***Документация****Crypto-Chat, Антон Чолаков, wd\_anch0113*

**Кратко Въведение:**“Crypto-Chat” се оказа едно много интересно задание, което ме накара да използвам освен усвоените вече методи за работа, и куп други (криптиране, работа с голям брой заявки, стартиране и рестартиране на таймери и т.н.). В крайна сметка, след доста четене и редактиране, мисля, че мога да се похваля с едно добре работещо и приятно изглеждащо приложение.

**Описание на дейността:**Започнах, разбира се, с преглед на цялостната „картина” на уеб-чата – функционалностите, най-добрия начин за начало и последователност на действията.

1. Уеб дизайн (основни линии) – Първата ми задача бе да оформя в приличен вид html страницата (всички div-ове, скрити div-box-ове за допълнителна информация, манипулирани чрез скриптове, нужните полета), за да мога да се заема с функциите на чата. В края направих последните промени и „доизкусурявания” по уеб-дизайна, който е авторски – разработен от мен (PSD файлът е качен в архива).
2. CSS файл – основно float-ване на елементи, за да бъде страницата ‘dev-friendly’ и да бъде улеснена работата по цялостния дизайн и писане на скриптове в последствие.
3. Функции (със сигурност най-трудната част от цялостния проект) – бих могъл да опиша всяка една поотделно, но не би имало смисъл, тъй като част от тях са съвсем елементарни, затова ще разгледам по-сложните и трудно за изпълнение, свързани директно с функционалностите на проекта:

* Във файлът с функциите, още преди началото на същинските функции присъства service url-a и два глобални обекта съхраняващи информация, за по-нататъшното й достъпване и използване
* Стартиране на таймера – извършва се в функция, в която първо се рестартира таймерът, ако е стартиран, и след това се започва/задава отначало. „Свързан” е с по-значими събития от страна на юзъра и е с цел – логаут при изтичане на 5 минути, в които юзърът е неактивен.
* Функции, които се използват главно за генериране на html съдържание, тъй като съм разработил проекта си на една страница, с динамично генериране чрез javascript
* Функции при натискане на Login и Register бутони – те са сравнително сходни – вземат информацията от текстовите полета, извършват проверки за валидност (като по този начин се избягват излишни заявки към сървъра), криптират се името и паролата и се извършва съответната post заявка.
* Функция за логаут – една от най-лесните заявки, при която (при успех) се връща първоначалния изглед на страницата.
* Взимане на всички онлайн потребители – Използвайки масивът от потребители, който се връща от сървъра, добавих всеки един елемент към ‘li’, изведени в специален бокс. Като ‘label’ на бокса изписвам общия брой на онлайн потребителите, взимайки големината на масива.
* При клик върху който и да е юзър се зарежда функция за изпращане на покана. Ако се опитате да изпратите покана на себе си, сървърът ще върне грешка със съответния текст, визуализиран в специален бокс за информация от сървъра (те са 2, единия от които е централен за приемане на покани и вадене на грешки, а другият се използва за изписване на това кой потребител е онлайн, офлайн или отказва поканата ви и т.н.).
* При функцията за същинската покана се въвежда ключ за свързване, който криптира случайно генерирано рандъм число от 1 до 999999999. При успех, потребителят получава съобщение, че поканата му е изпратена (между временно таймерът се нулира).
* Поканеният потребител трябва да въведе ключа за достъп: при правилен ключ се стартира чат сесия, а при грешен ключ, който се проверява чрез декриптиране, се вади съобщение за грешен ключ.
* Междувременно, крипто-чат-потребителят получава съобщения от сървъра. Сравнявайки msgType-а на получените от него данни, се изпълняват различни операции. По специална е Challenge операцията, свързана със започването на старт сесия и по-точно поканата за нея. Точно чрез този msgType се зарежда същинската функция, описана по-горе. Друга важна част от тази функция е получаването на чат съобщения. Те се взимат от сървъра, декриптират се и се изписват в чат-бокса.
* Отказ от чат – може да се извърши при изпращане на покана, отговор на покана или по време на чат сесия. След успех на тази функция, другия потребител получава известие, че {username} е отказал поканата за чат или е затворил чат сесията.
* Изпращане на чат съобщения – същинската част е криптирането, а след това (при успех) и поставянето на съобщението в чат-бокса.
* Функции като onError, perfromGetRequest и performPostRequest ‘завършват’ файла. Използвани са, за да се направи кода като цяло по-четим и по-лесно достъпен.

1. Цялата тази трета част е свързана постепенна променя на css и html-a, за да ‘паснат’ на замислените идеи. Добавяне на Gibberish AES, Sha1 и Jquery са други части от проекта, които направих по време на разработката на функциите.
2. PSD файл – след като функционалността на цялото приложение бе готова, довърших първоначалния модел на уеб-дизайна.
3. Проверка на кода за грешки, лошо именуване и др. Важно е кодът да бъде добре четим и отделих достатъчно време да го направя такъв.

**Допълнение:**Из кодът, естествено, има доста други дребни елементи, които не споменах горе (Show, Fadeout, Fadein анимации и др.).

Приложението е съвместимо с Google Chrome, Mozilla Firefox, Opera, Safari, Internet Explorer.

Относно по-старите версии – поддръжка на всички функции от IE8 и IE7, където обаче дизайнът до голяма степен се „чупи”.

**Заключение:** В крайна сметка, не представляваше голяма трудност да се направи функционалността на проекта със знанията, които придобих в изминалите 9-10 месеца. Беше наистина интересна задача, а всяко нещо, което направиш сам доставя страхотно удоволствие!

*Offtopic: Обучението на Telerik Academy бе наистина прекрасно и вложих максимално усилия с много желание. Надявам се продуктът на моя труд да бъде достатъчен, за да стигна до така желания сертификат! Благодаря Ви, Телерик ☺*