Документация

Версия 1.x

Съдържание

[1 Въведение 2](#_Toc57320861)

[1.1 Цел 2](#_Toc57320862)

[1.2 Резюме 2](#_Toc57320863)

[1.3 Дефиниции и акроними 3](#_Toc57320864)

[2 Използвани технологии 4](#_Toc57320865)

[3 Реализация на базата от данни 6](#_Toc57320866)

[4 Реализация на бизнес логиката 8](#_Toc57320867)

[5 Реализация на потребителския интерфейс 10](#_Toc57320868)

[6 Внедряване на системата 15](#_Toc57320869)

[7 Идеи за възможни подобрения 16](#_Toc57320870)

# Въведение

## Цел

Настоящият документ описва имплементацията на системата TravelShare. В него е налична подробна информация за всички техники, които са използвани при създаването на проекта( какви технологии са използвани, описание на това как са реализирани), изображения от потребителските интерфейси и подробно обяснение на всяка една от функционалностите, които се свързват с дадения и интерфейс и как да бъдат достъпени те, някои основни идеи за бъдещо развитие и подобрение на проекта, които на този етап не са налични. Освен запознаване с начина на реализиране на проекта и неговия визуален интерфейс, документа има за цел да покаже как платформата да бъде внедрена на дадено устройство.

## Резюме

Документът e разделен е на следните няколко секции:

1. **Въведение**

Обяснява целта на документа и коя част от него какво описва и къде се намира.

1. **Използвани технологии**

Описва технологиите и програмните средства, използвани за реализация на системата.

1. **Реализация на базата от данни**

Описва типа и структурата на базата от данни.

1. **Реализация на бизнес логиката**

Описва реализацията на бизнес логиката на системата с избраните технологии и програмни средства.

1. **Реализация на потребителския интерфейс**

Описва реализираните елементи на потребителския интерфейс и различните изгледи в различните случаи. Има прикачено кратко демо под формата на клип в папката на проекта озаглавен Demo, а тук в документа има добавени изображения и описания на всяко едно от тях за визуализиране на дизайна.

1. **Внедряване на системата**

Описва изискванията и последователността от действия за внедряване на системата.

1. **Идеи за възможни подобрения**

Описва всички идеи, с които системата може да бъде подобрена, които са възникнали в различни етапи по време на създаването и реализирането на проекта, но не са били внедрени, поради липса на достатъчно време.

## Дефиниции и акроними

* Rest API - уеб приложение, което използва принципите на HTTP и REST.
* HTTP - Протокол за пренос на хипертекст.
* REST - е стил софтуерна архитектура за реализация на уеб услуги. Основната идея е да се определи системен ресурс, който се променя в резултат на взаимодействието между доставчика на услуги и потребителя. Архитектурният модел REST включва взаимодействията между сървър и клиент, осъществени по време на трансфера на данни.
* Потребителски интерфейс - интерфейс, при който елементите, предоставени нa потребителя за управление, са изпълнени във вид на графични изображения (менюта, бутони, списъци и др.).
* Разпределена софтуерна архитектура - архитектура, в която всеки отделен модул е отделна самостоятелна единица.
* Модул – логически обособена софтуерна единица
* Уеб приложение - приложение , до което потребителите имат достъп през мрежа като Интернет.
* Потребител – човек, който използва компютърна или мрежова услуга.
* Регистрация – създаване на профил в системата от страна на потребител – въвежда лични данни като потребителско име и парола, с които да ползва системата (уникален потребителски профил на всеки потребител – така че всеки има достъп само до своята лична информация, но не и на останалите потребители).
* Логване – осъществяване на взаимовръзка със системата под формата на аутентикирано вписване в нея.
* Аутентикация – потвърждаване от страна на системата за коректно въведени потребителски данни.
* Client-Server - тип мрежова архитектура, която отделя клиента от сървъра, и най-често се използва в компютърни мрежи.
* Client - Подава заявки, изчаква отговор, взаимодейства с крайния потребител чрез графичен интерфейс
* База от данни – колекция от информация, която е така организирана, че да може лесно да се достъпва, управлява и актуализира.
* Сървър – стартирана инстанция на софтуерна система, която може да приема заявки от клиент и да връща подходящи отговори.
* HTML- Hypertext Markup Language - декларативен език за описване на информация(текст).
* MVC (Model-View-Controller) - Модел-Изглед-Контролер е архитектурен шаблон за дизайн в програмирането, основан на разделянето на бизнес логиката от графичния интерфейс и данните в дадено приложение.

# Използвани технологии

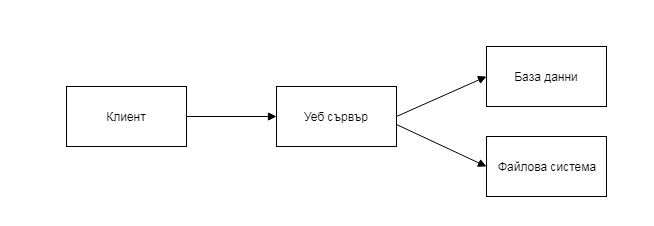
1. Apache server (XAMPP)
2. HTML + CSS (+Bootstrap)

За графичния интерфейс на системата са използвани в по-голямата й част чист HTML и CSS. Всички страници и намиращите се на тях менюта, бутони, логота, визуализиране на обекти (изображения, текст) и други са реализирани без използването на допълнителни библиотеки. Изключение прави страницата, на която се намира live чата – за нея са ползвани готови визуални ефекти от Bootstrap.

1. JS (client-side language) + jQuery (3.4.1)

Използван е за проверки на данни в отделните страници, както и за реализиране на клиентската част най-вече при live chat-а – клиентът е написан на JS, като се ползва jQuery за създаване, изпращане и получаване на заявки от клиента.

1. PHP (server-side language) + Ratchet Library (WebSockets) – сървърната част е писана на PHP, като за създаването и поддръжката на сокети са използвани WebSockets, реализирани в библиотеката Ratchet, която е инсталирана допълнително. Използвана е тази библиотека, тъй като дава най-добрите резултати при ползване на сокети за комуникация в реално време за PHP.
2. Composer – използван е за инсталиране на Ratchet библиотеката и всички нейни dependencies на сървъра.
3. MySQL (XAMPP) – представлява базата данни на системата – съхранява информация за потребителите (техните лични данни) , за албумите (описания, заглавия, изображения, лайкове и дислайкове), както и за съобщенията в live чат-а (всички изпратени съобщения до момента, техния изпращач, времето, в което са пратени и др.)



Клиент - Уеб браузър, който през javascript изпраща данни по HTTP към уеб сървъра. Клиентът може да добавя албуми, заглавия, описания, да харесва/не харесва албуми, да редактира албуми и да осъществява комуникация в реално време с останалите потребители. Преди да се достъпва цялата функционалност, има форми за регистриране и логване.

Уеб сървър - Приложение, написано на PHP, приема заявките от клиентите. Сървърът поддържа authentication и authorization. Заявките се получават от клиентите, спрямо които се извършват съответните операции върху данните.

Базата данни е MySQL и е достъпвана от уеб сървъра. В нея се съдържат информация за отделните потребители под криптиран вид(потребителски имена, пароли, имейли), информация за албумите(изображения , към които се съдържат пътищата във файловата система, информация за самите албуми и кой потребител ги е публикувал), информация за чата – всички съобщения до момента, налични потребители, време на изпращане и кой е изпратил съобщението.

Файлова система – в нея се намират всички изображения, които се добавят от потребителите в албумите. Уеб сървъра се грижи за извършването на добавяне, изтриване, редактиране след преглед и нанасяне на промени в базите данни.

# Реализация на базата от данни

Тъй като системата е реализирана с две основни функционалност – 1) Създаване и качване на албуми от изображения и описания за тях и 2) Възможност за комуникация между потребители в реално време (Live chat), то са създадени две бази от данни, съответно БД imagesdatabase (в нея има информация само за албумите и всичко свързано с тях – път към дадения файл, описание на албум, заглавие, брой харесвания/нехаресвания) и БД websocket, в която се намира информация за потребителите, които са налични на системата и информация за чата (съобщения и подробни детайли за тях).

В БД imagesdatabase се намира 1 таблица наименувана photos, която има следната структура:

Таблица **photos**, състояща се от следните 6 колони:

1) id INT 11 символа AUTO\_INCREMENT PRIMARY\_KEY  
2) file\_name varchar 255 символа   
3) user\_name varchar 100 символа  
4) uploaded\_on datetime  
5) status enum(‘1’, ‘0’) ПО ПОДРАЗБИРАНЕ 1  
6) text text  
7) album varchar(100) 100 символа  
8) likesUsers text  
9) dislikesUsers text  
10) likesCount int(11) 11 символа ПО ПОДРАЗБИРАНЕ 0  
12) dislikesCount int(11) 11 символа ПО ПОДРАЗБИРАНЕ 0

Където file\_name е името на изображението, което се качва, user\_name е името на свързания с изображението/албума потребител, uploaded\_on е съответно датата и часът, в който е качен албумът, или, ако е добавено изображение при редактиране на албум, съответно кога е добавено, text представлява текстовото описание, album е името на албума/дестинацията, likesUsers и dislikesUsers се ползват за следене на кой е харесал/не харесал даден албум (за да не се случва даден потребител да може да харесва/не харесва един албум повече от 1 път), likesCount и dislikesCount съответно са за броя харесвания/нехаресвания, за да могат да се представят визуално за всеки потребител, разглеждащ дадения албум.

В БД websocket има 2 таблици със следните структури:

Таблица **users**, състояща се от следните 4 колони:   
1) id INT 11 символа AUTO\_INCREMENT PRIMARY KEY  
2) username varchar 100 символa UNIQUE   
3) email varchar 200 символа UNIQUE  
4) password varchar 255 символа

Всеки потребител има автоматично конфигуриран ид номер, който е уникален (конфигурира се при регистрация на потребител), потребителско име и имейл, които също са уникални за всеки отделен потребител – също се задават при регистрация, но първо се осъществява проверка в базата данни дали вече няма потребител със същите данни. Паролата се криптира и се запазва в БД, за избягване на злоупотреби.

Таблица **chatrooms**, състояща се от следните 4 колони:

1) id INT 11 символа AUTO\_INCREMENT PRIMARY KEY  
2) userid varchar 11 символa UNIQUE   
3) msg varchar 200 символа   
4) created\_on datetime

Всеки чатрум съдържа автоматично конфигуриращо се ид, с което се характеризира всяко едно ново съобщение(задава се при пращане на съобщение), userid, което пояснява даденото съобщение от кого е пратено – препратка към id от таблицата users, за да може всяко едно съобщение да се свързва съответно със своя изпращач, msg – представлява самото съобщение под текстов формат и created\_on представлява момента, в който съобщението е изпратено.

Благодарение на запазването на съобщенията в БД и индексирането им подред може лесно да се визуализират в чат(ако е бил затворен и след това отворен отново, предните съобщения няма да бъдат загубени).

Благодарение на пазенето на userid за всяко съобщение може да се визуализира в чата името на потребителя, изпратил конкретното съобщение като се направи препратка в таблицата users и се вземе потребителското име на потребителя с даденото id.

Благодарение на пазенето на точния момент, в който е изпратено съобщението може да се визуализира и това в чата под формата на низ с часа и датата на пращане.

# Реализация на бизнес логиката

Във файла **dbConfig** се намират настройките за свързване към базата данни за изображенията, а във файловете, намиращи се в папка **dbB** се намират съответно данните за свързване с базата данни на потребителите и чатрумовете и извършването на самото свързване с базата данни. Там се намират и представянето на потребители и чатове под формата на класове. В папка **uploads** се запазват изображенията след качването им в системата.

В папка **bin** се намира сървърът, който се грижи за осъществяване на реална комуникация за чата чрез WebSockets.

В папка **src** се намира клиентската част от архитектурата Client-Server, която се грижи за чата.

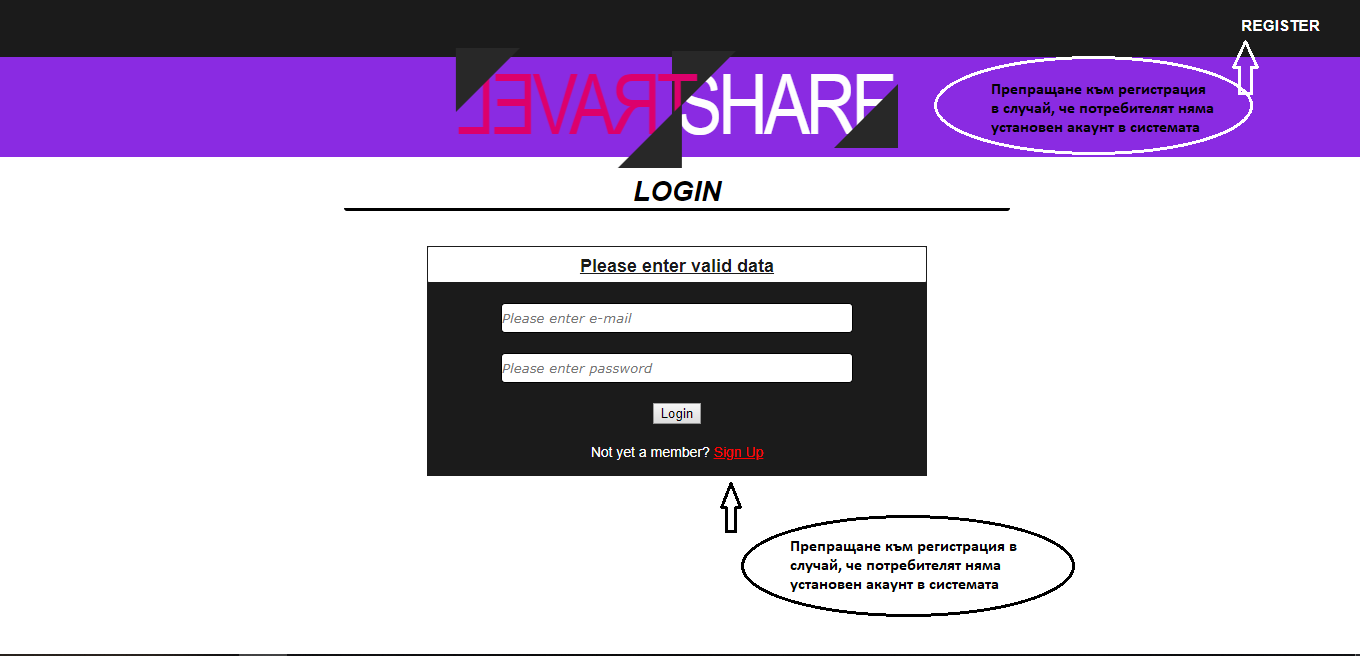
В папка **controllers** във файла **authController.php** се имплементира логин/регистрация/логаут функционалностите – когато потребителят натисне върху един от трите бутона и валидирането на данните, които се приемат от съответните форми. Във файла **upload.php** се имплементира основната функционалност по запазването на публикуваните изображения в базата данни със снимки – валидират се данните, проверяват се дали е добавено нещо и дали правилно се приема от базата данни. Във файла **testIndex.php** е самата визуализация на потребителския интерфейс, когато вече е влязъл в системата – от там може да види, че е лог-нат с дадено потребителско име, както и ползва формата за добавяне на албуми. Освен това там се визуализират и всички албуми според потребителите и историите, които са добавени в системата. Във файла **profile.php** се намира функционалността разглеждане на собствения профил – всеки потребител може да разглежда профила си с всички добавени от него албуми. В случая освен визуализация на албумите има и форма, чрез която потребителят може да променя избрано от него нещо по даден албум. При ползването на тази форма се използва файла **delete.php,** в който е реализирана цялата идея при натискане на един от следните 4 бутона – Изтриване на албум, Промяна на заглавието на албум, Промяна на текстовата информация за албум, Добавяне на нови изображения към избрания албум.

Във файла **likes.php** е реализирана фунцкионалността на бутоните за харесване и нехаресване на даден албум – всеки потребител може да хареса/не нареса даден албум, но може да даде само един вот. В случай, че вече е оценил по един начин албума, то ако реши да го оцени по същия начин, вотът му се премахва (както е във Facebook – можеш да харесаш публикация, но ако натиснеш харесване отново, се анулира последното). За целта се пазят броя харесвания и нехаресвания и списък на потребителите харесали/не харесали даден албум в базата данни с изображения. По този начин, всеки път, когато потребител гласува по единия от двата начина, се извършва проверка в този списък, ако името му фигурира, то се изтрива от списъка и броят на харесванията(нехаресванията) се намаля с единица. Ако пък името му го няма в списъка, вотът се увеличава с единица и името му се добавя във списъка на съответния вот.

# Реализация на потребителския интерфейс

В папката има прикачен и видеофайл, озаглавен Demo, който представлява кратко клипче-демо, в което всички долуописани функционалности са показани как работят.

Началната страница на системата е достъпна само след регистриране в системата и установяване на връзка с профил. Затова при опит за достъп до началната страница, в случай че няма вече установена връзка като потребител, то биваме препратени към страницата Login:



По средата на страницата има форма с две текстови полета: в първото се въвежда имейл адрес, а във второто – парола, а под тях има бутон Login. При въвеждане на данни в двете полета и натискане на бутона, сървърът проверява в базата данни за съответствие между имейл и парола, и ако такова има, то потребителят успешно влиза в системата и бива препратен към началната страница.

В случай, че някое от полетата е невалидно, се получава съобщение за грешка и кое от съответните полета е грешно. Потребителят бива подканен да въведе нови коректни данни, ако иска да влезе в профила си.

В случай, че потребителят няма създаден профил до момента, при натискането на един от двата бутона – Sign up или Register горе вдясно в заглавната част, то той бива препратен към страницата за регистрация.



Отново в средата на страницата има форма, но този път полетата за въвеждане са 4: потребителско име(уникално), имейл адрес(уникален), парола и повтаряне на парола. В случай, че потребителят погрешка е избрал тази страница – вече има създаден профил от предно ползване на платформата, то лесно може да бъде препратен към страницата за вход, като натисне бутонът Sign In, намиращ се под формата за регистрация.

По същата логика, както и при входната страница, и тук при въвеждане на данни в дадено поле и натискане на бутон Register, уеб сървърът осъществява проверка в базата данни дали въведените данни вече не съществуват – в случай на съществуващ имейл ще върне грешка, че потребителят вече е наличен, а в случай на съществуващо потребителско име – съответно, че вече се ползва от друг потребител.

Извършва се и проверка за съвпадението на двете пароли.

Ако всичките данни са коректни (тоест няма такъв потребител за момента), то уеб сървърът криптира тези данни и ги запазва в базата данни (добавя се нов потребител и за всяко поле от таблицата users в БД се добавя конкретна стойност). След това потребителят бива препратен към началната страница на системата.



В горния ляв ъгъл се изписва приветстващо съобщение с потребителското име на текущо ползващия системата потребител. В горния десен ъгъл има три основни бутона, представляващи меню за избор – Profile е страницата, в която всеки потребител може да види всичките публикувани албуми само и единствено от него и подробна информация за тях, Messenger представлява Live чатрум, в който всички потребители могат да контактуват в реално време един с друг, а при натискане на бутона Logout, потребителят напуска системата и сесията приключва (за да влезе отново, ще трябва да въвежда имейл и парола отново).

Под логото на TravelShare в дясно има 2 текстови полета и 2 бутона.

1) В лявото текстово поле при въвеждане на символен низ, потребителят бива препратен към страница, в която са заредени всички албуми на потребител, чието потребителско име съдържа част или целия символен низ. Тоест, ако напишем в полето Pesh, ще бъдем препратени към страница, в която се визуализират всички албуми на потребители, чиито потребителски имена съдържат думата Pesh в себе си.

2) В дясното текстово поле при въвеждане на символен низ, потребителят бива препратен към страница, в която са заредени всички албуми, чието заглавие/име на албум съдържа в себе си въведения символен низ. Тоест, ако напишем в полето Ba, то ще бъдат визуализирани всички албуми, които съдържат в заглавието си думата Ba (например албуми със заглавия Bali, Bangladesh, Bahrein)

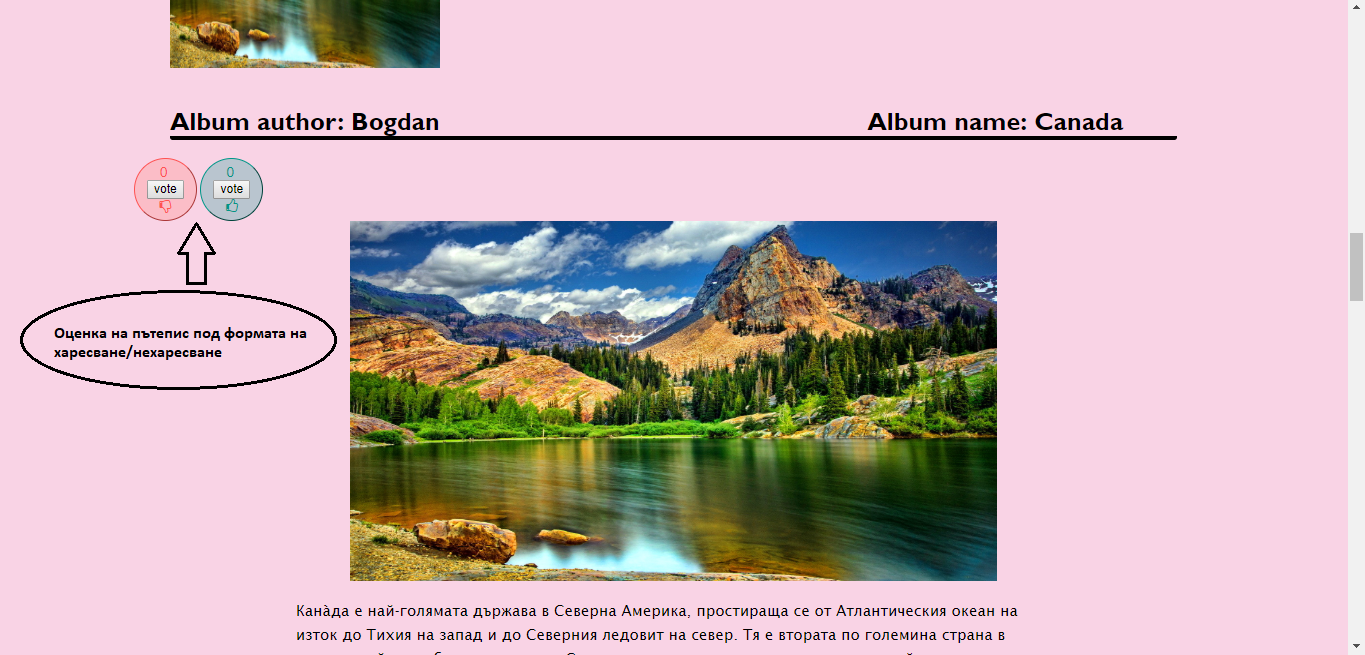
Под логото на TravelShare в ляво има форма с бутон за избор на файлове, текстово поле, поле за въвеждане на текст и бутон.

При натискане на бутона за избор на файлове, потребителят може да избере 1 или повече изображения (поддържат се само основните формати .jpg .bmp .png .jpeg и има лимит за размера на файловете, които се качват). В случай, че форматът на избраните файлове е друг или не е спазен размера на качването, албум не се създава, промените се изтриват, и на потребителя бива изписано съобщение, в което е подканен да качи валидни данни.

В текстовото поле потребителят въвежда име на албум.

В полето за въвеждане на текст, потребителят въвежда текстово описание за преживяното от себе си по време на почивката или времето, което е прекарал на дадената дестинация.

С натискането на бутона Upload се извършва проверка дали всички данни са коректни и дали няма празни полета, и ако няма върнато съобщение за грешка, то албумът се създава, страницата се рефрешва и албумът се визуализира заедно с всичката информация за него.

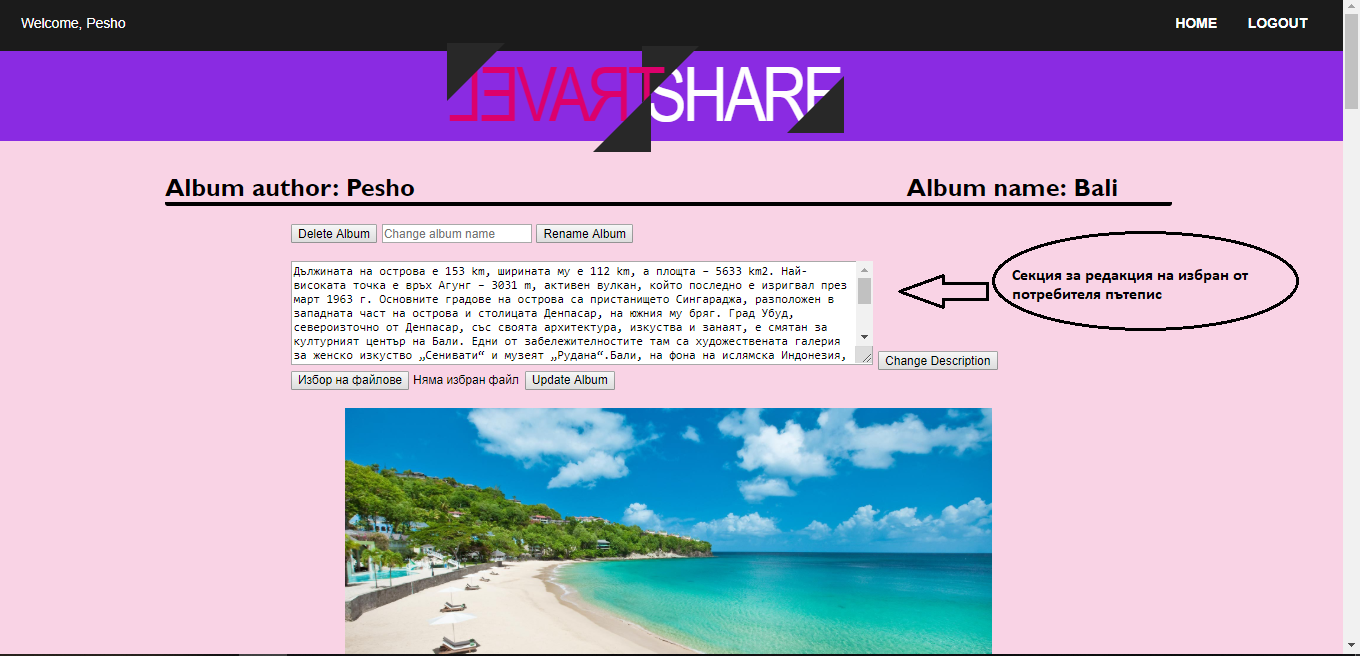


Началната страница представлява Newsfeed, който е подобие на Newsfeed-a ползван в страници като Facebook и Instagram: Албумите се визуализират един под друг като са ясно ограничени къде започват и свършват. Всеки албум започва с начална лента, в лявата страна на която се намира потребителското име на автора на албума, а в дясната страна на лентата стои името на дадения албум. След това се визуализира първата добавена снимка като корица – изображение с по-голям и открояващ се размер. Под изображението корица след описанието, което потребителят е въвел, и чак под него се намира галерия с останалите изображения(ако има такива), описващи дадения албум.

Подреждането на албумите се осъществява по азбучен ред на албумите, а в случай, че има няколко албума с едно и също име, но са на различни потребители, се подреждат по азбучен ред и на потребителите. Системата не позволява един и същи потребител да публикува албум с еднакви имена – ако потребител реши да качи албум с име X, но вече има такъв негов албум, то към името на новия албум се добавя и текущата дата на качване.

Освен всичката информация, която потребител въвежда с качването на даден албум, то под името на автора на албума има 2 бутона – Харесване и Отхаресване. Подобно на Facebook и Instagram отново всеки потребител може да харесва/не харесва даден албум. Запазена е логиката, че ако даден потребител вече е харесал/отхаресал даден албум и отново натисне съответния бутон, то броят съответни вотове намалява с 1, а в случай, че не го е харесал/отхаресал до момента, се увеличават с 1. (Например потребителят с потребителско име Pesho не е харесал до този момент албума Canada, който е показан горе на снимката. Тогава, с натискането на бутона vote в зеления кръг, броят Likes става от 0 на 1. И обратното, ако вече го е харесал, и отново натисне vote, to Likes спадат с 1 брой.)

Следващата страница, до която регистриран потребител има достъп е Profile – всички албуми на текущо ползващия системата потребител.



Албумите отново са заредени в лексикографски ред по заглавие на албум.

Към всеки албум, преди неговото фоново изображение, описание и допълнителни изображения, откриваме форма. Във формата има бутон, текстово поле, бутон, поле за въвеждане на текст, отново бутон, бутон за избор на файлове друг бутон.

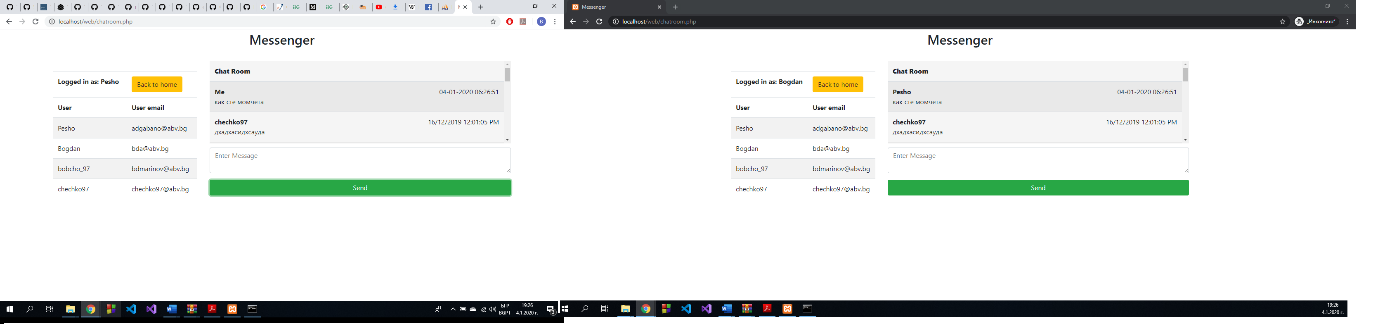
При натискане на бутона Delete Album, потребителят изтрива албума, към който сочи бутона.

При въвеждане на текст в текстовото поле и натискане на бутона Rename Album, потребителят променя заглавието на албума, към който сочат текстовото поле и бутона.

При въвеждане или пък изтриване на част от предишното описание от текстовото поле и натискане на бутона Change Description, потребителят променя описанието на албума, към който сочат полето за въвеждане на текст и бутона.

При натискане на бутона за избор на файлове и избиране на 1 или повече изображения (отново както в главната страница има ограничение за типа на файловете и размера) и натискане на бутона Update Album, към албума, към който сочат бутоните биват добавени новоизбраните изображения.

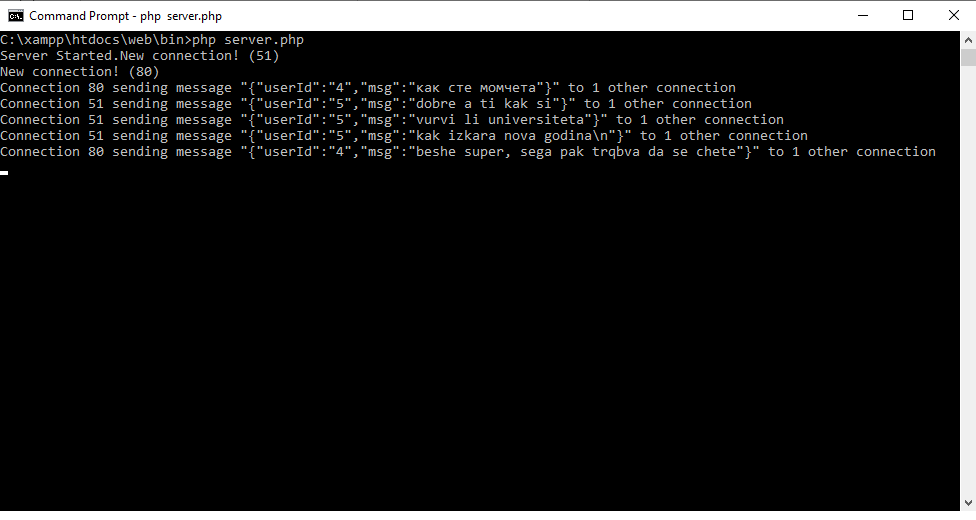
Следващата страница е Messenger – Chatroom, в който потребителите контактуват едни с други в реално време без забавяне или рефрешване на страницата при пращане на съобщение.



В левия прозорец на Chrome сме влезли в системата като потребител Pesho, а в десния в инкогнито режим на браузъра сме влезли в системата с потребителско име Bogdan.

Самият чат е разделен на две секции – в лявата най-горе ни показва потребителското име, с което сме логнати в момента, а под него всички потребители и техните имейли, които някога са влизали в системата. В дясната секция в прозореца Chat Room се изписва всяко едно съобщение в момента, в който бъде пратено. Над текста на съобщението в ляво стои потребителското име, което изпраща съобщението (или Me, ако го пращаме ние), а в дясно стои точния момент, в който е пратено съобщението. Под изпратените съобщения има текстово поле за въвеждане на съобщение, а под него бутон за изпращане.

Всички съобщения се запазват в БД и затова дори при затваряне на чата, следващия път, когато го отворим, всички предходни съобщения няма да бъдат загубени. В случай на установяване на връзка със системата от даден потребител и пращане на съобщение в чата, то получаваме следното в сървърната част:



# Внедряване на системата

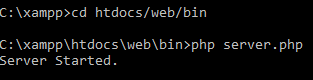
*Опишете изискванията и последователността от действия за внедряване на системата.*

1. Инсталиране на xampp и стартиране на Apache Server и MySQL Admin за конфигуриране на базите данни – създават се две бази данни: БД imagesdatabase и БД websocket.

В базата данни imagesdatabase се поставя файла с конфигуриране на структура на таблица и съответно примерни данни /newDB/photos.sql (sql файл създаващ таблица със структура и примерни данни за изображения, намиращ се в папката newDB на проекта).

В базата данни websocket се поставят два файла: /newDB/users.sql и /newDB/chatrooms.sql (2 sql файла, като 1-вият създава таблица със структура и примерни данни за потребители, а 2-рият създава таблица със структура и примерни данни за чат със съобщения. И двата файла се намират в папката newDB на проекта).

1. След създаване на базите данни се пуска сървърът чрез терминал/command line за Windows с командите:



Излиза съобщение, което индикира, че сървърът е стартирал – Server started.

1. Влизаме в браузър (тествано е на Chrome и работи) и въвеждаме следния адрес:

localhost/web/login.php

1. Въвеждаме някакви потребителски данни или се регистрираме. За улеснение в системата има 4 потребителски профила със следните данни:

Имейл Парола

[adgabano@abv.bg](mailto:adgabano@abv.bg) 123456

[bdmarinov@abv.bg](mailto:bdmarinov@abv.bg) 123456

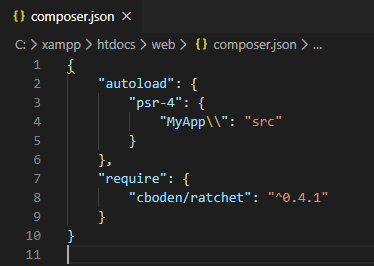
[chechko97@abv.bg](mailto:chechko97@abv.bg) 123456

[bda@abv.bg](mailto:bda@abv.bg) 123456

1. За използване на чата натискаме бутона Messenger в заглавното меню на началната страница. Отваряме още един таб в режим инкогнито (отново тествано и работи при Chrome), логваме се с друг потребителски акаунт и отново избираме Messenger в заглавното меню.
2. В случай, че чатът не работи, на вашата система трябва да инсталирате Composer <https://getcomposer.org/download/> - използва се за създаване на dependencies и изтегляне на библиотеката, която сме ползвали за WebSockets – Ratchet. В папката, в която е проекта (в случая web) от командната линия въведете следния код:

C:\xampp\htdocs\web>composer require cboden/ratchet

Това ще инсталира библиотеката в папката. След това заменете съдържанието на composer.json със следното:



От командната линия за папката, в която е проекта (в случая web) въведете:

C:\xampp\htdocs\web>composer update

Отново се върнете на точка 2) от описанието за внедряване на системата.

# Идеи за възможни подобрения

В тази секция са изложени някои примери, с които проектът може да бъде подобрен в бъдеще и да има по-завършен вид:

1. Слабото звено на проекта до момента е графичният интерфейс – заложено е повече на работещи функционалности, отколкото на външен вид на този етап, тъй като работещ правилно проект е по-добре отколкото открояваща се хубаво на външен вид система, но със слабо реализирани функционалности.
2. Добавяне на различни типове потребители – за момента системата поддържа два типа потребители – нерегистриран и регистриран. Първият не може да достъпва системата – за него важат единствено регистрация и логин. Вторият потребител притежава всички функционалности на системата – добавяне/изтриване/редактиране/преглед/търсене на албуми, комуникация с други потребители в реално време. Хубаво би било да има и администратор на системата, който може да следи за злоупотреби с нея, за неправилно публикувани материали, които не съответстват на нуждите на потребителите, ползващи системата и да има правото да изтрива/редактира дадена публикация и дори да забранява достъпа на даден потребител за определено време.
3. Идея, която имаше в самото начало на проекта беше да се реализира възможност за добавяне на коментари под всеки албум. За момента това не направено, следва да бъде добавен
4. Самият чат може да бъде подобрен – вместо да показва имейлите на потребителите, може да бъде реализиран да показва кой потребител е на линия в момента, така че всеки потребител да може да вижда кои са останалите, с които може да контактува в този момент в чатрума.