данни и се връща резултат на клиента, ако е заявено.

Apache HTTP Server се използва за установяване на връзката с клиента. За интерпретатор на PHP заявките е използвана системата предоставяна от Apache. Тя е с безплатен лиценз и е достъпна на повечето познати платформи. Използвана е версия 2.4, която включва PHP версия 5.6.30.

1. *База данни:*

Осигуряването на MySQL релационната база данни също е реализирано с Apache. В този пакет е включена система за бази данни, която е безплатна. Администрирането на база данни е улеснено от модула phpMyAdmin, който предоставя интерактивен интерфейс за моделиране и промяна на данните.

Данните се съхраняват в база от пет таблици:

1. users – Таблица с регистрираните потребители и техните данни. За ключ се използва потребителското име.
2. groups – Таблица с информацията за всички групи – име, описание и уникален номер, който е и ключ на таблицата.
3. groupusers – Тази таблица има само две колони, които образуват съставен ключ. Тя се използва за връзка между групите и потребителите, които членуват в нея. Всеки ред съдържа ключ от съответната таблица.
4. issues – Таблица с информацията на всяка една от задачите – заглавие, описание, статус, уникален номер на задачата за ключ, уникален номер на групата към която принадлежи и текстови списък с ключове на отговорните потребители разделени със запетая.
5. checks – Всеки ред на тази таблица съдържа информацията на подзадача – уникален номер, номер на задачата от която е част, описание и статус.
6. **Контрол на версиите:**

За контрол на версиите е използван GitHub.com. Това е онлайн система, базирана на Git и е широко използвана. По-голямата част от използваните технологии и библиотеки се съхраняват и разработват на GitHub. Системата е безплатна за употреба за проекти с отворен код. Адресът на този проект е: *https://github.com/yordan-milkov/TaskBo-rd*

1. **Възможности за подобрения**:

Приложението може да бъде разширено като му се добавят още функционалности, които да улеснят потребителя. Добавяне на краен срок за изпълнение на задачата. Добавяне на място за коментари и списък с последните активности по задачите в групата.

1. **Източници:**
2. <http://aurelia.io/training.html> - Официален сайт на фреймуърка Aureliea. Страница с документация и обучения.
3. <https://www.typescriptlang.org/docs/home.html> - Официален сайт на езика Typescript. Страница с документацията на езика.
4. <https://www.w3schools.com/> - Сайт за обучения по различни езици и технологии за уеб програмиране. Справочник за използване на PHP, MySQL, HTML и CSS.
5. <https://www.stackoverflow.com/> - Универсален източник на информация и съвети, примерни програмни подходи и решения на проблеми.