School of Information and Communication Technologies ICT, Belgrade

Documentation

NYCEstate

https://nycestate.000webhostapp.com/

Web Programming PHP 1

Ilija Krstić 155/21

Major: Internet technologies

Module: Web programming

Subject: Web Programming PHP 1

Table of contents

Table of contents 2
1. Introduction 6
1.1 Technologies used 6
1.2 Broad description of functionality 6
General functionality 6
index.html 6
listings.html 6
favorites.html 6
listing.html 6
survey.html 7
contact.html 7
admin.html 7
register.html 7
login.html 7
2. Organizational structure 8
2.1 Organizational diagram 8
2.2 Sitemap 8
2.3 Images of pages and descriptions of functionality 10
2.3.1 index.html 10
2.3.2 listings.html 11
2.3.3 favorites.html 12
2.3.4 listing.html 12
2.3.5 survey.html 13
2.3.6 contact.html 13
2.3.7 admin.html 14
2.3.8 register.html 15
2.4 Database Diagram 16
3. Code 17
3.1 PHP 17
3.1.1 PHP Files in /BusinessLogic 17
addToFavoriteListings.php 17
attemptLogin.php 18

createNewBorough.php 20
createNewBuildingType.php 21
createNewLink.php 22
createNewListing.php 25
createNewMessage.php 28
createNewMessageType.php 30
createNewQuestion.php 31
createNewRoomType.php 33
createNewUser.php 34
deleteFromTable.php 36
editBorough.php 38
editBuildingType.php 40
editLink.php 41
editListing.php 44
editMessageType.php 49
editRoomType.php 50
editUser.php 52
getAllAccessLevels.php 54
getAllBoroughs.php 55
getAllBoroughsCount.php 56
getAllBoroughsWithListings.php 56
getAllBuildingTypes.php 57
getAllBuildingTypesCount.php 57
getAllBuildingTypesWithListings.php 58
getAllDeletedListings.php 59
getAllDeletedQuestions.php 59
getAllLinks.php 60
getAllListings.php 60
getAllMessages.php 61
getAllMessageTypes.php 61
getAllMessageTypesCount.php 62
getAllNavigationLocations.php 62
getAllQuestions.php 63

getAllRoles.php 64
getAllRoomTypes.php 64
getAllUsers.php 65
getLinks.php 66
getListingsForFilter.php 67
getQuestionsForUser.php 69
getSpecificBorough.php 71
getSpecificBuildingType.php 72
getSpecificDetailedListing.php 73
getSpecificLink.php 74
getSpecificListing.php 76
getSpecificMessageType.php 77
getSpecificQuestion.php 79
getSpecificRoomType.php 83
getSpecificUser.php 85
logout.php 86
removeFromFavoriteListings.php 87
restoreListing.php 88
restoreQuestion.php 89
submitAnswer.php 90
3.1.2 PHP Files in /DataAccess 91
accessLevelFunctions.php 91
boroughFunctions.php 91
buildingTypeFunctions.php 93
favoriteFunctions.php 95
generalFunctions.php 96
imageFunctions.php 100
informationFunctions.php 102
linkFunctions.php 102
listingFunctions.php 106
messageFunctions.php 117
navigationLocationFunctions.php 119
roomTypeFunctions.php 120

surveyFunctions.php 121
userFunctions.php 127
3.1.3 PHP Files outside of root 132
connection.php (not actual credentials) 132
3.2 CSS 133
3.2.1 style.css 133
3.3 HTML 142
3.3.1 index.html 142
3.3.2 admin.html 144
3.3.3 author.html 154
3.3.4 contact.html 157
3.3.5 favorites.html 159
3.3.6 listing.html 160
3.3.7 login.html 161
3.3.8 register.html 163
3.3.9 survey.html 165
3.3.10 listings.html 167
3.4 Javascript 169
3..4.1 main.js 169

1. Introduction

1.1 Technologies used

Primary technologies used for this website are PHP, JavaScript, HTML and CSS with additional styling from bootstrap

1.2 Broad description of functionality

General functionality

- Dynamic menus, only showing users the links they are allowed to access
- Protection from unauthorized access both on the frontend and on the backend
- Frontend error handling
- Success messages
- Checks mentioned throughout the documentation are performed both on the frontend and the backend whenever possible

index.html

- Loads all boroughs with listings into a dropdown field.
- Pressing the search button redirects the user to the listings page, preselecting the borough selected from the dropdown field, if one is selected.

listings.html

- Allows for querying listings based on the text in their title, the borough that they are in as well as the building type of the listing
- Allows for sorting listings based on recency, price, and size in feet
- Displays listings according to the filters and sorting mentioned above
- Initially the title, description, size and price of the listings are displayed.
- On hover, the aforementioned data is superseded by additional information, that being the building type, borough and any rooms provided during listing creation of the listing.
- Listings can be added to favorites by pressing the heart outline icon present on all non-favorite listings, only logged in users can add listings to favorites.
- Listings can be removed from favorites by pressing the heart icon present on all favorite listings.
- Users can open the listing page of the listing they'd like by pressing the read more buttons present on all listings.
- Users' sort and filter selections are saved upon every search

favorites.html

- Displays all listings previously added to favorites
- The listings are displayed are displayed in the same format as in listings.html

listing.html

- Displays complete information about a given listing based on URL search parameters
- Users can click on the borough/building type of the listing and be taken back to listings.html with the borough/building type clicked on applied as the appropriate filter

- Users can click on the copy phone number button to copy the phone number of the agency into their clipboard (the phone number is universal and not specific to the listing, loaded from a PHP file)
- Shows appropriate message if the listing does not exist or is no longer active

survey.html

- Allows logged in users to answer well formatted questions provided by the website admins.
- Removes question from view once it is answered.
- Only shows currently active questions/answers.
- Questions have at least two active answers each at all times.

contact.html

- Allows logged in users to send messages to the website admins.
- Users must define the title of the message, between two and five words, before submitting their message.
- Users must choose from message types defined by the admins before submitting their message.
- Users must define the body of the message, between three and fifteen words, before submitting their message.

admin.html

- Allows website admins broad control over all dynamically displayed items on the website, including inserting, editing, deleting, soft deleting, and restoring items.
- Allows website admins to view messages submitted by other users
- Allows website admins to view answer statistics of questions, current and previous ones.

register.html

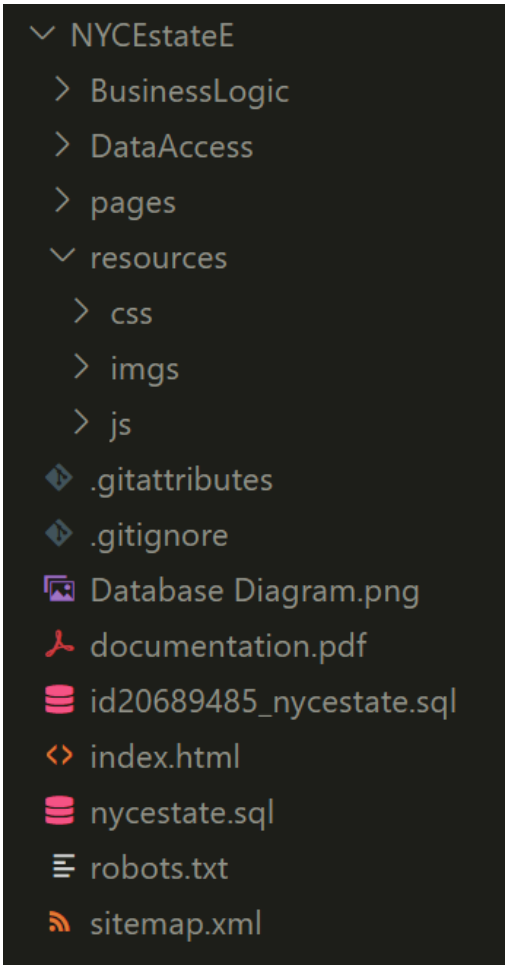
- Allows users to register to the website, allowing greater use.
- Guarantees strong user passwords with modern regex checks.
- Informs the user if the email they are trying to register with is already taken.
- All passwords are stored in a hashed and salted format.

login.html

- Allows users who have previously registered at register.html to log in with the appropriate credentials.
- Does not allow access for users whose accounts have been deleted or banned

2. Organizational structure

2.1 Organizational diagram



- connection.php is located outside of the root of the website for better security.
- /pages contains all .html pages excluding the landing page (index.html).
- /resources/imgs contains all images.
- /resources/js contains the javascript used on the website.
- /resources/css contains the css used on the website.
- /BusinessLogic contains all API endpoints consumed by the frontend.
- /DataAccess contains functions that the API endpoints use.
- The database diagram as well as the databases visible in the screenshot above are only available locally.

2.2 Sitemap

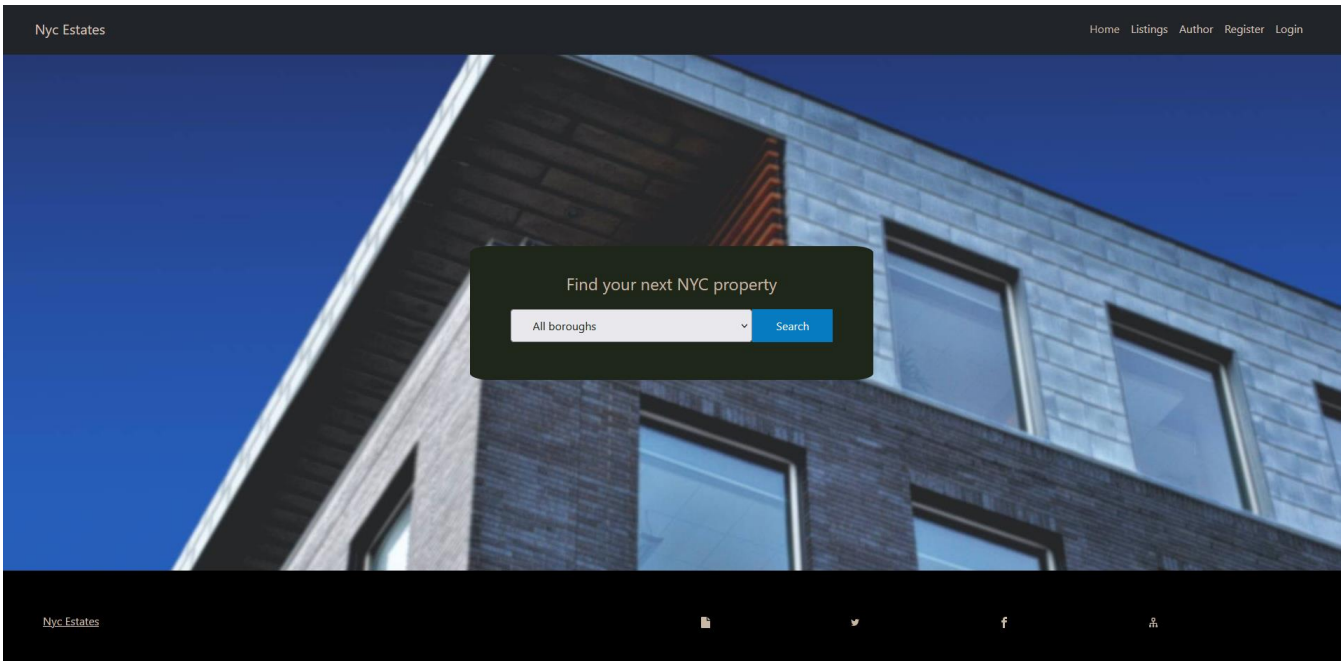
```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
   <url>
      <loc>https://nycestate.000webhostapp.com/</loc>
      <lastmod>02-05-2023</lastmod>
```

```
    <changefreq>daily</changefreq>
    <priority>1</priority>
  </url>
   <url>
    <loc>https://nycestate.000webhostapp.com/pages/listings.html</loc>
    <lastmod>02-05-2023</lastmod>
    <changefreq>weekly</changefreq>
    <priority>0.95</priority>
  </url>
  <url>
    <loc>https://nycestate.000webhostapp.com/pages/listing.html</loc>
    <lastmod>02-05-2023</lastmod>
    <changefreq>weekly</changefreq>
    <priority>0.90</priority>
  </url>
   <url>
    <loc>https://nycestate.000webhostapp.com/pages/favorites.html</loc>
    <lastmod>02-05-2023</lastmod>
    <changefreq>monthly</changefreq>
    <priority>0.85</priority>
  </url>
   <url>
    <loc>https://nycestate.000webhostapp.com/pages/login.html</loc>
    <lastmod>02-05-2023</lastmod>
    <changefreq>yearly</changefreq>
    <priority>0.80</priority>
  </url>
   <url>
    <loc>https://nycestate.000webhostapp.com/pages/register.html</loc>
    <lastmod>02-05-2023</lastmod>
    <changefreq>yearly</changefreq>
    <priority>0.75</priority>
  </url>
   <url>
    <loc>https://nycestate.000webhostapp.com/pages/survey.html</loc>
    <lastmod>02-05-2023</lastmod>
    <changefreq>yearly</changefreq>
    <priority>0.70</priority>
  </url>
   <url>
    <loc>https://nycestate.000webhostapp.com/pages/contact.html</loc>
    <lastmod>02-05-2023</lastmod>
    <changefreq>yearly</changefreq>
    <priority>0.65</priority>
  </url>
```

```
    <url>
      <loc>https://nycestate.000webhostapp.com/pages/admin.html</loc>
      <lastmod>02-05-2023</lastmod>
      <changefreq>yearly</changefreq>
      <priority>0.60</priority>
   </url>
    <url>
      <loc>https://nycestate.000webhostapp.com/pages/author.html</loc>
      <lastmod>02-05-2023</lastmod>
      <changefreq>yearly</changefreq>
      <priority>0.55</priority>
   </url>
</urlset>
```

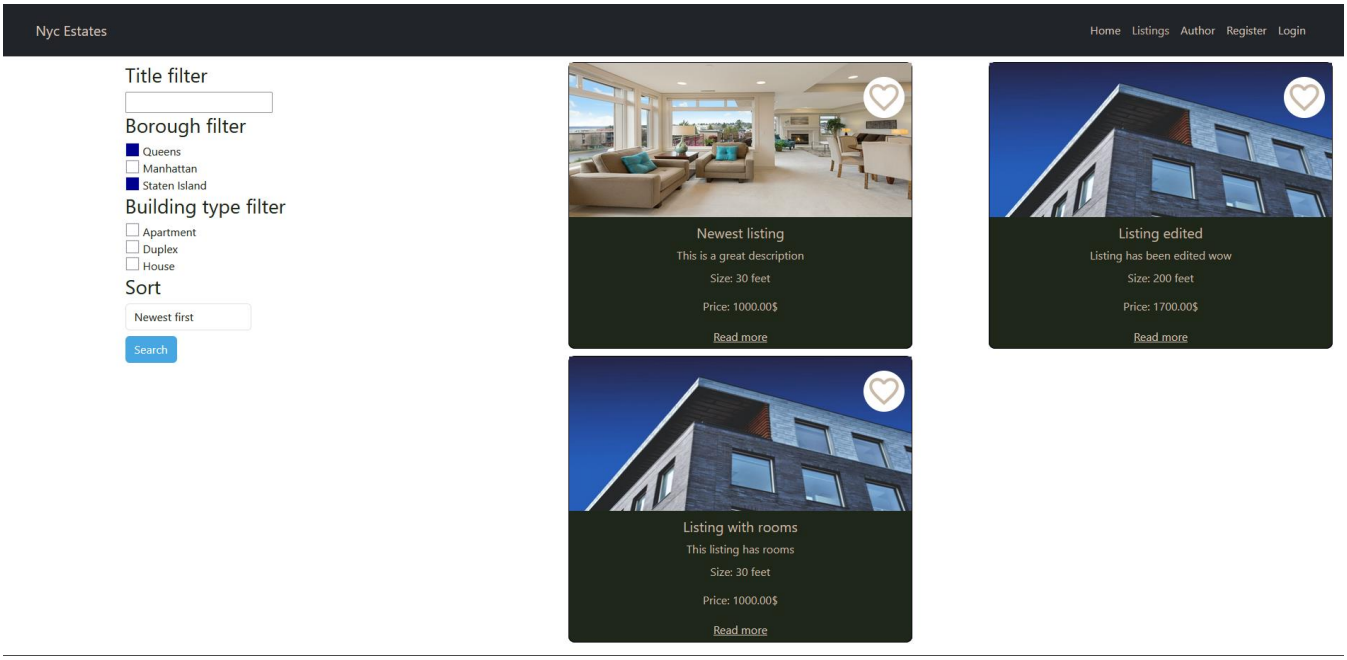
2.3 Images of pages and descriptions of functionality

2.3.1 index.html

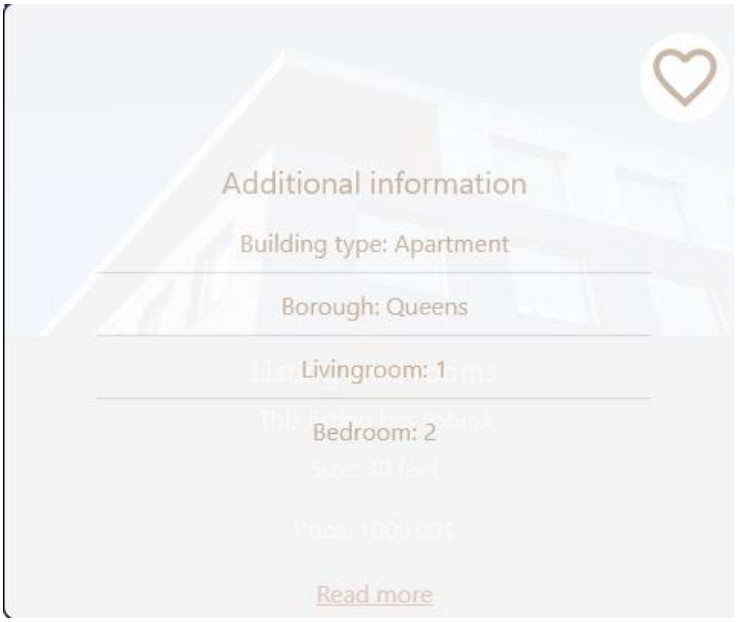


- The landing page of the website
- Allows for immediate searching and querying of listings by taking the user to the listings page with the borough they selected

2.3.2 listings.html

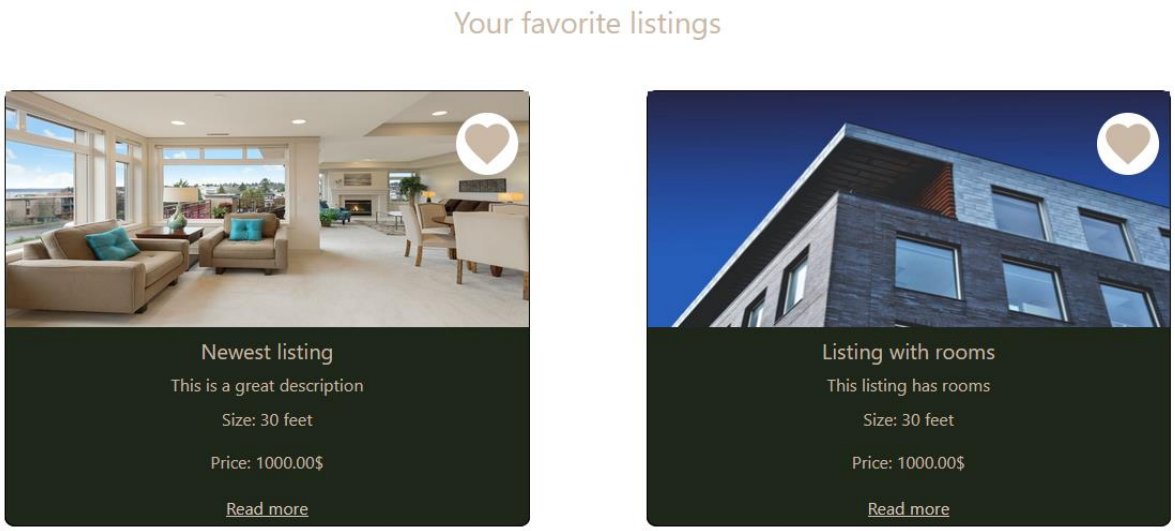


- Allows for searching and previewing of listings based on the filters selected by the user.
- Allows logged in users to add and remove listings from their favorites.
- Only boroughs and building types with active listings are shown.
- Users can sort listings by recency, price, and size in feet.



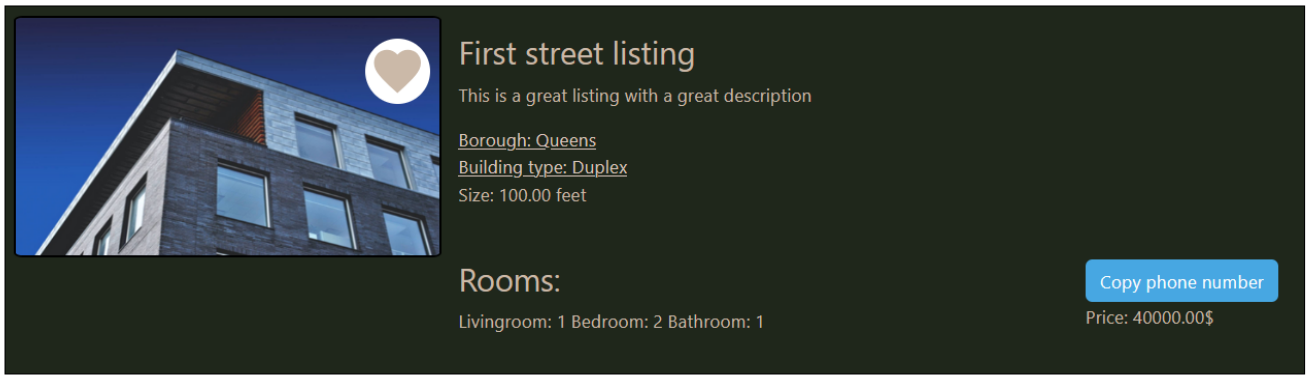
- Pictured above is additional information provided to the user when a listing is hovered, note that the read more and favorite buttons are still interactable.

2.3.3 favorites.html



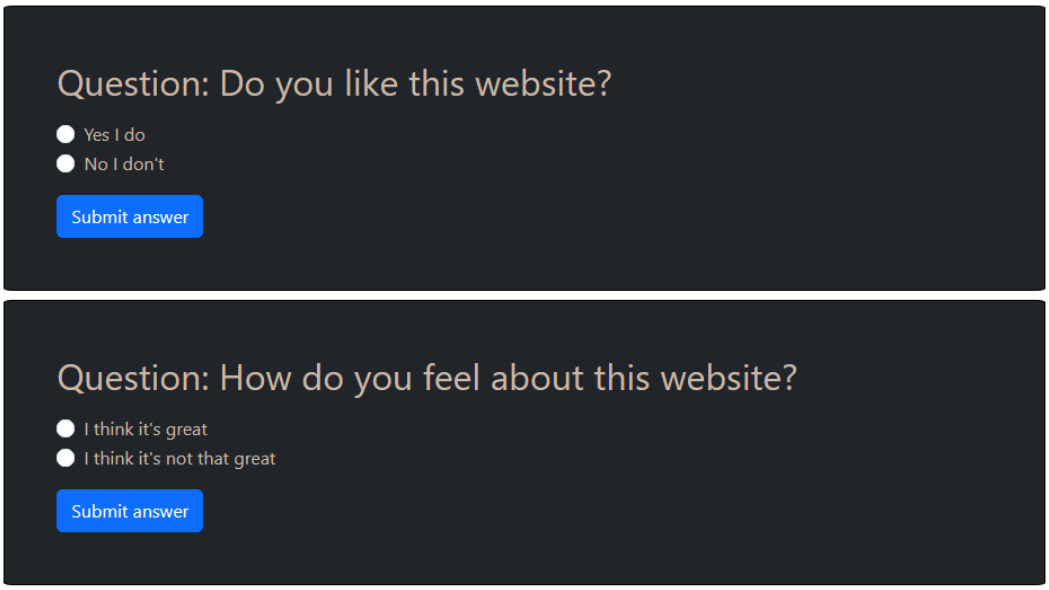
- Allows the user to preview listings previously added to favorites.

2.3.4 listing.html



- Allows the user to obtain all information available about the select listing.
- Allows the user to search other listings in the borough of the given listing.
- Allows the user to search other listings of the same building type as the given listing.
- Allows the user to copy the phone number of the agency, the phone number is universal and not specific to the listing.
- Allows the user to add/remove the listing from their favorites.
- Shows appropriate message in case the listing does not exist or is no longer active.

2.3.5 survey.html

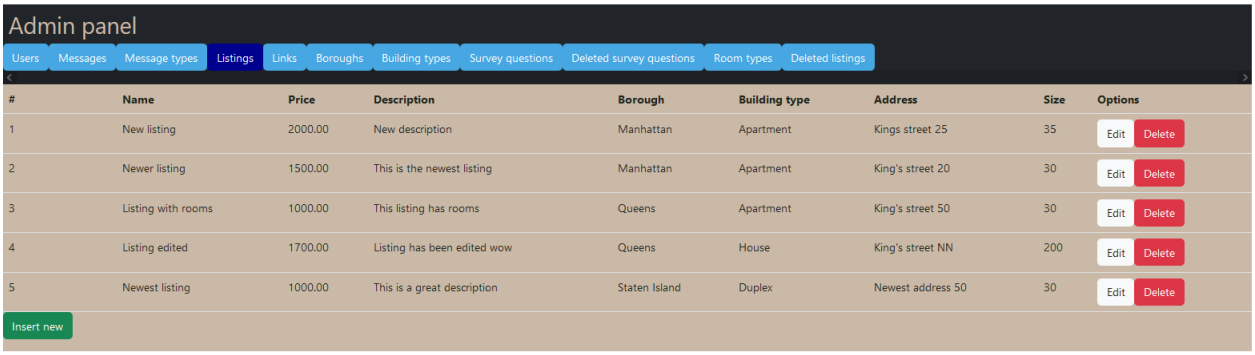


- Allows logged in users of the website to give input to the admins by responding to questions defined by said admins in the admin panel with answers predefined by the admins.
- Users are only shown questions they have yet to respond to.
- Upon answering a question, it is removed from the user's view and a query is made to check for new questions or changes to existing ones

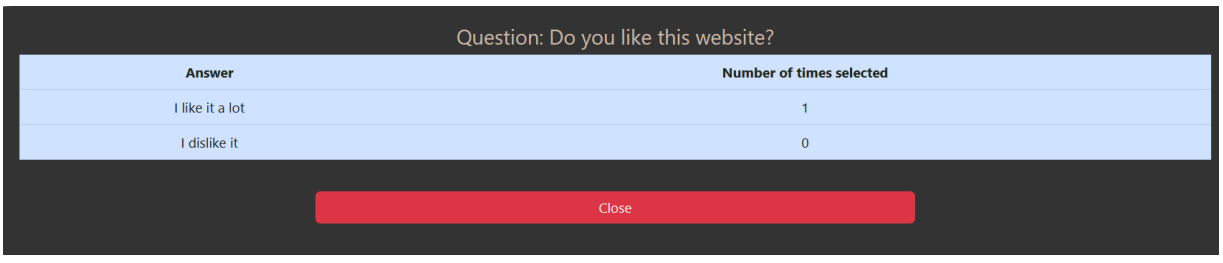
2.3.6 contact.html

- Allows logged in users to send messages to the website admins.
- Users must define the title of the message, between two and five words, before submitting their message.
- Users must choose from message types defined by the admins before submitting their message.
- Users must define the body of the message, between three and fifteen words, before submitting their message.

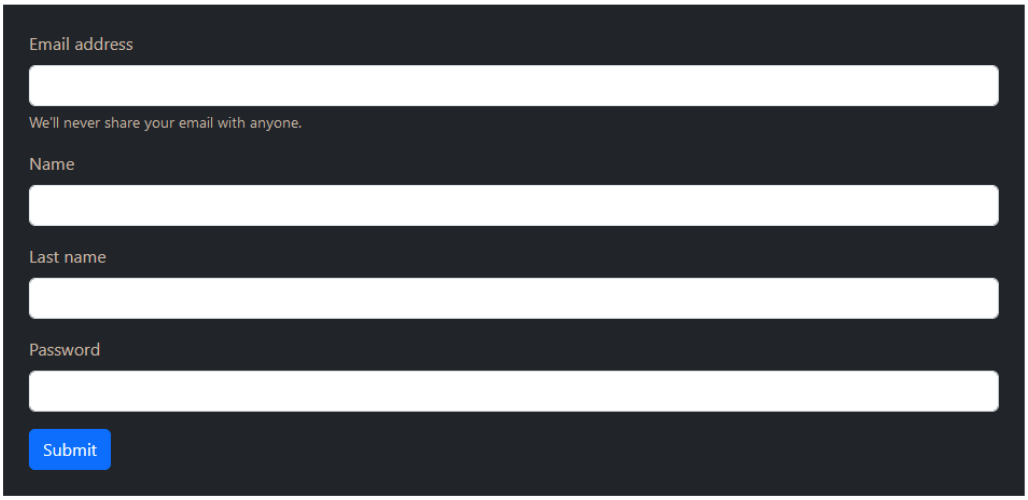
2.3.7 admin.html



- Pictured above is one of the many tabs of the admin panel, each granting the website admins great control over the content on the website.
- Admins can edit user accounts, message types, listings, links, boroughs, building types, survey questions, deleted survey questions, room types and deleted listings
- Admins can soft delete listings, questions and question answers
- Admins can restore deleted listings and questions
- Admins can ban users
- Admins can delete users, messages, links, boroughs, building types and room types
- Admins can only delete the boroughs, building types and room types that are not referenced in listings
- Admins can create new message types, listings, links, boroughs, building types, survey questions and room types
- Admins can view the responses to the questions (both active and inactive questions as well as active and inactive answers to those questions).
- Any change made is immediately represented.
- New boroughs, building types, and room types are immediately available to admins for listing creation and editing.

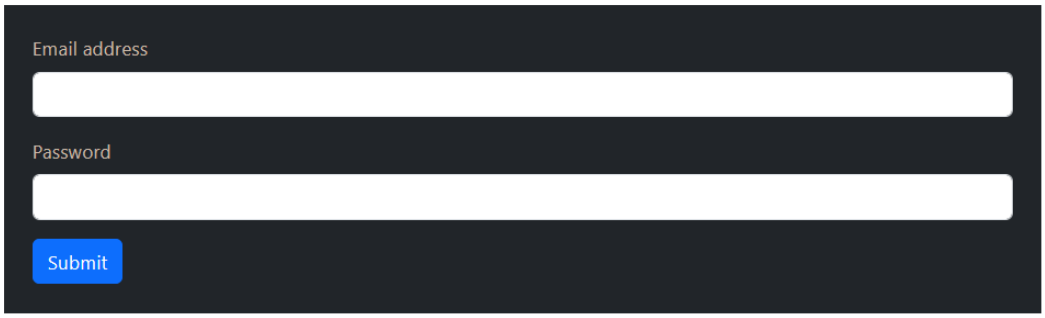


2.3.8 register.html



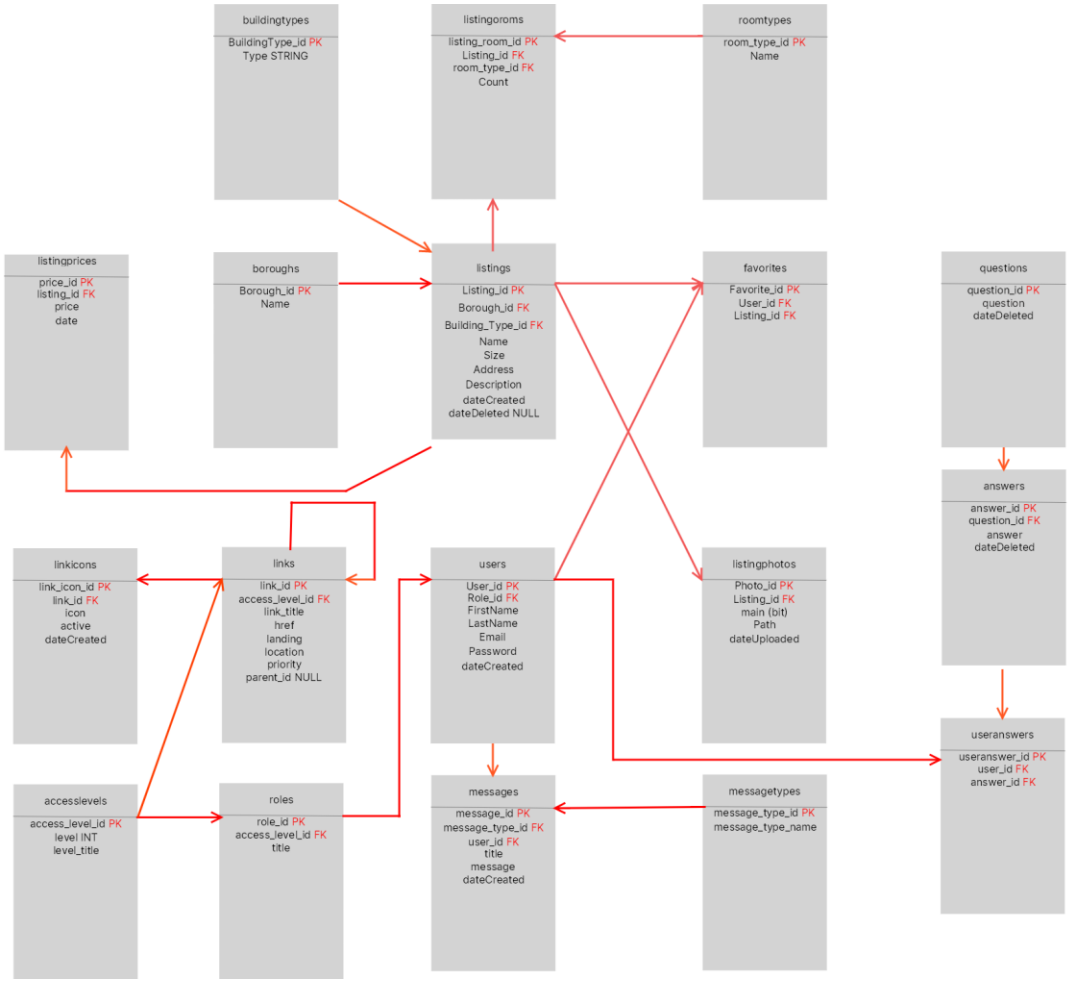
- Allows users to register to the website, allowing greater use.
- Guarantees strong user passwords with modern regex checks.
- Informs the user if the email they are trying to register with is already taken.
- All passwords are stored in a hashed and salted format.

2.3.9 login.html



- Allows users who have previously registered at register.html to log in with the appropriate credentials.
- Does not allow access for users whose accounts have been deleted or banned.
- Tells users who enter credentials that do not fit format that their login attempt failed, without sending it to the server. (Two-way checks are done everywhere else as well, the only difference is that here the user has the illusion of communicating with the server, allowing for greater protection from people engaging in malicious activity)

2.4 Database Diagram



3. Code

3.1 PHP

3.1.1 PHP Files in /BusinessLogic

addToFavoriteListings.php

```php
<?php
session_start();
$requiredLevel = 2;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel, "You must be logged in to add listings to
favorites");


$result;


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


if(!isset($_POST["listingId"])){
    echoImproperRequest("All fields are required");
}


$userId = $_SESSION["user"]["user_id"];
$listingId = $_POST["listingId"];


require("../DataAccess/favoriteFunctions.php");


$alreadyFavorite = checkIfAlreadyFavorite($userId, $listingId);


if($alreadyFavorite){
    echoUnprocessableEntity("Cannot favorite listing more than once");
}
```

```php
try{
    saveUserFavorite($userId, $listingId);
    $result["general"] = "Successfully added listing to favorites";
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

attemptLogin.php

```php
<?php
session_start();
require("../DataAccess/generalFunctions.php");


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


//This check would not exist in a stateless design
//So it should not exist here (Groundwork for switching to JWT)
/* if(isset($_SESSION["user"])){
        echoNoPermission("You are already logged in");
    }
*/


if(!isset($_POST["pass"])
|| !isset($_POST["email"])
)
{
    echoImproperRequest("All fields are required");
}


$pass = $_POST["pass"];
```

```php
$email = $_POST["email"];


$rePass1 = '/[A-Z]/';
$rePass2 = '/[a-z]/';
$rePass3 = '/[0-9]/';
$rePass4 = '/[!\?\.]/';
$rePass5 = '/^[A-Za-z0-9!\?\.]{7,30}$/';


$errors = 0;


if(!preg_match($rePass1, $pass) || !preg_match($rePass2, $pass)
|| !preg_match($rePass3, $pass) || !preg_match($rePass4, $pass)
|| !preg_match($rePass5, $pass))
{
    $errors++;
}


if(!filter_var($email, FILTER_VALIDATE_EMAIL)){
    $errors++;
}


if($errors != 0){
    echoUnauthorized("Incorrect email/password");
}


require("../DataAccess/userFunctions.php");
try{
    $loginAttempt = attemptLogin($email, $pass);
    if(!$loginAttempt){
        echoUnauthorized("Incorrect email/password");
    }
    $user = getUserInformation($loginAttempt);
    if(!$user["level"] > 0){
        echoUnauthorized("User banned");
    }
    http_response_code(200);
    $_SESSION["user"]["user_id"] = $loginAttempt;
    $result["general"] = "Successful login";
    echo json_encode($result);
```

```
}
catch(PDOException $e){
    echoUnexpectedError();
}
?>
```

createNewBorough.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["boroughName"]))
{
    echoImproperRequest("All fields are required");
}


$boroughName = $_POST["boroughName"];


$reBoroughName = '/^[A-Z][a-z\']{1,50}(\s[A-Za-z][a-z\']{1,50}){0,3}$/';


if(!preg_match($reBoroughName, $boroughName)){
    echoUnprocessableEntity("Borough name does not match format");
}


try{
    insertSingleParamater("boroughs", $boroughName, "borough_name");
```

```
}
catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully created new borough";
http_response_code(201);
echo json_encode($result)
?>
```

createNewBuildingType.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["buildingTypeName"]))
{
    echoImproperRequest("All fields are required");
}


$buildingTypeName = $_POST["buildingTypeName"];


$rebuildingTypeName = '/^[A-Z][a-z\']{1,50}(\s[A-Za-z][a-z\']{1,50}){0,3}$/';


if(!preg_match($rebuildingTypeName, $buildingTypeName)){
```

```
    echoUnprocessableEntity("Building type name does not match format");
}


try{
    insertSingleParamater("buildingtypes", $buildingTypeName, "type_name");
}


catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully created new building type";
http_response_code(201);
echo json_encode($result)
?>
```

createNewLink.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


    if
    (
    (!isset($_POST["title"]))
    || (!isset($_POST["href"]))
    || (!isset($_POST["icon"]))
    || (!isset($_POST["aLevel"]))
```

```
    || (!isset($_POST["location"]))
    || (!isset($_POST["main"]))
    || (!isset($_POST["priority"]))
    )
    {
        echoImproperRequest("All fields are required");
    }


$LinkTitle = $_POST["title"];
$LinkHref = $_POST["href"];
$LinkIcon = $_POST["icon"];
$AccessLevelId = $_POST["aLevel"];
$LinkLocation = $_POST["location"];
$main = $_POST["main"];
$priority = $_POST["priority"];


$reTitle = '/^[A-Z][a-z]{2,15}(\s[A-Za-z][a-z]{2,15}){0,2}$/';
$reHref = '/^(https?:\/\/(www\.)?[-a-zA-Z0-9@:%. \+~#=]{1,256}\.[-a-zA-Z0-
9()]{1,6}\b([-a-zA-Z0-9()@:% \+.~#?&\/=]*))||([a-z]{3,40}\.[a-z]{2,5})$/';
$reIcon = '/^[a-z:-]{5,30}$/';


if(!preg_match($reTitle, $LinkTitle)){
    echoUnprocessableEntity("Link title does not match format");
}


if(!preg_match($reHref, $LinkHref)){
    echoUnprocessableEntity("Link does not match format");
}


if(!empty($LinkIcon) && !preg_match($reIcon, $LinkIcon)){
    echoUnprocessableEntity("Link icon does not match format");
}


if($priority < 1){
    echoUnprocessableEntity("Link priority cannot be lower than 1");
}


if($priority > 99){
```

```
    echoUnprocessableEntity("Link priority cannot be higher than 99");
}


require("../DataAccess/navigationLocationFunctions.php");
$acceptableLocations = getAllNavigationLocations();


$locationAcceptable = in_array($LinkLocation, $acceptableLocations);


if(!$locationAcceptable){
    echoUnprocessableEntity("Invalid location provided");
}


$acceptableALevelIds = getEveryParamFromTable("accesslevels",
"access_level_id");
$idAcceptable = false;
foreach($acceptableALevelIds as $aLevelId){
    if($aLevelId["access_level_id"] == $AccessLevelId){
        $idAcceptable = true;
        break;
    }
}


if(!$idAcceptable){
    echoUnprocessableEntity("Invalid access level provided");
}


//Success
require("../DataAccess/linkFunctions.php");
try{
    $lastInsertedId = createNewLink($LinkTitle, $LinkHref, $AccessLevelId,
$LinkLocation, $main, $priority);
    if(!empty($LinkIcon)){
        createNewLinkIcon($lastInsertedId, $LinkIcon);
    }
}
catch(PDOException $e){
    echoUnexpectedError();
}
```

```php
http_response_code(201);
$result["general"] = "Successfully created new link";
echo json_encode($result);
?>
```

createNewListing.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$result;


if(!isset($_FILES["listingPhoto"])){
    echoUnprocessableUnit("All fields are required");
}


if
(!isset($_POST["listingTitle"])
|| !isset($_POST["listingDescription"])
|| !isset($_POST["listingAddress"])
|| !isset($_POST["listingSize"])
|| !isset($_POST["listingPrice"])
|| !isset($_POST["listingBorough"])
|| !isset($_POST["listingBuildingType"])
)
{
    echoImproperRequest("All fields are required");
}


$listingTitle = $_POST["listingTitle"];
$listingDescription = $_POST["listingDescription"];
$listingAddress = $_POST["listingAddress"];
$listingSize = $_POST["listingSize"];
$listingPrice = $_POST["listingPrice"];
$listingBorough = $_POST["listingBorough"];
$listingBuildingType = $_POST["listingBuildingType"];
```

```php
$reTitle = '/^[A-Z][a-z]{2,15}(\s[A-Za-z][a-z]{2,15}){0,2}$/';
$reAddress = '/^(([A-Z][a-z\d\']+)|([0-9][1-9]*\.?))(\s[A-Za-z\d][a-
z\d\']+){0,7}\s(([1-9][0-9]{0,5}[\/-]?[A-Z])|([1-9][0-9]{0,5})|(NN))\.?$/';
$reDescription = '/^[A-Z][a-z\']{0,50}(\s[A-Za-z][a-z\']{0,50})*$/';


if(!preg_match($reTitle, $listingTitle)){
    echoUnprocessableEntity("Title does not match format");
}
if(!preg_match($reDescription, $listingDescription)){
    echoUnprocessableEntity("Description does not match format");
}
if(!preg_match($reAddress, $listingAddress)){
    echoUnprocessableEntity("Address does not match format", $listingAddress);
}


if($listingSize < 30){
    echoUnprocessableEntity("Size cannot be below 30 feet");
}


if($listingSize > 100000){
    echoUnprocessableEntity("Size cannot be above 100000 feet");
}


if($listingPrice < 1000){
    echoUnprocessableEntity("Price cannot be below 1000$");
}


if($listingPrice > 1000000000){
    echoUnprocessableEntity("Price cannot be above 1000000000$");
}


$target_dir = "../resources/imgs/";
$nameToSave = basename($_FILES["listingPhoto"]["name"]);
$imageFileType = strtolower(pathinfo($nameToSave,PATHINFO_EXTENSION));
$newFileName = time().uniqid(rand());
$target_file = $target_dir.$newFileName.".".$imageFileType;
```

```php
$check = getimagesize($_FILES["listingPhoto"]["tmp_name"]);


if($check === false) {
    echoUnprocessableEntity("Uploaded file is not an image");
}


if ($_FILES["listingPhoto"]["size"] > 8000000) {
    echoUnprocessableEntity("Uploaded file is too large");
}


if($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType !=
"jpeg") {
    echoUnprocessableEntity("Uploaded file of incorrect type".$imageFileType);
}


$boroughExists = count(getEveryRowWhereParamFromTable("boroughs", "borough_id",
$listingBorough)) > 0;
if(!$boroughExists){
    echoUnprocessableEntity("Invalid borough selected");
}
$buildingTypeExists = count(getEveryRowWhereParamFromTable("buildingtypes",
"building_type_id", $listingBuildingType)) > 0;


if(!$buildingTypeExists){
    echoUnprocessableEntity("Invalid building type selected");
}
$rooms = array();


if(isset($_POST["listingRooms"])){
    $rooms = json_decode($_POST["listingRooms"]);
}


foreach($rooms as $room){
    $roomExists = count(getEveryRowWhereParamFromTable("roomtypes",
"room_type_id", $room->roomId)) > 0;
    if(!$roomExists){
```

```
        echoUnprocessableEntity("Invalid room type selected");
    }
    if($room->count < 1){
        echoUnprocessableEntity("Number of any room cannot be below one");
    }
    if($room->count > 99){
        echoUnprocessableEntity("Number of any room cannot be above 99");
    }
}


require("../DataAccess/listingFunctions.php");
require("../DataAccess/imageFunctions.php");
try{
    //First attempt to save main image of the listing to disk
    $imgUploadSuccess = saveAdjustedPhotoToDisk($_FILES["listingPhoto"],
$target_file, 640, 360);
    //If saving the image locally fails, stop execution
    if(!$imgUploadSuccess){
        echoUnexpectedError();
    }
    /*If saving the image locally does not fail, continue
      and save image location in the database*/
    $lastInsertedId = createNewListing($listingBorough, $listingBuildingType,
$listingTitle, $listingDescription, $listingAddress, $listingSize);
    saveMainListingPhoto($lastInsertedId, $newFileName."."$imageFileType);
    saveListingPrice($lastInsertedId, $listingPrice);
    foreach($rooms as $room){
        saveListingRoom($lastInsertedId, $room->roomId, $room->count);
    }
}
catch (PDOException $e){
    echoUnexpectedError();
}


// move_uploaded_file($_FILES["listingPhoto"]["tmp_name"], $target_file);
$result["general"] = "Successfully created new listing";
http_response_code(201);
echo json_encode($result)
?>
```

createNewMessage.php

```
<?php
session_start();
```

```php
$requiredLevel = 2;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["message_type_id"])
|| !isset($_POST["title"])
|| !isset($_POST["body"]))
{
    echoImproperRequest("All fields are required");
}


$messageType = $_POST["message_type_id"];
$title = $_POST["title"];
$body = $_POST["body"];


$reTitle = '/^[A-Z][a-z\']{0,19}(\s[A-Za-z][a-z\']{0,20}){1,4}$/';
$reBody = '/^[A-Z][a-z\']{0,19}(\s[A-Za-z][a-z\']{0,20}){2,14}$/';


if(!preg_match($reTitle, $title)){
    echoUnprocessableEntity("Message title does not match format (Between two
and five words)");
}


if(!preg_match($reBody, $body)){
    echoUnprocessableEntity("Message body does not match format (Between three
and fifteen words)");
}
```

```php
$messageTypeExists = count(getEveryRowWhereParamFromTable("messagetypes",
"message_type_id", $messageType)) > 0;
if(!$messageTypeExists){
    echoUnprocessableEntity("Invalid message type selected");
}


require("../DataAccess/messageFunctions.php");
try{
    createNewMessage($_SESSION["user"]["user_id"], $messageType, $title,
$body);
}
catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully sent message";
http_response_code(201);
echo json_encode($result)
?>
```

createNewMessageType.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;
```

```php
if(!isset($_POST["messageTypeName"]))
{
    echoImproperRequest("All fields are required");
}


$messageTypeName = $_POST["messageTypeName"];


$reMessageTypeName = '/^[A-Z][a-z]{1,19}$/';


if(!preg_match($reMessageTypeName, $messageTypeName)){
    echoUnprocessableEntity("Message type name has to be one capitalized
word");
}


try{
    insertSingleParamater("messagetypes", $messageTypeName,
"message_type_name");
}


catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully created new message type";
http_response_code(201);
echo json_encode($result)
?>
```

createNewQuestion.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");
```

```php
if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["questionName"])
|| !isset($_POST["questionAnswers"]))
{
    echoImproperRequest("All fields are required");
}


$questionName = $_POST["questionName"];
$questionAnswers = $_POST["questionAnswers"];


$reQuestion = '/^[A-Z][a-z\']{1,20}(\s[A-Za-z][a-z\']{0,20}){2,10}$/';
$reAnswer = '/^[A-Z][a-z\']{0,20}(\s[A-Za-z][a-z\']{0,20}){2,10}$/';


if(!preg_match($reQuestion, $questionName)){
    echoUnprocessableEntity("Question does not match format (Between three and
eleven words)");
}


foreach($questionAnswers as $answer){
    if(!preg_match($reAnswer, $answer)){
        echoUnprocessableEntity("Answer does not match format (Between three
and eleven words)");
    }
}
require("../DataAccess/surveyFunctions.php");
try{
    $newQuestion = saveQuestion($questionName);
    foreach($questionAnswers as $answer){
        saveQuestionAnswer($newQuestion, $answer);
    }
}
```

```
catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully created new question";
http_response_code(201);
echo json_encode($result)
?>
```

createNewRoomType.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["roomTypeName"]))
{
    echoImproperRequest("All fields are required");
}


$roomTypeName = $_POST["roomTypeName"];


$reRoomTypeName = '/^[A-Z][a-z\']{1,50}(\s[A-Za-z][a-z\']{1,50}){0,3}$/';


if(!preg_match($reRoomTypeName, $roomTypeName)){
```

```
    echoUnprocessableEntity("Room type name does not match format, eg
Livingroom");
}


try{
    insertSingleParamater("roomtypes", $roomTypeName, "room_name");
}


catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully created new room type";
http_response_code(201);
echo json_encode($result)
?>
```

createNewUser.php

```
<?php
require("../DataAccess/generalFunctions.php");


session_start();
$result;


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


if(isset($_SESSION["user"])){
    echoNoPermission("You are already logged in");
}


if(
   !isset($_POST["name"])
```

```
|| !isset($_POST["lastName"])
|| !isset($_POST["password"])
|| !isset($_POST["email"])
)
{
    echoImproperRequest("All fields are required");
}
$name = $_POST["name"];
$lastName = $_POST["lastName"];
$pass = $_POST["password"];
$email = $_POST["email"];


$reName = '/^[A-Z][a-z]{1,14}(\s[A-Z][a-z]{1,14}){0,2}$/';


$rePass1 = '/[A-Z]/';
$rePass2 = '/[a-z]/';
$rePass3 = '/[0-9]/';
$rePass4 = '/[!\?\.]/';
$rePass5 = '/^[A-Za-z0-9!\?\.]{7,30}$/';


$reEmail = '/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/';


if(!preg_match($reName, $name) || !preg_match($reName, $lastName))
{
    echoUnprocessableEntity("Name/last name does not fit criteria");
}


if(!preg_match($rePass1, $pass) || !preg_match($rePass2, $pass)
|| !preg_match($rePass3, $pass) || !preg_match($rePass4, $pass)
|| !preg_match($rePass5, $pass))
{
    echoUnprocessableEntity("Password does not match format");
}
if(!preg_match($reEmail, $email)){
    echoUnprocessableEntity("Invalid email");
}


require("../DataAccess/userFunctions.php");
try{
```

```php
    $emailInUse = checkIfEmailInUse($email);
    if($emailInUse){
        http_response_code(409);
        $result["error"] = "Email already in use";
        echo json_encode($result);
        die();
    }
    else{
        createNewUser($email, $pass, $name, $lastName);
        http_response_code(201);
        $result["general"] = "Successful registration";
        echo json_encode($result);
        die();
    }
}
catch(PDOException $e){
    echoUnexpectedError();
}
?>
```

deleteFromTable.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$data = json_decode(file_get_contents('php://input'), true);
$_POST = $data;


if(!isset($_POST["table"])
|| !isset($_POST["id"])
)
{
    echoImproperRequest("All fields are required");
}


$requestedTable = $_POST["table"];
$requestedId = $_POST["id"];


$dataTable = "";
```

```
$dataParam = "";


$type = "hard";


switch($requestedTable){
    case "Users" :
        $dataTable = "users";
        $dataParam = "user_id";
        break;
    case "Boroughs" :
        $dataTable = "boroughs";
        $dataParam = "borough_id";
        break;
    case "Listings" :
        $dataTable = "listings";
        $dataParam = "listing_id";
        $type = "soft";
        break;
    case "Links" :
        $dataTable = "links";
        $dataParam = "link_id";
        break;
    case "Building types" :
        $dataTable = "buildingtypes";
        $dataParam = "building_type_id";
        break;
    case "Room types" :
        $dataTable = "roomtypes";
        $dataParam = "room_type_id";
        break;
    case "Survey questions" :
        $dataTable = "questions";
        $dataParam = "question_id";
        $type = "soft";
        break;
    case "Message types" :
        $dataTable = "messagetypes";
        $dataParam = "message_type_id";
        break;
    case "Messages" :
        $dataTable = "messages";
        $dataParam = "message_id";
        break;
```

```php
}


if($dataTable == ""){
    echoUnprocessableEntity("No such table");
}


$exists = count(getEveryRowWhereParamFromTable($dataTable, $dataParam,
$requestedId)) > 0;


if(!$exists){
    echoUnprocessableEntity(substr($requestedTable, 0, -1)." with given id does
not exist");
}


try{
    if($type == "hard"){
        deleteSingleRow($dataTable, $dataParam, $requestedId);
    }
    else{
        softDeleteSingleRow($dataTable, $dataParam, $requestedId);
    }
    $result["general"] = "Successfully deleted
".strtolower(substr($requestedTable, 0, -1));
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    //If constraint error, echoUnprocessableEntity instead
    if($e->errorInfo[1] == 1451){
        echoUnprocessableEntity("Cannot delete a
".strtolower(substr($requestedTable, 0, -1)." referenced in other tables"));
    }
    echoUnexpectedError($e);
}


?>
```

editBorough.php

```php
<?php
session_start();
```

```php
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["boroughName"]) || !isset($_POST["id"]))
{
    echoImproperRequest("All fields are required");
}


$boroughName = $_POST["boroughName"];
$boroughId = $_POST["id"];


$reBoroughName = '/^[A-Z][a-z\']{1,50}(\s[A-Za-z][a-z\']{1,50}){0,3}$/';


if(!preg_match($reBoroughName, $boroughName)){
    echoUnprocessableEntity("Borough name does not match format");
}


$boroughExists = count(getEveryRowWhereParamFromTable("boroughs", "borough_id",
$boroughId)) > 0;


if(!$boroughExists){
    echoUnprocessableEntity("Borough with provided id does not exist");
}
```

```
try{
    updateTextValue("boroughs", "borough_name", $boroughName, "borough_id",
$boroughId);
}


catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully edited borough";
http_response_code(200);
echo json_encode($result)
?>
```

editBuildingType.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["buildingTypeName"]))
{
    echoImproperRequest("All fields are required");
}


$buildingTypeName = $_POST["buildingTypeName"];
$buildingTypeId = $_POST["id"];
```

```
$reBuildingTypeName = '/^[A-Z][a-z\']{1,50}(\s[A-Za-z][a-z\']{1,50}){0,3}$/';


if(!preg_match($reBuildingTypeName, $buildingTypeName)){
    echoUnprocessableEntity("Building type does not match format");
}


$buildingTypeExists = count(getEveryRowWhereParamFromTable("buildingtypes",
"building_type_id", $buildingTypeId)) > 0;


if(!$buildingTypeExists){
    echoUnprocessableEntity("Building type with provided id does not exist");
}


try{
    updateTextValue("buildingtypes", "type_name", $buildingTypeName,
"building_type_id", $buildingTypeId);
}


catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully edited building type";
http_response_code(200);
echo json_encode($result)
?>
```

editLink.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");
```

```php
if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if
(
   (!isset($_POST["linkId"]))
|| (!isset($_POST["title"]))
|| (!isset($_POST["href"]))
|| (!isset($_POST["icon"]))
|| (!isset($_POST["aLevel"]))
|| (!isset($_POST["location"]))
|| (!isset($_POST["main"])
|| (!isset($_POST["priority"])))
)
{
    echoImproperRequest("All fields are required");
}


$LinkId = $_POST["linkId"];
$LinkTitle = $_POST["title"];
$LinkHref = $_POST["href"];
$LinkIcon = $_POST["icon"];
$AccessLevelId = $_POST["aLevel"];
$LinkLocation = $_POST["location"];
$main = $_POST["main"];
$priority = $_POST["priority"];


$reTitle = '/^[A-Z][a-z]{2,15}(\s[A-Za-z][a-z]{2,15}){0,2}$/';
$reHref = '/^(https?:\/\/(www\.)?[-a-zA-Z0-9@:%._\+~#=]{1,256}\.[-a-zA-Z0-
9()]{1,6}\b([-a-zA-Z0-9()@:%_\+.~#?&\/=]*))||([a-z]{3,40}\.[a-z]{2,5})$/';
$reIcon = '/^[a-z:-]{5,30}$/';


if(!preg_match($reTitle, $LinkTitle)){
    echoUnprocessableEntity("Link title does not match format");
```

```php
}


if(!preg_match($reHref, $LinkHref)){
    echoUnprocessableEntity("Link does not match format");
}


if(!empty($LinkIcon) && !preg_match($reIcon, $LinkIcon)){
    echoUnprocessableEntity("Link icon does not match format");
}


if($priority < 1){
    echoUnprocessableEntity("Link priority cannot be lower than 1");
}


if($priority > 99){
    echoUnprocessableEntity("Link priority cannot be higher than 99");
}


$linkExists = count(getEveryRowWhereParamFromTable("links", "link_id",
$LinkId)) > 0;


if(!$linkExists){
    echoUnprocessableEntity("Provided id link does not exist");
}


require("../DataAccess/navigationLocationFunctions.php");
$acceptableLocations = getAllNavigationLocations();


$locationAcceptable = in_array($LinkLocation, $acceptableLocations);


if(!$locationAcceptable){
    echoUnprocessableEntity("Invalid location provided");
}
```

```php
$acceptableALevelIds = getEveryParamFromTable("accesslevels",
"access_level_id");
$aLIdAcceptable = false;
foreach($acceptableALevelIds as $aLevelId){
    if($aLevelId["access_level_id"] == $AccessLevelId){
        $aLIdAcceptable = true;
        break;
    }
}


if(!$aLIdAcceptable){
    echoUnprocessableEntity("Invalid access level provided");
}


//Success
require("../DataAccess/linkFunctions.php");
$currentLinkIcon = "";
try{
    editLink($LinkId, $LinkTitle, $LinkHref, $AccessLevelId, $LinkLocation,
$priority, $main);
    if(!empty($LinkIcon)){
        $currentLinkIcon = getLinkIcon($LinkId);
        if($currentLinkIcon == ""){
            createNewLinkIcon($LinkId, $LinkIcon);
        }
        if($currentLinkIcon != $LinkIcon){
            removeAllLinkIcons($LinkId);
            createNewLinkIcon($LinkId, $LinkIcon);
        }
    }
    http_response_code(200);
    $result["general"] = "Successfully edited link";
    echo json_encode($result);
}
catch(PDOException $e){
    echoUnexpectedError();
}
?>
```

editListing.php

```php
<?php
session_start();
$requiredLevel = 3;
```

```php
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$result;


if
(!isset($_POST["listingTitle"])
|| !isset($_POST["listingDescription"])
|| !isset($_POST["listingId"])
|| !isset($_POST["listingAddress"])
|| !isset($_POST["listingSize"])
|| !isset($_POST["listingPrice"])
|| !isset($_POST["listingBorough"])
|| !isset($_POST["listingBuildingType"])
)
{
    echoImproperRequest("All fields are required");
}


$listingTitle = $_POST["listingTitle"];
$listingDescription = $_POST["listingDescription"];
$listingAddress = $_POST["listingAddress"];
$listingSize = $_POST["listingSize"];
$listingPrice = $_POST["listingPrice"];
$listingBorough = $_POST["listingBorough"];
$listingBuildingType = $_POST["listingBuildingType"];
$listingId = $_POST["listingId"];


$reTitle = '/^[A-Z][a-z]{2,15}(\s[A-Za-z][a-z]{2,15}){0,2}$/';
$reAddress = '/^(([A-Z][a-z\d\']+)|([0-9][1-9]*\.?))(\s[A-Za-z\d][a-
z\d\']+){0,7}\s(([1-9][0-9]{0,5}[\/-]?[A-Z])|([1-9][0-9]{0,5})|(NN))\.?$/';
$reDescription = '/^[A-Z][a-z\']{0,50}(\s[A-Za-z][a-z\']{0,50})*$/';


if(!preg_match($reTitle, $listingTitle)){
    echoUnprocessableEntity("Title does not match format");
}
if(!preg_match($reDescription, $listingDescription)){
    echoUnprocessableEntity("Description does not match format");
}
if(!preg_match($reAddress, $listingAddress)){
```

```
    echoUnprocessableEntity("Address does not match format", $listingAddress);
}


if($listingSize < 30){
    echoUnprocessableEntity("Size cannot be below 30 feet");
}


if($listingSize > 100000){
    echoUnprocessableEntity("Size cannot be above 100000 feet");
}


if($listingPrice < 1000){
    echoUnprocessableEntity("Price cannot be below 1000$");
}


if($listingPrice > 1000000000){
    echoUnprocessableEntity("Price cannot be above 1000000000$");
}
$imgUpload = false;
if(isset($_FILES["listingPhoto"])){
    $target_dir = "../resources/imgs/";
    $nameToSave = basename($_FILES["listingPhoto"]["name"]);
    $imageFileType = strtolower(pathinfo($nameToSave,PATHINFO_EXTENSION));
    $newFileName = time().uniqid(rand());
    $target_file = $target_dir.$newFileName.".".$imageFileType;

    $check = getimagesize($_FILES["listingPhoto"]["tmp_name"]);

    if($check === false) {
        echoUnprocessableEntity("Uploaded file is not an image");
    }

    if ($_FILES["listingPhoto"]["size"] > 8000000) {
        echoUnprocessableEntity("Uploaded file is too large");
    }

    if($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType !=
"jpeg") {
        echoUnprocessableEntity("Uploaded file of incorrect
type".$imageFileType);
    }
```

```
    $imgUpload = true;
}


$listingExists = count(getEveryRowWhereParamFromTable("listings", "listing_id",
$listingId)) > 0;
if(!$listingExists){
    echoUnprocessableEntity("Invalid listing selected");
}


$boroughExists = count(getEveryRowWhereParamFromTable("boroughs", "borough_id",
$listingBorough)) > 0;
if(!$boroughExists){
    echoUnprocessableEntity("Invalid borough selected");
}


$buildingTypeExists = count(getEveryRowWhereParamFromTable("buildingtypes",
"building_type_id", $listingBuildingType)) > 0;
if(!$buildingTypeExists){
    echoUnprocessableEntity("Invalid building type selected");
}


$rooms = array();
if(isset($_POST["listingRooms"])){
    $rooms = json_decode($_POST["listingRooms"]);
}


foreach($rooms as $room){
    //Optimize this by calling once to get every room_type_id, then checking
the current id against the list
    $roomExists = count(getEveryRowWhereParamFromTable("roomtypes",
"room_type_id", $room->roomId)) > 0;
    if(!$roomExists){
        echoUnprocessableEntity("Invalid room type selected");
    }
    if($room->count < 1){
        echoUnprocessableEntity("Number of any room cannot be below one");
    }
    if($room->count > 99){
        echoUnprocessableEntity("Number of any room cannot be above 99");
    }
```

```
}


require("../DataAccess/listingFunctions.php");
require("../DataAccess/imageFunctions.php");
try{
    $lastInsertedId = editListing($listingId, $listingBorough,
$listingBuildingType, $listingTitle, $listingDescription, $listingAddress,
$listingSize);
    $currentPrice = getPriceOfListing($listingId)["price"];
    //If current price different from submitted price
    if($currentPrice != $listingPrice){
        saveListingPrice($listingId, $listingPrice);
    }
    //If new image submitted
    if($imgUpload){
        //If successfully saved new image to disk, save it in the database
        $imgUploadSuccess = saveAdjustedPhotoToDisk($_FILES["listingPhoto"],
$target_file, 640, 360);
        if($imgUploadSuccess){
            saveMainListingPhoto($listingId, $newFileName.".".$imageFileType);
        }
    }
    $roomsSubmited = array_map(function($elem){
        return $elem->roomId;
    }, $rooms);
    $listingRooms = getRoomsOfListing($listingId);
    foreach($listingRooms as $room){
        //If room in list of rooms submitted, update count
        if(in_array($room["room_type_id"], $roomsSubmited)){
            $count = 0;
            foreach($rooms as $key => $r){
                if($r->roomId == $room["room_type_id"]){
                    $count = $r->count;
                    //Remove rooms who's value is only being changed from
submitted list of rooms
                    unset($rooms[$key]);
                    break;
                }
            }
            updateListingRoomCount($listingId, $room["room_type_id"], $count);
        }
        //If room not in the list of rooms submitted
        else{
            removeListingRoom($listingId, $room["room_type_id"]);
```

```
        }
    }
    //For every remaining submitted room
    foreach($rooms as $room){
        saveListingRoom($listingId, $room->roomId, $room->count);
    }
}
catch (PDOException $e){
    echoUnexpectedError();
}
$result["general"] = "Successfully edited listing";
http_response_code(200);
echo json_encode($result)
?>
```

editMessageType.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);



$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["messageTypeName"]) || !isset($_POST["id"]))
{
    echoImproperRequest("All fields are required");
}


$messageTypeName = $_POST["messageTypeName"];
$messageTypeId = $_POST["id"];
```

```php
$reMessageTypeName = '/^[A-Z][a-z]{1,19}$/';


if(!preg_match($reMessageTypeName, $messageTypeName)){
    echoUnprocessableEntity("Message type name has to be one capitalized
word");
}


$messageTypeExists = count(getEveryRowWhereParamFromTable("messagetypes",
"message_type_id", $messageTypeId)) > 0;


if(!$messageTypeExists){
    echoUnprocessableEntity("Message type with provided id does not exist");
}


try{
    updateTextValue("messagetypes", "message_type_name", $messageTypeName,
"message_type_id", $messageTypeId);
}


catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully edited message type";
http_response_code(200);
echo json_encode($result)
?>
```

editRoomType.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);
```

```php
$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["roomTypeName"]))
{
    echoImproperRequest("All fields are required");
}


$roomTypeName = $_POST["roomTypeName"];
$roomTypeId = $_POST["id"];


$reRoomTypeName = '/^[A-Z][a-z\']{1,50}(\s[A-Za-z][a-z\']{1,50}){0,3}$/';


if(!preg_match($reRoomTypeName, $roomTypeName)){
    echoUnprocessableEntity("Room type name does not match format");
}


$roomTypeExists = count(getEveryRowWhereParamFromTable("roomtypes",
"room_type_id", $roomTypeId)) > 0;


if(!$roomTypeExists){
    echoUnprocessableEntity("Room type with provided id does not exist");
}


try{
    updateTextValue("roomtypes", "room_name", $roomTypeName, "room_type_id",
$roomTypeId);
}
```

```
catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully edited room type";
http_response_code(200);
echo json_encode($result)
?>
```

editUser.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if
(
   (!isset($_POST["userId"]))
|| (!isset($_POST["email"]))
|| (!isset($_POST["name"]))
|| (!isset($_POST["lastName"]))
|| (!isset($_POST["roleId"]))
|| (!isset($_POST["password"]))
)
{
    echoImproperRequest("All fields are required");
}


$userId = $_POST["userId"];
```

```php
$email = $_POST["email"];
$name = $_POST["name"];
$lastName = $_POST["lastName"];
$roleId = $_POST["roleId"];
$password = $_POST["password"];


$reName = '/^[A-Z][a-z]{1,14}(\s[A-Z][a-z]{1,14}){0,2}$/';


$rePass1 = '/[A-Z]/';
$rePass2 = '/[a-z]/';
$rePass3 = '/[0-9]/';
$rePass4 = '/[!\?\.]/';
$rePass5 = '/^[A-Za-z0-9!\?\.]{7,30}$/';


$reEmail = '/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/';


if(!preg_match($reName, $name) || !preg_match($reName, $lastName))
{
    echoUnprocessableEntity("Name/last name does not fit criteria");
}
//If new password provided, check if valid
if(!empty($password)){
    if(!preg_match($rePass1, $password) || !preg_match($rePass2, $password)
    || !preg_match($rePass3, $password) || !preg_match($rePass4, $password)
    || !preg_match($rePass5, $password))
    {
        echoUnprocessableEntity("Password does not fit criteria");
    }
}


if(!preg_match($reEmail, $email)){
    echoUnprocessableEntity("Invalid email");
}


//Check if provided role exists
$roleExists = count(getEveryRowWhereParamFromTable("roles", "role_id",
$roleId)) > 0;
if(!$roleExists){
    echoUnprocessableEntity("Invalid role provided");
```

```
}


//Check if provided user exists
$userExists = count(getEveryRowWhereParamFromTable("users", "user_id",
$userId)) > 0;
if(!$userExists){
    echoUnprocessableEntity("Provided user does not exist");
}


//Success
require("../DataAccess/userFunctions.php");
try{
    editUser($userId, $email, $name, $lastName, $roleId);
    if(!empty($password)){
        editUserPassword($userId, $password);
    }
}
catch(PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully edited user";
http_response_code(200);
echo json_encode($result);
?>
```

getAllAccessLevels.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/accessLevelFunctions.php");
$result;


try{
    $result["general"] = getAllAccessLevels();
    http_response_code(200);
    echo json_encode($result);
```

```
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllBoroughs.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/boroughFunctions.php");
$result;


try{
    $result["general"] = getAllBoroughs();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllBoroughsCount.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/boroughFunctions.php");
$result;


try{
    $result["general"] = getAllBoroughsCount();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllBoroughsWithListings.php

```php
<?php
session_start();
require("../DataAccess/generalFunctions.php");


require("../DataAccess/boroughFunctions.php");
$result;


try{
    $result["general"] = getAllBoroughsWithListings();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllBuildingTypes.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");


checkAccessLevel($requiredLevel);


require("../DataAccess/buildingTypeFunctions.php");
$result;


try{
    $result["general"] = getAllBuildingTypes();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllBuildingTypesCount.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");


checkAccessLevel($requiredLevel);


require("../DataAccess/buildingTypeFunctions.php");
$result;


try{
    $result["general"] = getAllBuildingTypesCount();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllBuildingTypesWithListings.php

```php
<?php
session_start();
require("../DataAccess/generalFunctions.php");


require("../DataAccess/buildingTypeFunctions.php");
$result;


try{
    $result["general"] = getAllBuildingTypesWithListings();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllDeletedListings.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/listingFunctions.php");
$result;


try{
    $result["general"] = getAllDeletedListings();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllDeletedQuestions.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/surveyFunctions.php");
$result;


try{
    $result["general"] = getAllDeletedQuestions();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllLinks.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/linkFunctions.php");
$result;


try{
    $result["general"] = getAllLinks();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllListings.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/listingFunctions.php");
$result;


try{
    $result["general"] = getAllListings();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllMessages.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");


checkAccessLevel($requiredLevel);


require("../DataAccess/messageFunctions.php");
$result;


try{
    $result["general"] = getAllMessages();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllMessageTypes.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");


checkAccessLevel($requiredLevel);


require("../DataAccess/messageFunctions.php");
$result;


try{
    $result["general"] = getAllMessages();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllMessageTypesCount.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");


checkAccessLevel($requiredLevel);


require("../DataAccess/messageFunctions.php");
$result;


try{
    $result["general"] = getAllMessageTypesCount();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllNavigationLocations.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");


checkAccessLevel($requiredLevel);


require("../DataAccess/navigationLocationFunctions.php");
$result;


try{
    $result["general"] = getAllNavigationLocations();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
```

```
    echoUnexpectedError();
}
```

getAllQuestions.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/surveyFunctions.php");
$result;


try{
    $result["general"] = getAllQuestions();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllRoles.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


require("../DataAccess/userFunctions.php");
$result;


try{
    $result["general"] = getAllUserRoles();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllRoomTypes.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");


checkAccessLevel($requiredLevel);


require("../DataAccess/roomTypeFunctions.php");
$result;


try{
    $result["general"] = getAllRoomTypes();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getAllUsers.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);
require("../DataAccess/userFunctions.php");
$result;


try{
    $result["general"] = getAllUsers();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
?>
```

getLinks.php

```php
<?php
include("../DataAccess/linkFunctions.php");
include("../DataAccess/userFunctions.php");
require("../DataAccess/generalFunctions.php");


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


//Initialize access level as one
$accessLevel = 1;
$loggedIn = false;
$result;
$currentPage = $_POST["currentPage"];
$allowed = false;
session_start();
//If user is logged in, get their access level and set logged in to true
if(isset($_SESSION["user"])){
    $accessLevel = getUserLevel($_SESSION["user"]["user_id"])["level"];
    if($accessLevel != 1){
        $loggedIn = true;
    }
}
try{
    $result["general"]["links"] = getLinks($accessLevel, $loggedIn);
    $result["general"]["accessLevel"] = $accessLevel;
    //For every link found, check if it matches current page
    foreach($result["general"]["links"] as $link){
        if($link["href"] == $currentPage){
            $allowed = true;
            break;
        }
    }
    //If page is in list of pages available to user
    if($allowed){
        http_response_code(200);
        echo json_encode($result);
        die();
```

```
    }
    //If page is not available to user, echo 401 or 403
    else{
        if($loggedIn){
            //Echo 403
            echoNoPermission("You are not permitted to view this page");
        }
        else{
            //Echo 401
            echoUnauthorized();
        }
    }
}
catch (PDOException $e){
    echoUnexpectedError();
}


?>
```

getListingsForFilter.php

```
<?php
session_start();
require("../DataAccess/generalFunctions.php");


require("../DataAccess/listingFunctions.php");


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}



$result;


$listingTitleFilter = "";
$listingBuildingTypeFilter = array();
$listingBoroughFilter = array();
```

```php
$onlyFavorite = false;
$user_id = 0;
$sortType = 0;


if(isset($_POST["titleFilter"])){
    $listingTitleFilter = $_POST["titleFilter"];
}


if(isset($_POST["buildingTypeFilter"])){
    $listingBuildingTypeFilter = $_POST["buildingTypeFilter"];
}


if(isset($_POST["boroughFilter"])){
    $listingBoroughFilter = $_POST["boroughFilter"];
}


if(isset($_POST["onlyFavorite"]) && isset($_SESSION["user"])){
    $onlyFavorite = true;
}


if(isset($_POST["sortType"])){
    $sortType = $_POST["sortType"];
}



if(isset($_SESSION["user"])){
    $user_id = $_SESSION["user"]["user_id"];
}



try{
    $listings = getListingsForFilter($listingTitleFilter,
$listingBuildingTypeFilter, $listingBoroughFilter, $user_id, $onlyFavorite,
$sortType);
    $result["general"] = array();
    foreach($listings as $listing){
        $listingWithInformation["body"] = $listing;
```

```
        $listingWithInformation["img"] =
getCurrentMainListingPhoto($listing["id"])["path"];
        $listingWithInformation["rooms"] = getRoomsOfListing($listing["id"]);
        array_push($result["general"], $listingWithInformation);
    }
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getQuestionsForUser.php

```
<?php
session_start();
$requiredLevel = 2;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


require("../DataAccess/surveyFunctions.php");


try{
    $questions = getQuestions($_SESSION["user"]["user_id"]);
    $result["general"] = array();
    foreach($questions as $question){
        $questionWithAnswer["question"] = $question;
        $questionWithAnswer["answers"] = getQuestionAnswers($question["id"]);
        array_push($result["general"], $questionWithAnswer);
    }
    http_response_code(200);
    echo json_encode($result);
```

```
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getSpecificBorough.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid borough id");
}


$id = $_POST["id"];


require("../DataAccess/boroughFunctions.php");


try{
    $borough = getSpecificBorough($id);
    if(!$borough){
        echoNotFound();
    }
    $result["general"] = $borough;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getSpecificBuildingType.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid building type id");
}


$id = $_POST["id"];


require("../DataAccess/buildingTypeFunctions.php");


try{
    $buildingType = getSpecificBuildingType($id);
    if(!$buildingType){
        echoNotFound();
    }
    $result["general"] = $buildingType;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getSpecificDetailedListing.php

```php
<?php
session_start();
require("../DataAccess/generalFunctions.php");


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}



$result;


if(!isset($_POST["listing_id"])){
    echoImproperRequest("All fields are required");
}


$listing_id = $_POST["listing_id"];


$user_id = 0;



if(isset($_SESSION["user"])){
    $user_id = $_SESSION["user"]["user_id"];
}


require("../DataAccess/listingFunctions.php");
require("../DataAccess/informationFunctions.php");


try{
    $listing = getDetailedListing($listing_id, $user_id);
    if(!$listing){
        echoNotFound("Listing not found");
    }
```

```
    if(!$listing["active"]){
        echoGone("Listing no longer active");
    }
    $result["general"]["body"] = $listing;
    $result["general"]["img"] =
getCurrentMainListingPhoto($listing_id)["path"];
    $result["general"]["rooms"] = getRoomsOfListing($listing_id);
    $result["general"]["number"] = getListingPhoneNumber();
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getSpecificLink.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid link id");
}


$id = $_POST["id"];


require("../DataAccess/linkFunctions.php");
```

```php
try{
    $link = getSpecificLink($id);
    if(!$link){
        echoNotFound();
    }
    $result["general"] = $link;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getSpecificListing.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid listing id");
}


$id = $_POST["id"];


require("../DataAccess/listingFunctions.php");


try{
    $listing["main"] = getSpecificListing($id);
    if(!$listing["main"]){
        echoNotFound();
    }
    $listing["photo"] = getCurrentMainListingPhoto($id);
    $listing["rooms"] = getRoomsOfListing($id);
    $result["general"] = $listing;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getSpecificMessageType.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid message type id");
}


$id = $_POST["id"];


require("../DataAccess/messageFunctions.php");


try{
    $messageType = getSpecificMessageType($id);
    if(!$messageType){
        echoNotFound();
    }
    $result["general"] = $messageType;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

getSpecificQuestion.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["questionId"])){
    echoImproperRequest("Must provide a valid question id");
}


$questionId = $_POST["questionId"];


require("../DataAccess/surveyFunctions.php");


try{
    $questionExists = getSpecificQuestion($questionId);
    if(!$questionExists){
        echoNotFound();
    }
    $question["name"] = $questionExists["question"];
    $question["answers"] = getQuestionAnswers($questionId);
    $result["general"] = $question;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

[redacted]

getSpecificQuestionAnswers.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["questionId"])){
    echoImproperRequest("Must provide a valid question id");
}


$questionId = $_POST["questionId"];


require("../DataAccess/surveyFunctions.php");


try{
    $questionExists = getSpecificQuestion($questionId);
    if(!$questionExists){
        echoNotFound();
    }
    $question["name"] = $questionExists["question"];
    $question["answers"] = getQuestionAnswersCount($questionId);
    $result["general"] = $question;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
```

```
}
```

getSpecificRoomType.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid room type id");
}


$id = $_POST["id"];


require("../DataAccess/roomTypeFunctions.php");


try{
    $roomType = getSpecificRoomType($id);
    if(!$roomType){
        echoNotFound();
    }
    $result["general"] = $roomType;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

[redacted]

getSpecificUser.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid user id");
}


$id = $_POST["id"];


require("../DataAccess/userFunctions.php");


try{
    $user = getSpecificUser($id);
    if(!$user){
        echoNotFound();
    }
    $result["general"] = $user;
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

logout.php

```php
<?php
session_start();
$result;
require("../DataAccess/generalFunctions.php");
if(isset($_SESSION["user"])){
    session_unset();
    http_response_code(200);
    $result["general"] = "Successfully logged out";
    echo json_encode($result);
    die();
}
else{
    echoUnauthorized();
}
?>
```

removeFromFavoriteListings.php

```php
<?php
session_start();
$requiredLevel = 2;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel, "You must be logged in to add listings to
favorites");


$result;


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


if(!isset($_POST["listingId"])){
    echoImproperRequest("All fields are required");
}


$userId = $_SESSION["user"]["user_id"];
$listingId = $_POST["listingId"];


require("../DataAccess/favoriteFunctions.php");
try{
    deleteUserFavorite($userId, $listingId);
    $result["general"] = "Successfully removed listing from favorites";
    http_response_code(200);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

restoreListing.php

```php
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid listing id");
}


$id = $_POST["id"];


$exists = count(getEveryRowWhereParamFromTable("listings", "listing_id", $id))
> 0;


if(!$exists){
    echoUnprocessableEntity("Provided listing id does not exist");
}


require("../DataAccess/listingFunctions.php");


try{
    restoreListing($id);
}
catch (PDOException $e){
    echoUnexpectedError();
```

```
}


$result["general"] = "Successfully restored listing";
http_response_code(200);
echo json_encode($result);
```

restoreQuestion.php

```
<?php
session_start();
$requiredLevel = 3;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["id"])){
    echoImproperRequest("Must provide a valid question id");
}


$id = $_POST["id"];


$exists = count(getEveryRowWhereParamFromTable("questions", "question_id",
$id)) > 0;


if(!$exists){
    echoUnprocessableEntity("Provided question id does not exist");
}


require("../DataAccess/surveyFunctions.php");
```

```
try{
    restoreQuestion($id);
}
catch (PDOException $e){
    echoUnexpectedError();
}


$result["general"] = "Successfully restored question";
http_response_code(200);
echo json_encode($result);
```

submitAnswer.php

```
<?php
session_start();
$requiredLevel = 2;
require("../DataAccess/generalFunctions.php");
checkAccessLevel($requiredLevel);


$json_params = file_get_contents("php://input");


if (strlen($json_params) > 0 && isValidJSON($json_params)){
    $decoded_params = json_decode($json_params, true);
    $_POST = $decoded_params;
}


$result;


if(!isset($_POST["answerId"])){
    echoImproperRequest("All fields are required");
}


$answerId = $_POST["answerId"];


require("../DataAccess/surveyFunctions.php");
```

```php
$userAllowedToAnswer = checkIfUserAllowedToAnswer($_SESSION["user"]["user_id"],
$answerId);


if(!$userAllowedToAnswer){
    echoUnprocessableEntity("You already answered this question");
}


try{
    saveUserAnswer($_SESSION["user"]["user_id"], $answerId);
    $result["general"] = "Successfully answered question";
    http_response_code(201);
    echo json_encode($result);
}
catch (PDOException $e){
    echoUnexpectedError();
}
```

3.1.2 PHP Files in /DataAccess

accessLevelFunctions.php

```php
<?php
function getAllAccessLevels(){
    include ("../../connection.php");


    $statement = "SELECT access_level_id AS id, level_title as title FROM
accesslevels ORDER BY level";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
?>
```

boroughFunctions.php

```php
<?php
function getAllBoroughs(){
    include ("../../connection.php");
```

```
    $statement = "SELECT borough_id AS id, borough_name as title FROM boroughs
ORDER BY borough_name";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getAllBoroughsCount(){
    include ("../../connection.php");


    $statement = "SELECT b.borough_id AS id, borough_name as title,
COUNT(l.listing_id) as Count
                  FROM boroughs b LEFT JOIN listings l ON b.borough_id =
l.borough_id
                  GROUP BY b.borough_id, borough_name
                  ORDER BY COUNT(l.listing_id) DESC";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getAllBoroughsWithListings(){
    include ("../../connection.php");


    $statement = "SELECT b.borough_id AS id, borough_name as title,
COUNT(l.listing_id) as Count
                  FROM boroughs b INNER JOIN listings l ON b.borough_id =
l.borough_id
                  WHERE l.dateDeleted IS NULL
                  GROUP BY b.borough_id, borough_name
                  ORDER BY COUNT(l.listing_id) DESC";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();
```

```
    return $prepSt->fetchAll();
}
function getSpecificBorough($id){
    include ("../../connection.php");


    $statement = "SELECT borough_name AS title FROM boroughs
                  WHERE borough_id = :borough_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("borough_id", $id, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetch();
}
?>
```

buildingTypeFunctions.php

```
<?php
function getAllBuildingTypes(){
    include ("../../connection.php");


    $statement = "SELECT building_type_id AS id, type_name as title FROM
buildingtypes ORDER BY type_name";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getAllBuildingTypesCount(){
    include ("../../connection.php");


    $statement = "SELECT bt.building_type_id AS id, type_name as title,
COUNT(l.listing_id) as Count
                  FROM buildingtypes bt LEFT JOIN listings l ON
bt.building_type_id = l.building_type_id
```

```
                    GROUP BY bt.building_type_id, type_name
                    ORDER BY COUNT(l.listing_id) DESC";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getAllBuildingTypesWithListings(){
    include ("../../connection.php");


    $statement = "SELECT bt.building_type_id AS id, type_name as title,
COUNT(l.listing_id) as Count
                    FROM buildingtypes bt INNER JOIN listings l ON
bt.building_type_id = l.building_type_id
                    WHERE l.dateDeleted IS NULL
                    GROUP BY bt.building_type_id, type_name
                    ORDER BY COUNT(l.listing_id) DESC";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getSpecificBuildingType($id){
    include ("../../connection.php");


    $statement = "SELECT type_name AS title FROM buildingtypes
                    WHERE building_type_id = :building_type_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("building_type_id", $id, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetch();
```

```
}
?>
```

favoriteFunctions.php

```
<?php
function saveUserFavorite($user_id, $listing_id){
    include ("../../connection.php");


    $statement = "INSERT INTO favorites (user_id, listing_id) VALUES (:user_id,
:listing_id)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("user_id", $user_id, PDO::PARAM_INT);
    $prepSt->bindParam("listing_id", $listing_id, PDO::PARAM_INT);


    $prepSt->execute();


    return $conn->lastInsertId();
}
function deleteUserFavorite($user_id, $listing_id){
    include ("../../connection.php");


    $statement = "DELETE FROM favorites WHERE user_id = :user_id AND listing_id
= :listing_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("user_id", $user_id, PDO::PARAM_INT);
    $prepSt->bindParam("listing_id", $listing_id, PDO::PARAM_INT);


    $prepSt->execute();


    return $conn->lastInsertId();
}
function checkIfAlreadyFavorite($user_id, $listing_id){
    include ("../../connection.php");
```

```
    $statement = "SELECT user_id FROM favorites WHERE user_id = :user_id AND
listing_id = :listing_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("user_id", $user_id, PDO::PARAM_INT);
    $prepSt->bindParam("listing_id", $listing_id, PDO::PARAM_INT);


    $prepSt->execute();


    $result = $prepSt->fetch();


    if($result){
        return true;
    }
    return false;
}
?>
```

generalFunctions.php

```
<?php
function echoUnprocessableEntity($message, $input = ""){
    $result["error"] = $message.$input;
    http_response_code(422);
    echo json_encode($result);
    die();
}
function echoImproperRequest($message = "All fields are required"){
    $result["error"] = $message;
    http_response_code(400);
    echo json_encode($result);
    die();
}
function echoUnexpectedError($e = "An unexpected error occured"){
    $result["error"] = $e;
    http_response_code(500);
    echo json_encode($result);
    die();
}
function echoUnauthorized($e = "You have to log in first"){
    $result["error"] = $e;
    http_response_code(401);
```

```php
    echo json_encode($result);
    die();
}
function echoNoPermission($e = "You are not permitted this action"){
    $result["error"] = $e;
    http_response_code(403);
    echo json_encode($result);
    die();
}
function echoNotFound($e = "Not found"){
    $result["error"] = $e;
    http_response_code(404);
    echo json_encode($result);
    die();
}
function echoGone($e = "No longer exists"){
    $result["error"] = $e;
    http_response_code(410);
    echo json_encode($result);
    die();
}
function getUserLevel($id){
    include ("../../connection.php");


    $statement = "SELECT user_id, level
    FROM users u
    INNER JOIN roles r ON u.role_id = r.role_id
    INNER JOIN accesslevels al ON r.access_level_id = al.access_level_id
    WHERE user_id = ? AND level > 0";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $id, PDO::PARAM_INT);


    $prepSt->execute();
    $result = $prepSt->fetch();


    if($prepSt->rowCount() > 0){
        return $result;
    }
    else{
        $result = array("user_id" => 0, "level" => 1);
```

```
        session_unset();
        return $result;
    }
}
function checkAccessLevel($requiredLevel, $e = "You are not permitted this
action"){
    //If user not logged in, die and return 401
    if(!isset($_SESSION["user"])){
        echoUnauthorized($e);
    }


    //If user's access level is too low, die and return 403
    if(getUserLevel($_SESSION["user"]["user_id"])["level"] < $requiredLevel){
        echoNoPermission($e);
    }
}
function isValidJSON($str) {
    json_decode($str);
    return json_last_error() == JSON_ERROR_NONE;
}
function updateTextValue($table,$toChange, $toChangeValue, $paramater,
$paramaterValue){
    include ("../../connection.php");


    $statement = "UPDATE $table SET $toChange = ?
                  WHERE $paramater = ?";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $toChangeValue);
    $prepSt->bindParam(2, $paramaterValue, PDO::PARAM_INT);


    $prepSt->execute();
}


function deleteSingleRow($table, $paramater, $paramaterValue){
    include ("../../connection.php");


    $statement = "DELETE FROM $table WHERE $paramater = ?";
    $prepSt = $conn->prepare($statement);
```

```php
    $prepSt->bindParam(1, $paramaterValue, PDO::PARAM_INT);


    $prepSt->execute();


}


function softDeleteSingleRow($table, $parameter, $parameterValue){
    include ("../../connection.php");


    $statement = "UPDATE $table SET dateDeleted = NOW() WHERE $parameter = ?";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $parameterValue, PDO::PARAM_INT);


    $prepSt->execute();
}



function insertSingleParamater($table, $paramater, $paramaterName){
    include ("../../connection.php");


    $statement = "INSERT INTO $table ($paramaterName)
    VALUES (?)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $paramater);

    $prepSt->execute();


    return $conn->lastInsertId();
}
```

```php
function getEverythingFromTable($table){
    include ("../../connection.php");


    $statement = "SELECT * FROM $table";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getEveryParamFromTable($table, $param){
    include ("../../connection.php");


    $statement = "SELECT $param FROM $table";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getEveryRowWhereParamFromTable($table, $param, $value){
    include ("../../connection.php");


    $statement = "SELECT $param FROM $table WHERE $param = ?";
    $prepSt = $conn->prepare($statement);
    $prepSt->bindParam(1, $value, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
?>
```

imageFunctions.php

```php
<?php
function saveAdjustedPhotoToDisk($image, $targetFile, $maxW, $maxH){
```

```
    //Get information from about provided
    $file_name = $image['tmp_name'];
    list($width, $height, $type, $attr) = getimagesize( $file_name );
    /*/If the width is greater than the height, calculate scaling
    ratio based on width, else calculated based on height */
    if($width > $height){
        $ratio = $maxW / $width;
    }
    else{
        $ratio = $maxH / $height;
    }
    $target_filename = $file_name;
    //Set the initial values of new width and new height to the current ones in
case no resizing is to be done.
    $new_width = $width;
    $new_height = $height;
    //If image would get smaller by multiplying with the calculated ratio,
multiply.
    if($ratio < 1){
        $new_width = $width*$ratio;
        $new_height = $height*$ratio;
    }
    $src = imagecreatefromjpeg($file_name);
    $dst = imagecreatetruecolor($new_width, $new_height);
    //Copy image onto image of rescaled size, ie make the image resized
    imagecopyresampled($dst, $src, 0, 0, 0, 0,  $new_width, $new_height,
$width, $height);
    //Make a new image of desired maximum size (Background image)
    $newDst = imagecreatetruecolor($maxW, $maxH);
    //Change the color of the background image
    $bg = imagecolorallocate ($newDst, 31, 39, 27);
    imagefilledrectangle($newDst, 0, 0, $maxW, $maxH, $bg);
    //Merge the resized image onto the background image, positioning it to the
center in case of smaller width
    imagecopymerge($newDst, $dst, ($maxW - $new_width) / 2, 0, 0, 0,
$new_width, $new_height, 100);
    //Save the new image as a jpeg
    $result = imagejpeg($newDst, $targetFile);
    //Destory the source image
    imagedestroy($src);
    return $result;
}
?>
```

informationFunctions.php

```php
<?php
function getListingPhoneNumber(){
    return "+1 202-918-2132";
}
?>
```

linkFunctions.php

```php
<?php
function getLinks($accessLevel, $loggedIn){
    include ("../../connection.php");


    $statement = "SELECT link_title, href, landing, location, parent_id, level,
(SELECT icon FROM linkicons WHERE link_id = l.link_id AND active = 1) as icon
    FROM links l
    INNER JOIN accesslevels a ON l.access_level_id = a.access_level_id
    WHERE level <= ?";


    if($loggedIn){
        $statement.= " AND level <> 0";
    }


    $statement.=" ORDER BY l.priority DESC";


    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $accessLevel, PDO::PARAM_INT);


    $prepSt->execute();
    $results = $prepSt->fetchAll();


    return $results;
}
function getAllLinks(){
    include ("../../connection.php");
```

```
    $statement = "SELECT l.link_id AS id, link_title, level_title, href,
IF(landing, \"Root\", \"Pages\"), location, priority, IFNULL((SELECT link_title
FROM links WHERE link_id = l.parent_id),\"None\") AS Parent,
                  IFNULL((SELECT icon FROM linkicons WHERE link_id = l.link_id
AND active = 1), \"None\") AS icon
                  FROM links l
                  INNER JOIN accesslevels a ON l.access_level_id =
a.access_level_id
                  ORDER BY location DESC, priority DESC
                  ";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();
    $results = $prepSt->fetchAll();


    return $results;
}
function getSpecificLink($linkId){
    include ("../../connection.php");


    $statement = "SELECT link_title as title, href, (SELECT icon FROM linkicons
WHERE link_id = l.link_id AND active = 1) as icon, access_level_id, location,
priority, landing FROM links l
                  WHERE l.link_id = :link_id";
    $prepSt = $conn->prepare($statement);
    $prepSt->bindParam("link_id", $linkId);


    $prepSt->execute();
    $results = $prepSt->fetch();


    return $results;
}
function createNewLink($title, $href, $aLevel, $location, $landing, $priority){
    include ("../../connection.php");


    $statement = "INSERT INTO links (link_title, href, access_level_id,
location, landing, priority) VALUES(?, ?, ?, ?, ?, ?)";
    $prepSt = $conn->prepare($statement);
```

```
    $prepSt->bindParam(1, $title);
    $prepSt->bindParam(2, $href);
    $prepSt->bindParam(3, $aLevel, PDO::PARAM_INT);
    $prepSt->bindParam(4, $location);
    $prepSt->bindParam(5, $landing, PDO::PARAM_INT);
    $prepSt->bindParam(6, $priority, PDO::PARAM_INT);


    $prepSt->execute();


    return $conn->lastInsertId();
}
function editLink($linkId, $title, $href, $aLevel, $location, $priority,
$landing){
    include ("../../connection.php");


    $statement = "UPDATE links SET link_title = :title, href = :href,
access_level_id = :aLevel, location = :location, priority = :priority, landing
= :landing
                  WHERE link_id = :linkId";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("title", $title);
    $prepSt->bindParam("href", $href);
    $prepSt->bindParam("aLevel", $aLevel, PDO::PARAM_INT);
    $prepSt->bindParam("location", $location);
    $prepSt->bindParam("priority", $priority, PDO::PARAM_INT);
    $prepSt->bindParam("landing", $landing, PDO::PARAM_INT);
    $prepSt->bindParam("linkId", $linkId, PDO::PARAM_INT);


    $return = $prepSt->execute();


    return $return;
}
function createNewLinkIcon($linkId, $icon){
    include ("../../connection.php");
```

```php
    $statement = "INSERT INTO linkicons (link_id, icon) VALUES(?,?)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $linkId, PDO::PARAM_INT);
    $prepSt->bindParam(2, $icon);


    $prepSt->execute();


    return $conn->lastInsertId();
}
function getLinkIcon($link_id){
    include ("../../connection.php");


    $statement = "SELECT icon FROM linkicons
                  WHERE link_id = :link_id AND active = 1
                  ORDER BY dateCreated DESC
                  LIMIT 1";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("link_id", $link_id, PDO::PARAM_INT);


    $prepSt->execute();


    $result = $prepSt->fetch();


    if($result){
        return $result["icon"];
    }
    else{
        return "";
    }
}
function removeAllLinkIcons($linkId){
    include ("../../connection.php");
```

```
    $statement = "UPDATE linkicons SET active = 0 WHERE link_id = :linkId AND
active = 1";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("linkId", $linkId);


    return $prepSt->execute();
}
?>
```

listingFunctions.php

```
<?php
function createNewListing($borough, $building_type, $name, $description,
$address, $size){
    include ("../../connection.php");


    $statement = "INSERT INTO listings (borough_id, building_type_id,
listing_name, description, address, size)
                  VALUES (?, ?, ?, ?, ?, ?)";
    $prepSt = $conn->prepare($statement);

    $prepSt->bindParam(1, $borough);
    $prepSt->bindParam(2, $building_type);
    $prepSt->bindParam(3, $name);
    $prepSt->bindParam(4, $description);
    $prepSt->bindParam(5, $address);
    $prepSt->bindParam(6, $size);


    $prepSt->execute();


    $last_id = $conn->lastInsertId();

    return $last_id;
}


function editListing($listing_id, $borough_id, $building_type_id, $name,
$description, $address, $size){
    include ("../../connection.php");
```

```
    $statement = "UPDATE listings SET borough_id = :borough_id,
building_type_id = :building_type_id, listing_name = :name,
                  description = :description, address = :address, size = :size
                  WHERE listing_id = :listing_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("listing_id", $listing_id, PDO::PARAM_INT);
    $prepSt->bindParam("borough_id", $borough_id, PDO::PARAM_INT);
    $prepSt->bindParam("building_type_id", $building_type_id, PDO::PARAM_INT);
    $prepSt->bindParam("name", $name);
    $prepSt->bindParam("description", $description);
    $prepSt->bindParam("address", $address);
    $prepSt->bindParam("size", $size);


    return $prepSt->execute();
}


function saveListingPrice($listing, $price){
    include ("../../connection.php");


    $statement = "INSERT INTO listingprices (listing_id, price) VALUES (?, ?)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $listing, PDO::PARAM_INT);
    $prepSt->bindParam(2, $price);


    $result = $prepSt->execute();


    return $result;
}


function saveMainListingPhoto($listing, $path){
    include ("../../connection.php");
```

```php
    $main = true;


    $statement = "INSERT INTO listingphotos (listing_id, path, main) VALUES (?,
?, ?)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $listing, PDO::PARAM_INT);
    $prepSt->bindParam(2, $path);
    $prepSt->bindParam(3, $main);


    $result = $prepSt->execute();


    return $result;
}


function saveListingRoom($listing, $room, $count){
    include ("../../connection.php");


    $statement = "INSERT INTO listingrooms (listing_id, room_type_id, numberOf)
VALUES (?, ?, ?)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $listing, PDO::PARAM_INT);
    $prepSt->bindParam(2, $room, PDO::PARAM_INT);
    $prepSt->bindParam(3, $count, PDO::PARAM_INT);


    $result = $prepSt->execute();


    return $result;
}


function updateListingRoomCount($listing_id, $room_type_id, $numberOf){
    include ("../../connection.php");
```

```
    $statement = "UPDATE listingrooms SET numberOf = :numberOf
                  WHERE listing_id = :listing_id AND room_type_id =
:room_type_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("listing_id", $listing_id, PDO::PARAM_INT);
    $prepSt->bindParam("room_type_id", $room_type_id, PDO::PARAM_INT);
    $prepSt->bindParam("numberOf", $numberOf, PDO::PARAM_INT);


    return $prepSt->execute();
}


function removeListingRoom($listing_id, $room_type_id){
    include ("../../connection.php");


    $statement = "DELETE FROM listingrooms
                  WHERE listing_id = :listing_id AND room_type_id =
:room_type_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("listing_id", $listing_id, PDO::PARAM_INT);
    $prepSt->bindParam("room_type_id", $room_type_id, PDO::PARAM_INT);


    return $prepSt->execute();
}


function updateMainListingPhoto($listing, $path){
    include ("../../connection.php");


    $statement = "UPDATE listingphotos SET path = ? WHERE listing_id = ? AND
main = true";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $path);
    $prepSt->bindParam(2, $listing, PDO::PARAM_INT);
```

```php
    $result = $prepSt->execute();

    return $result;
}

function getCurrentMainListingPhoto($listing){
    include ("../../connection.php");

    $statement = "SELECT path FROM listingphotos
                  WHERE main = 1 AND listing_id = :listing_id
                  ORDER BY dateUploaded DESC
                  LIMIT 1";
    $prepSt = $conn->prepare($statement);

    $prepSt->bindParam("listing_id", $listing, PDO::PARAM_INT);

    $prepSt->execute();
    $result = $prepSt->fetch();

    return $result;
}

function getRoomsOfListing($listing){
    include ("../../connection.php");

    $statement = "SELECT rt.room_name, rt.room_type_id, lr.numberOf
                  FROM roomtypes rt
                  INNER JOIN listingrooms lr ON rt.room_type_id =
lr.room_type_id
                  INNER JOIN listings li ON lr.listing_id = li.listing_id
                  WHERE li.listing_id = :listing_id";
    $prepSt = $conn->prepare($statement);

    $prepSt->bindParam("listing_id", $listing, PDO::PARAM_INT);
```

```
    $prepSt->execute();
    $result = $prepSt->fetchAll();


    return $result;
}


function getSpecificListing($listing){
    include ("../../connection.php");


    $statement = "SELECT listing_name, description, (SELECT price FROM
listingprices WHERE listing_id = :listing_id ORDER BY date DESC LIMIT 1) as
price, size, address, borough_id, building_type_id
                FROM listings li
                WHERE listing_id = :listing_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("listing_id", $listing, PDO::PARAM_INT);


    $prepSt->execute();
    $result = $prepSt->fetch();


    return $result;
}


function getAllListings(){
    include ("../../connection.php");


    $statement = "SELECT l.listing_id AS id, listing_name, price, description,
b.borough_name, bt.type_name, address, size
                FROM listings l INNER JOIN listingprices lp ON l.listing_id =
lp.listing_id
                INNER JOIN boroughs b on l.borough_id = b.borough_id
                INNER JOIN buildingtypes bt ON l.building_type_id =
bt.building_type_id
```

```
                    WHERE lp.date = (SELECT MAX(date) FROM listingprices WHERE
listing_id = l.listing_id)
                    AND l.dateDeleted IS NULL
                    ORDER BY l.listing_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();
    $result = $prepSt->fetchAll();


    return $result;
}


function getDetailedListing($listing_id, $user_id){
    include ("../../connection.php");


    if($user_id != 0){
        $statement = "SELECT l.listing_id AS id, listing_name, b.borough_name
AS borough, b.borough_id AS borough_id, bt.building_type_id AS type_id,
bt.type_name AS Type, price, description, address, size,
        (
        SELECT COUNT(*) AS list FROM favorites WHERE user_id = :user_id AND
listing_id = l.listing_id
        ) AS favorite, IF(l.dateDeleted IS NULL, true, false) AS active
        FROM listings l
        INNER JOIN listingprices lp ON l.listing_id = lp.listing_id
        INNER JOIN boroughs b on l.borough_id = b.borough_id
        INNER JOIN buildingtypes bt ON l.building_type_id = bt.building_type_id
        LEFT JOIN favorites f ON l.listing_id = f.listing_id
        ";
    }
    else{
        $statement = "SELECT l.listing_id AS id, listing_name, b.borough_name
AS borough, bt.type_name AS Type, price, description, address, size, 0 AS
favorite,
        IF(l.dateDeleted IS NULL, true, false) AS active
        FROM listings l
        INNER JOIN listingprices lp ON l.listing_id = lp.listing_id
        INNER JOIN boroughs b on l.borough_id = b.borough_id
        INNER JOIN buildingtypes bt ON l.building_type_id = bt.building_type_id
        ";
    }
```

```
    $statement .=
    "WHERE lp.date = (SELECT MAX(date) FROM listingprices WHERE listing_id =
l.listing_id)
    AND l.listing_id = :listing_id";


    $prepSt = $conn->prepare($statement);


    if($user_id != 0){
        $prepSt->bindParam(":user_id", $user_id, PDO::PARAM_INT);
    }


    $prepSt->bindParam(":listing_id", $listing_id, PDO::PARAM_INT);


    $prepSt->execute();


    $result = $prepSt->fetch();


    return $result;
}


function getListingsForFilter($listingTitleFilter, $listingBuildingTypeFilter,
$listingBoroughFilter, $user_id, $userFavoriteFilter, $sortType){
    include ("../../connection.php");


    if($user_id != 0){
        $statement = "SELECT l.listing_id AS id, listing_name, b.borough_name
AS borough, bt.type_name AS Type, price, description, address, size,
        (
        SELECT COUNT(*) AS list FROM favorites WHERE user_id = :user_id AND
listing_id = l.listing_id
        ) AS favorite
        FROM listings l
        INNER JOIN listingprices lp ON l.listing_id = lp.listing_id
        INNER JOIN boroughs b on l.borough_id = b.borough_id
        INNER JOIN buildingtypes bt ON l.building_type_id = bt.building_type_id
```

```
        LEFT JOIN favorites f ON l.listing_id = f.listing_id
        ";
    }
    else{
        $statement = "SELECT l.listing_id AS id, listing_name, b.borough_name
AS borough, bt.type_name AS Type, price, description, address, size, 0 AS
favorite
        FROM listings l
        INNER JOIN listingprices lp ON l.listing_id = lp.listing_id
        INNER JOIN boroughs b on l.borough_id = b.borough_id
        INNER JOIN buildingtypes bt ON l.building_type_id = bt.building_type_id
        ";
    }



    $statement .=
    "WHERE lp.date = (SELECT MAX(date) FROM listingprices WHERE listing_id =
l.listing_id)
    AND l.dateDeleted IS NULL ";



    $titleFilter = false;
    $buildingTypeFilter = false;
    $boroughFilter = false;



    if($listingTitleFilter != ""){
        $titleFilter = true;
        $statement .= " AND l.listing_name LIKE :listingTitleFilter";
    }



    if(count($listingBuildingTypeFilter) > 0){
        $buildingTypeFilter = true;
        $counter = 0;
        $placeholders = "";
        for($i = 0; $i < count($listingBuildingTypeFilter) - 1; $i++){
            $tag = ":lbuildingtype".$counter++;
            $placeholders .= $tag.", ";
        }
        $tag = ":lbuildingtype".$counter++;
        $placeholders .= $tag;
        $statement .= " AND l.building_type_id IN ($placeholders)";
    }
```

```php
    if(count($listingBoroughFilter) > 0){
        $boroughFilter = true;
        $counter = 0;
        $placeholders = "";
        for($i = 0; $i < count($listingBoroughFilter) - 1; $i++){
            $tag =":lborough".$counter++;
            $placeholders .= $tag.", ";
        }
        $tag =":lborough".$counter++;
        $placeholders .= $tag;
        $statement .= " AND l.borough_id IN ($placeholders)";
    }


    if($userFavoriteFilter && $user_id != 0){
        $statement .= " AND f.user_id = :user_id";
    }


    $sort = "";


    $sortTypes = array(0 => "l.listing_id DESC", 1 => "price ASC", 2 => "price
DESC", 3 => "size ASC", 4 => "size DESC");


    $statement .= " ORDER BY ".$sortTypes[$sortType];


    $prepSt = $conn->prepare($statement);


    if($titleFilter){
        $listingTitleFilter = "%".$listingTitleFilter."%";
        $prepSt->bindParam(":listingTitleFilter", $listingTitleFilter);
    }


    if($buildingTypeFilter){
        for($i = 0; $i < count($listingBuildingTypeFilter); $i++){
            $tag = ':lbuildingtype'.$i;
            $prepSt->bindValue($tag, $listingBuildingTypeFilter[$i],
PDO::PARAM_INT);
```

```
        }
    }


    if($boroughFilter){
        for($i = 0; $i < count($listingBoroughFilter); $i++){
            $tag = ':lborough'.$i;
            $prepSt->bindValue($tag, $listingBoroughFilter[$i],
PDO::PARAM_INT);
        }
    }

    if($user_id != 0){
        $prepSt->bindParam(":user_id", $user_id, PDO::PARAM_INT);
    }


    $prepSt->execute();
    $result = $prepSt->fetchAll();
    return $result;
}


function getPriceOfListing($listing_id){
    include ("../../connection.php");


    $statement = "SELECT price FROM listingprices WHERE
                  listing_id = :listing_id ORDER BY date DESC LIMIT 1";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("listing_id", $listing_id, PDO::PARAM_INT);


    $prepSt->execute();
    $result = $prepSt->fetch();


    return $result;
}


function getAllDeletedListings(){
    include ("../../connection.php");
```

```
    $statement = "SELECT l.listing_id AS id, listing_name, price, description,
b.borough_name, bt.type_name, address, size
                  FROM listings l INNER JOIN listingprices lp ON l.listing_id =
lp.listing_id
                  INNER JOIN boroughs b on l.borough_id = b.borough_id
                  INNER JOIN buildingtypes bt ON l.building_type_id =
bt.building_type_id
                  WHERE lp.date = (SELECT MAX(date) FROM listingprices WHERE
listing_id = l.listing_id)
                  AND l.dateDeleted IS NOT NULL
                  ORDER BY l.listing_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();
    $result = $prepSt->fetchAll();


    return $result;
}


function restoreListing($id){
    include ("../../connection.php");


    $statement = "UPDATE listings SET dateDeleted = NULL
                  WHERE listing_id = :listing_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("listing_id", $id, PDO::PARAM_INT);


    return $prepSt->execute();
}
?>
```

messageFunctions.php

```
<?php
function getAllMessageTypes(){
    include ("../../connection.php");
```

```
    $statement = "SELECT message_type_id AS id, message_type_name as title FROM
messagetypes ORDER BY message_type_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getAllMessageTypesCount(){
    include ("../../connection.php");


    $statement = "SELECT mt.message_type_id AS id, message_type_name AS title,
COUNT(m.message_id) AS Count
                  FROM messagetypes mt LEFT JOIN messages m on
mt.message_type_id = m.message_type_id
                  GROUP BY mt.message_type_id, message_type_name
                  ORDER BY COUNT(m.message_id) DESC";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getAllMessages(){
    include ("../../connection.php");


    $statement = "SELECT m.message_id AS id, email, message_type_name, title,
message, m.dateCreated
                  FROM messages m
                  INNER JOIN messagetypes mt ON m.message_type_id =
mt.message_type_id
                  INNER JOIN users u ON u.user_id = m.user_id
                  ORDER BY m.dateCreated DESC";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();
```

```
    return $prepSt->fetchAll();
}
function getSpecificMessageType($id){
    include ("../../connection.php");


    $statement = "SELECT message_type_name AS title FROM messagetypes
                  WHERE message_type_id = :message_type_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("message_type_id", $id, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetch();
}
function createNewMessage($user_id, $message_type_id, $title, $message){
    include ("../../connection.php");


    $statement = "INSERT INTO messages (user_id, message_type_id, title,
message) VALUES (:user_id, :message_type_id, :title, :message)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("user_id", $user_id, PDO::PARAM_INT);
    $prepSt->bindParam("message_type_id", $message_type_id, PDO::PARAM_INT);
    $prepSt->bindParam("title", $title);
    $prepSt->bindParam("message", $message);


    return $prepSt->execute();
}
function createNewMessageType($message_type_name){
}
?>
```

navigationLocationFunctions.php

```
<?php
function getAllNavigationLocations(){
    return array("head", "navbar", "footer", "hidden");
}
```

```
?>
```

roomTypeFunctions.php

```
<?php
function getAllRoomTypes(){
    include ("../../connection.php");


    $statement = "SELECT room_type_id AS id, room_name as title FROM roomtypes
ORDER BY room_name";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getSpecificRoomType($id){
    include ("../../connection.php");


    $statement = "SELECT room_name AS title FROM roomtypes
                  WHERE room_type_id = :room_type_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("room_type_id", $id, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetch();
}
?>
```

surveyFunctions.php

```php
<?php
function getAllQuestions(){
    include ("../../connection.php");


    $statement = "SELECT q.question_id AS id, q.question, COUNT(ua.user_id) as
Count FROM questions q
                  LEFT JOIN answers a ON a.question_id = q.question_id
                  LEFT JOIN useranswers ua ON a.answer_id = ua.answer_id
                  WHERE q.dateDeleted IS NULL
                  GROUP BY q.question_id, q.question";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    $result = $prepSt->fetchAll();


    return $result;
}
function getAllDeletedQuestions(){
    include ("../../connection.php");


    $statement = "SELECT q.question_id AS id, q.question, COUNT(ua.user_id) as
Count FROM questions q
    LEFT JOIN answers a ON a.question_id = q.question_id
    LEFT JOIN useranswers ua ON a.answer_id = ua.answer_id
    WHERE q.dateDeleted IS NOT NULL
    GROUP BY q.question_id, q.question";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    $result = $prepSt->fetchAll();


    return $result;
}
function getQuestions($user_id){
```

```php
    include ("../../connection.php");


    $statement = "SELECT q.question_id as id, q.question FROM questions q
                  WHERE q.dateDeleted IS NULL AND q.question_id NOT IN
                  (SELECT question_id FROM useranswers ua INNER JOIN answers a
ON ua.answer_id = a.answer_id  WHERE user_id = :user_id)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("user_id", $user_id, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getSpecificQuestion($question_id){
    include ("../../connection.php");


    $statement = "SELECT question FROM questions
                  WHERE question_id = :question_id";


    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("question_id", $question_id, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetch();
}
function getQuestionAnswers($question_id){
    include ("../../connection.php");


    $statement = "SELECT a.answer_id, a.answer FROM answers a
                  INNER JOIN questions q ON a.question_id = q.question_id
                  WHERE q.question_id = :question_id
                  AND a.dateDeleted IS NULL";
```

```php
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("question_id", $question_id, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getQuestionAnswersCount($question_id){
    include ("../../connection.php");


    $statement = "SELECT a.answer, COUNT(ua.useranswer_id) AS count
                  FROM answers a
                  LEFT JOIN useranswers ua ON a.answer_id = ua.answer_id
                  WHERE a.question_id = :question_id
                  GROUP BY a.answer_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("question_id", $question_id, PDO::PARAM_INT);


    $prepSt->execute();


    return $prepSt->fetchAll();
}
function getQuestionAnswerIds($question_id){
    include ("../../connection.php");


    $statement = "SELECT a.answer_id FROM answers a
                  INNER JOIN questions q ON a.question_id = q.question_id
                  WHERE q.question_id = :question_id
                  AND a.dateDeleted IS NOT NULL";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("question_id", $question_id, PDO::PARAM_INT);
```

```php
    $prepSt->execute();


    return $conn->fetchAll();
}
function saveQuestion($text){
    include ("../../connection.php");


    $statement = "INSERT INTO questions (question) VALUES (:text)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("text", $text);


    $prepSt->execute();


    return $conn->lastInsertId();
}
function saveQuestionAnswer($question_id, $text){
    include ("../../connection.php");


    $statement = "INSERT INTO answers (question_id, answer) VALUES
(:question_id, :text)";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("question_id", $question_id, PDO::PARAM_INT);
    $prepSt->bindParam("text", $text);

    $prepSt->execute();


    return $conn->lastInsertId();
}
function saveUserAnswer($user_id, $answer_id){
    include ("../../connection.php");


    $statement = "INSERT INTO useranswers (user_id, answer_id) VALUES
(:user_id, :answer_id)";
    $prepSt = $conn->prepare($statement);
```

```php
    $prepSt->bindParam("answer_id", $answer_id, PDO::PARAM_INT);
    $prepSt->bindParam("user_id", $user_id, PDO::PARAM_INT);


    $prepSt->execute();


    return $conn->lastInsertId();
}
function editQuestion($question_id, $text){
    include ("../../connection.php");


    $statement = "UPDATE questions SET question = :text WHERE question_id =
:question_id";


    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("question_id", $question_id, PDO::PARAM_INT);
    $prepSt->bindParam("text", $text);


    return $prepSt->execute();
}
function editQuestionAnswer($answer_id, $text){
    include ("../../connection.php");


    $statement = "UPDATE answers SET answer = :text WHERE answer_id =
:answer_id";


    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("answer_id", $answer_id, PDO::PARAM_INT);
    $prepSt->bindParam("text", $text);


    return $prepSt->execute();
}
```

```php
function disableQuestionAnswer($answer_id){
    include ("../../connection.php");


    $statement = "UPDATE answers SET dateDeleted = NOW() WHERE answer_id =
:answer_id";


    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("answer_id", $answer_id, PDO::PARAM_INT);


    return $prepSt->execute();
}
function disableQuestion($question_id){
    include ("../../connection.php");


    $statement = "UPDATE questions SET dateDeleted = NOW() WHERE question_id =
:question_id";


    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("question_id", $question_id, PDO::PARAM_INT);


    return $prepSt->execute();
}
function restoreQuestion($question_id){
    include ("../../connection.php");


    $statement = "UPDATE questions SET dateDeleted = NULL
                  WHERE question_id = :question_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("question_id", $question_id, PDO::PARAM_INT);


    return $prepSt->execute();
```

```
}
function checkIfUserAllowedToAnswer($user_id, $answer_id){
    include ("../../connection.php");


    $statement = "SELECT u.user_id FROM users u
                  INNER JOIN useranswers ua ON u.user_id = ua.user_id
                  INNER JOIN answers a ON ua.answer_id = a.answer_id
                  INNER JOIN questions q ON a.question_id = q.question_id
                  WHERE a.question_id IN (SELECT question_id FROM answers WHERE
answer_id = :answer_id)
                  AND ua.user_id = :user_id
                  AND a.dateDeleted IS NULL
                  AND q.dateDeleted IS NULL";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("user_id", $user_id, PDO::PARAM_INT);
    $prepSt->bindParam("answer_id", $answer_id, PDO::PARAM_INT);


    $prepSt->execute();


    $hasRows = $prepSt->fetch();


    if($hasRows){
        return false;
    }
    return true;
}
?>
```

userFunctions.php

```
<?php
function encryptPassword($password){
    return password_hash($password, PASSWORD_DEFAULT);
}
function createNewUser($email, $password, $name, $lastName){
    include ("../../connection.php");


    $crypted = encryptPassword($password);
```

```
    $statement = "INSERT INTO users (email, password, name, lastName)
                  VALUES (?, ?, ?, ?)";
    $prepSt = $conn->prepare($statement);

    $prepSt->bindParam(1, $email);
    $prepSt->bindParam(2, $crypted);
    $prepSt->bindParam(3, $name);
    $prepSt->bindParam(4, $lastName);


    $prepSt->execute();
}
function editUser($userId, $email, $name, $lastName, $role){
    include ("../../connection.php");


    $statement = "UPDATE users SET email = :email, name = :name, lastName =
:lastName, role_id = :role
                  WHERE user_id = :userId";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("email", $email);
    $prepSt->bindParam("name", $name);
    $prepSt->bindParam("lastName", $lastName);
    $prepSt->bindParam("role", $role, PDO::PARAM_INT);
    $prepSt->bindParam("userId", $userId, PDO::PARAM_INT);

    return $prepSt->execute();
}
function editUserPassword($userId, $password){
    include ("../../connection.php");


    $crypted = encryptPassword($password);


    $statement = "UPDATE users SET password = :password
                  WHERE user_id = :userId";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("userId", $userId, PDO::PARAM_INT);
    $prepSt->bindParam("password", $crypted);
```

```php
    return $prepSt->execute();
}
function checkIfEmailInUse($email){
    include ("../../connection.php");


    $statement = "SELECT email FROM users WHERE email = ?";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $email);


    $prepSt->execute();
    $result = $prepSt->rowCount() != 0;


    return $result;
}


function attemptLogin($email, $password){
    include ("../../connection.php");


    $statement = "SELECT password, user_id FROM users WHERE email = ?";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $email);


    $prepSt->execute();


    $user = $prepSt->fetch();

    if(!$user){
        return 0;
    }

    $userId = $user["user_id"];
```

```php
    $encryptedPassword = $user["password"];


    if(password_verify($password, $encryptedPassword)){
        return $userId;
    }
    else{
        return 0;
    }
}


function getUserInformation($id){
    include ("../../connection.php");


    $statement = "SELECT user_id, CONCAT(name, ' ', lastName) AS username,
level
    FROM users u
    INNER JOIN roles r ON u.role_id = r.role_id
    INNER JOIN accesslevels al ON r.access_level_id = al.access_level_id
    WHERE user_id = ?";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam(1, $id, PDO::PARAM_INT);


    $prepSt->execute();
    $result = $prepSt->fetch();


    return $result;
}


function getAllUsers(){
    include ("../../connection.php");


    $statement = "SELECT user_id AS id, name, lastName, email, dateCreated,
role_name
                  FROM users u
```

```
                    INNER JOIN roles r on u.role_id = r.role_id
                    ORDER BY r.role_id DESC";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();
    $result = $prepSt->fetchAll();


    return $result;
}


function getSpecificUser($userId){
    include ("../../connection.php");


    $statement = "SELECT name, lastName, email, role_id FROM users WHERE
user_id = :user_id";
    $prepSt = $conn->prepare($statement);


    $prepSt->bindParam("user_id", $userId);


    $prepSt->execute();


    return $prepSt->fetch();
}


function getAllUserRoles(){
    include ("../../connection.php");


    $statement = "SELECT role_id AS id, role_name as title FROM roles ORDER BY
id";
    $prepSt = $conn->prepare($statement);


    $prepSt->execute();


    return $prepSt->fetchAll();
```

```
}
?>
```

3.1.3 PHP Files outside of root

connection.php (not actual credentials)

```php
<?php
$servername = "localhost";
$username = "admin";
$pass = "adminskasifra";
$database = "nycestate";
$conn = new PDO("mysql:host=$servername;dbname=$database", $username, $pass);
$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
?>
```

3.2 CSS

3.2.1 style.css

```css
:root{
    --bg-color: #FFFFFF;
   --holder-color: #1F271B;
   --accent-color: #DCC7BE;
   --strong-accent: #CBB9A8;
   --light-action: #47a7e2;
   --action-color: #067BC2;
   --active-action: #00008F;
   --search-color :#FF3864;
   --search-active :#E00031;
}
a, #text-logo{
    color: var(--accent-color) !important;
}
h1,h2,p,a:hover,.h4:hover, #text-logo:hover, input, *{
    color: var(--strong-accent) !important;
}
.mk-icon{
    color: var(--strong-accent);
}
.navbar-toggler{
    background-color: var(--light-action);
    color: var(--bg-color);
}
.navbar-toggler:hover{
    background-color: var(--active-action);
}
input[type="checkbox"]:checked > span{
    background-color: var(--strong-accent);
}
.custom-check{
    position: relative;
    padding-left: 25px;
    display: inline !important;
}
.custom-check input{
    position: absolute;
    top: 0;
    left: 0;
    opacity: 100%;
    height: 20px;
    width: 20px;
```

```
}
.custom-check-target{
    position: absolute;
    top: 0;
    left: 0;
    height: 20px;
    width: 20px;
    border: 1px solid #eee;
    pointer-events: none;
}
.custom-check input:checked ~ .custom-check-target {
    background-color: var(--active-action);
}
.custom-check:hover  input ~ .custom-check-target{
    background-color: var(--action-color);
}
.custom-check-target:after {
    content: "";
    position: absolute;
    display: none;
}
.custom-check input:checked ~ .custom-check-target:after {
    display: block;
}
.mk-text-center{
    display: flex;
    text-align: center;
    align-items: center;
    justify-content: center;
}
.text-dark{
    color: var(--holder-color) !important;
}
body{
    overflow-x:hidden;
    background-color: var(--bg-color) !important;
  }
.navbar{
    min-height: 8vh;
}
.mk-page{
    position: relative;
    display: flex;
    align-items: center;
    justify-content: space-around;
```

```
}
.mk-part-page{
    min-height: 60vh;
}
.mk-long-page{
    min-height: 92vh;
}
.mk-solo-page{
    min-height: 77vh;
}
.mk-background{
    background-color: black;
}
.mk-flex-center{
    display: flex;
    align-items: center;
    text-align: center;
}
#landing-input-holder{
    position: relative;
    margin-bottom: 6vh;
}
#landing-input{
    width: 75%;
    position: absolute;
    left: 0px;
}
#landing-search{
    width: 25%;
    position: absolute;
    left: 75%;
    text-align: center;
}
#landing-search:hover{
    background-color: var(--active-action) !important;
    color: var(--bg-color) !important;
}
#landing-search-icon{
    position: absolute;
    left: 10px;
    z-index: 1;
    font-size: 1.25em;
    height: 5vh;
    display: flex;
    align-items: center;
```

```css
}
#landing-page{
    padding: 0px;
    background-image: url("../imgs/main.jpg");
    background-size: cover;
    background-position: center;
  }
#search-holder{
    width: 80%;
    margin: 0% auto;
    min-height: 8vh;
    display: flex;
    position: relative;
    gap: 10px;
    align-items: center;
}
#search-header{
    font-size: 1.5em;
}
input, select, option, textarea, .btn-light, td, th{
    color: var(--holder-color) !important;
}
.btn-light:hover{
    color: var(--holder-color) !important;
}
.btn-danger, .btn-success, .btn-info, .btn-primary{
    color: var(--bg-color) !important;
}
.btn-danger:hover, .btn-success:hover, .btn-info:hover, .btn-primary:hover,
.soft-blue{
    color: var(--bg-color) !important;
}
.deep-blue{
    background-color: var(--active-action) !important;
}
.deep-blue:hover{
    color: var(--bg-color) !important;
}
.soft-blue{
    background-color: var(--light-action) !important;
}
.soft-blue:hover{
    background-color: var(--active-action) !important;
    color: var(--bg-color) !important;
}
```

```css
.mk-button{
    background-color: var(--action-color) !important;
    text-decoration: none;
    color: var(--bg-color) !important;
    height: 5vh;
}
.mk-fix{
    --bs-gutter-x: 0;
}
.mk-introduction{
    width: 100vw;
    height: 15vh;
    background-color: var(--holder-color);
    text-align: center;
    align-items: center;
    justify-content: center;
    padding: 10px 0px;
}
.search-input{
    text-align: left;
    padding: 3px;
    padding-left: 30px !important;
    height: 5vh;
}
.mk-modal {
    position: fixed;
    z-index: 2;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    overflow: hidden;
    overflow-y: scroll;
    background-color: rgba(0,0,0,0.4);
}
.mk-modal-content {
    margin: 8% auto;
    color: #f7bd2f !important;
    background-color: #333333;
    padding: 20px;
    border: 1px solid #888;
    width: 90%;
    border-radius: 1%;
}
footer .row{
```

```
    height: 100%;
    margin: 0%;
    width: 100%;
}
footer{
    height: 15vh;
}
.error-message{
    color: red !important;
}
.alert-success{
    color: var(--holder-color) !important;
}
.hidden{
    display: none;
}
.active{
    color: var(--strong-accent) !important;
}
.success-outline{
    border: 2px solid green;
}
.error-outline{
    border: 2px solid red;
}
/*Admin panel classes*/
#admin-panel-holder{
    display: flex;
    flex-direction: column;
    gap: 20px;
}
#admin-tabs-holder{
    overflow-x: scroll;
}
#whole-admin-table{
    background-color: var(--strong-accent);
}
.admin-tab{
    color: var(--bg-color) !important;
}
#error-holder{
    position: fixed;
    bottom: 40px;
    right: 20px;
    padding: 20px;
```

```
    z-index: 5;
    pointer-events: none;
}
.question-form{
    border: 1px solid black;
    padding: 5%;
    border-radius: 1%;
}
.error-message{
    opacity: 1;
    transition: opacity 200ms ease 0ms;
}
#buildingTypeFilters, #boroughFilters{
    display: inline-flex;
    flex-direction: column;
    flex-basis: auto;
}
.listing{
    position: relative;
    width: 100% !important;
    margin-bottom: 10px;
    padding-bottom: 2%;
    color: black !important;
    /* background-image: radial-gradient( circle 311px at 8.6% 27.9%,
rgba(62,147,252,0.57) 12.9%, rgba(239,183,192,0.44) 91.2% ); */
    background-color: var(--holder-color);
    border: 1px solid black;
    border-radius: 2%;
}
.listing:hover > .listing-cover{
    opacity: 95%;
}
.listing-main{
    position: relative;
    width: 100% !important;
    height: 100%;
    z-index: 1;
}
.listing-cover{
    top: 0px;
    left: 0px;
    position: absolute;
    border-radius: 2%;
    z-index: 2;
    opacity: 0%;
```

```
    width: 100%;
    height: 100%;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    background-color: #FFFFFF;
    /* pointer-events: none; */
    /* background-image: radial-gradient( circle 311px at 8.6% 27.9%,
rgba(62,147,252,0.57) 12.9%, rgba(239,183,192,0.44) 91.2% ); */
    color: black;
    transition: opacity 100ms ease 20ms;
}
.list-group{
    border-radius: 2%;
}
.listing-footer{
    position: absolute;
    box-sizing: content-box;
    bottom: 5px;
    z-index: 3;
}
.listing-padding{
    padding: 2%;
}
.mk-favorite-icon-holder{
    position: absolute;
    top: 20px;
    right: 10px;
    background-color: white;
    padding: 4px;
    color: red !important;
    border-radius : 50%;
    z-index: 5;
  }
  .mk-favorite-icon-holder:hover{
    color: red !important;
  }
  .mk-favorite-icon-holder:hover{
    scale: 1.10;
  }
  .mk-favorite-icon{
    font-size: 40px;
}
.listing-img{
```

```
    position: relative;
}
.listing-body{
    padding: 2%;
}
.listing-desc{
    min-height: 2rem;
    margin-bottom: 0%;
}
.mk-img-fluid{
    width: 100%;
    aspect-ratio: 20/9;
}
#singleListingHolder{
    border: 1px solid black;
    background-color: var(--holder-color);
}
.mk-main-img{
    max-width: 100%;
    width: 100%;
    aspect-ratio: 16/9 !important;
    border: 2px solid black;
    border-radius: 2%;
}
.portrait{
    aspect-ratio: 9/16 !important;
}


.hide{
    opacity: 0 !important;
    transition: opacity 1800ms ease 0ms;
}
#searchForm{
    width: 70%;
    align-items: center;
}
.text-center-on-mobile{
    text-align: center;
}
@media screen and (min-width: 992px) {
    .navbar{
        height: 8vh;
        padding: 0% 2% 0% 2%;
    }
```

```
    footer .row{
      padding: 0% 2% 0% 2% !important;
    }
    .listing{
        width: 45% !important;
    }
    .mk-introduction{
        width: 30vw;
        min-width: 30vw;
        min-height: 20vh;
        border-radius: 5%;
    }
    #searchForm{
        width: 100%;
        align-items: baseline;
    }
    .mk-favorite-icon{
        font-size: 50px;
    }
    .text-center-on-mobile{
        text-align: left;
    }
}
```

3.3 HTML

3.3.1 index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="resources/imgs/favicon.ico">
    <script src="resources/js/main.js"></script>
    <title>NYCEstate</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
```

```
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            <!--
              <li class="nav-item">
                <a class="nav-link active" aria-current="page"
href="#">Home</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="#">Link</a>
              </li>
              <li class="nav-item dropdown">
                <a class="nav-link dropdown-toggle" href="#" role="button"
data-bs-toggle="dropdown" aria-expanded="false">
                  Dropdown
                </a>
                <ul class="dropdown-menu">
                  <li><a class="dropdown-item" href="#">Action</a></li>
                  <li><a class="dropdown-item" href="#">Another action</a></li>
                  <li><hr class="dropdown-divider"></li>
                  <li><a class="dropdown-item" href="#">Something else
here</a></li>
                </ul>
              </li>
              <li class="nav-item">
                <a class="nav-link disabled">Disabled</a>
              </li>

            -->
            </ul>
          </div>
        </div>
      </nav>
      <!--Content-->
      <div class="container-fluid mk-page mk-solo-page" id="landing-page">
        <div class="row d-flex w-100">
        <form  class="mk-introduction d-flex flex-column justify-content-center
mx-auto" id="landing-form" action="pages/listings.html" method="get">
          <h2 id="search-header">Find your next NYC property</h2>
```

```
          <div id="search-holder">
            <div id="landing-input-holder" class="w-100">
            </div>
              <div id="landing-search-icon">
                <i class="mk-icon fa-solid fa-magnifying-glass"></i>
              </div>
              <select name="messageType" class="search-input"  id="landing-
input" required>
                  <option value="0">All boroughs</option>
              </select>
              <span class="error-msg hidden"></span>
              <a href="" id="landing-search" class="mk-button mk-text-
center">Search</a>
          </div>
        </form>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
      </div>
      <!--Footer-->
      <footer class="container-fluid mk-background">
        <div class="row w-100" >
          <div class="mk-logo col-6 mk-flex-center">
            <a id="footerHeaderHolder" href="#">Tekst</a>
          </div>
          <div class="col-6 flex  flex-md-row">
            <ul id="footerHolder" class="row align-items-center list-unstyled
p-0">

            </ul>
          </div>
        </div>
      </footer>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
  <script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.3.2 admin.html

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <script src="../resources/js/main.js"></script>
    <title>Admin panel</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
      <div class="container-fluid mk-page mk-solo-page">
        <!--modal background-->
        <div id="modal-background" class="mk-modal hidden">
          <!--One of modal contents-->
          <div id="user-modal" class="mk-modal-content hidden">
            <h5 id="user-modal-title" class="text-center">Edit existing
user</h5>
            <form class="row mk-flex-center">
              <div class="col-12">
                <label for="userName">First name</label>
                <input type="text" class="form-control"  name="userName"
id="userName" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="userLastName">Last name</label>
```

```
                <input type="text" class="form-control"  name="userLastName"
id="userLastName" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="userEmail">Email</label>
                <input type="email" class="form-control"  name="userEmail"
id="userEmail" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="userRole">Role</label>
                <select name="" class="form-control" id="userRole" required>
                  <option value="0">Select a role</option>
                </select>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="userLastName">Password (unchanged if empty)</label>
                <input type="password" class="form-control"
name="userPassword" id="userPassword"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12 my-2 p-2">
                <button id="user-submit" class="btn btn-success col-6 my-1"
type="submit">Edit user</button>
                <button id="user-cancel" class="btn btn-danger col-6 my-1
close-button" type="button">Cancel</button>
              </div>
              <input type="hidden" id="userId" name="userId" value="0" />
            </form>
          </div>
          <!--One field modal-->
          <div id="borough-modal" class="mk-modal-content hidden">
            <h5 id="borough-modal-title" class="text-center">Create new
borough</h5>
            <form class="row mk-flex-center">
              <div class="col-12">
                <label for="boroughName">Borough name</label>
                <input type="text" class="form-control"  name="boroughName"
id="boroughName" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12 my-2 p-2">
```

```
                <button id="borough-submit" class="btn btn-success col-6 my-1"
type="submit">Create new borough</button>
                <button id="borough-cancel" class="btn btn-danger col-6 my-1
close-button" type="button">Cancel</button>
              </div>
              <input type="hidden" id="boroughId" name="boroughId" value="0" />
            </form>
          </div>
          <!--End of one field modal-->
          <!--One field modal-->
          <div id="buildingType-modal" class="mk-modal-content hidden">
            <h5 id="buildingType-modal-title" class="text-center">Create new
building type</h5>
            <form class="row mk-flex-center">
              <div class="col-12">
                <label for="buildingTypeName">Building type name</label>
                <input type="text" class="form-control"
name="buildingTypeName" id="buildingTypeName" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12 my-2 p-2">
                <button id="buildingType-submit" class="btn btn-success col-6
my-1" type="submit">Create new borough</button>
                <button id="buildingType-cancel" class="btn btn-danger col-6
my-1 close-button" type="button">Cancel</button>
              </div>
              <input type="hidden" id="buildingTypeId" name="buildingTypeId"
value="0" />
            </form>
          </div>
          <!--End of one field modal-->
          <!--One field modal-->
          <div id="roomType-modal" class="mk-modal-content hidden">
            <h5 id="roomType-modal-title" class="text-center">Create new room
type</h5>
            <form class="row mk-flex-center">
              <div class="col-12">
                <label for="roomTypeName">Room type name</label>
                <input type="text" class="form-control"  name="roomTypeName"
id="roomTypeName" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12 my-2 p-2">
                <button id="roomType-submit" class="btn btn-success col-6 my-1"
type="submit">Create new roomType</button>
```

```
                <button id="roomType-cancel" class="btn btn-danger col-6 my-1
close-button" type="button">Cancel</button>
              </div>
              <input type="hidden" id="roomTypeId" name="roomTypeId"
value="0"/>
            </form>
          </div>
          <!--End of one field modal-->
          <!--One field modal-->
          <div id="messageType-modal" class="mk-modal-content hidden">
            <h5 id="messageType-modal-title" class="text-center">Create new
message</h5>
            <form class="row mk-flex-center">
              <div class="col-12">
                <label for="messageTypeName">Message type name</label>
                <input type="text" class="form-control"  name="messageTypeName"
id="messageTypeName" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12 my-2 p-2">
                <button id="messageType-submit" class="btn btn-success col-6
my-1" type="submit">Create new messageType</button>
                <button id="messageType-cancel" class="btn btn-danger col-6 my-
1 close-button" type="button">Cancel</button>
              </div>
              <input type="hidden" id="messageTypeId" name="messageTypeId"
value="0"/>
            </form>
          </div>
          <!--End of one field modal-->
          <!--Survey question modal-->
          <div id="question-modal" class="mk-modal-content hidden">
            <h5 id="question-modal-title" class="text-center">Create new survey
question</h5>
            <form class="row mk-flex-center">
              <div class="col-12">
                <label for="questionName">Question</label>
                <input type="text" class="form-control"  name="questionName"
id="questionName" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div>
                <button id="questionAddAnswer" class="btn btn-success col-6 my-
1" type="button">Add answer</button>
                <span class="error-msg hidden d-block"></span>
```

```
              </div>
              <div id="question-answer-holder">


              </div>
              <div class="col-12 my-2 p-2">
                <button id="question-submit" class="btn btn-success col-6 my-1"
type="submit">Create new question</button>
                <button id="question-cancel" class="btn btn-danger col-6 my-1
close-button" type="button">Cancel</button>
              </div>
              <input type="hidden" id="questionId" name="questionId"
value="0"/>
            </form>
          </div>
          <!--End of survey question modal-->
          <!--Question results modal-->
          <div id="question-results-modal" class="mk-modal-content hidden">
            <form class="row mk-flex-center">
              <div class="col-12">
                <p id="question-title" class="h4">Question: Do you like this
website?</p>
              </div>
              <div class="col-12 table-responsive">
                <table class="table table-primary table-responsive table-
hover">
                  <thead >
                    <tr>
                      <th>Answer</th>
                      <th>Number of times selected</th>
                    </tr>
                  </thead>
                  <tbody id="question-results">


                  </tbody>
                </table>
              </div>
              <div class="col-12 my-2 p-2">
                <button id="question-cancel" class="btn btn-danger col-6 my-1
close-button" type="button">Close</button>
              </div>
            </form>
          </div>
          <!--End of question results modal-->
```

```
          <div id="link-modal" class="mk-modal-content hidden">
            <h5 id="link-modal-title" class="text-center">Create a new
link</h5>
            <form class="row mk-flex-center">
              <div class="col-12">
                <label for="LinkTitle">Link title</label>
                <input type="text" class="form-control"  name="LinkTitle"
id="LinkTitle" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="LinkHref">Link href</label>
                <input type="text" class="form-control"  name="LinkHref"
id="LinkHref" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="LinkIcon">Iconify icon (If any)</label>
                <input type="text" class="form-control" name="LinkIcon"
id="LinkIcon"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="LinkReqLevel">Required access level</label>
                <select name="" class="form-control" id="LinkReqLevel"
required>
                  <option value="0">Select an access level</option>
                </select>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="LinkLocation">Location</label>
                <br/>
                <select name="" class="form-control"  id="LinkLocation"
required>
                  <option value="0">Select link location</option>
                  <!-- <option value="navbar">Navbar</option>
                  <option value="footer">Footer</option>
                  <option value="head">Head</option> -->
                </select>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="linkPriority">Link priority (higher goes
first)</label>
```

```
                <input type="number" class="form-control"  name="linkPriority"
id="linkPriority" value="1" min="1" max="99" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="LinkRoot">In root?</label>
                <input type="checkbox" name="LinkRoot" id="LinkRoot"/>
              </div>
              <div class="col-12 my-2 p-2">
              <button id="link-submit" class="btn btn-success my-1 col-6"
type="submit">Create new link</button>
              <button id="link-cancel" class="btn btn-danger my-1 col-6 close-
button" type="button">Cancel</button>
              </div>
              <input type="hidden" id="linkId" name="linkId" value="0" />
            </form>
          </div>
          <div id="listing-modal" class="mk-modal-content modal-dialog-
scrollable hidden">
            <h5 id="listing-modal-title" class="text-center">Create a new
listing</h5>
            <form method="POST" enctype="multipart/form-data" class="row mk-
flex-center">
              <div class="col-12">
                <label for="listingTitle">Listing title</label>
                <input type="text" class="form-control"  name="listingTitle"
id="listingTitle" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="listingDescription">Listing description</label>
                <textarea class="form-control"  name="listingDescription"
id="listingDescription" required="required"></textarea>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="listingAddress">Listing address</label>
                <input type="text" class="form-control"  name="listingAddress"
id="listingAddress" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="listingSize">Listing size (in feet)</label>
                <input type="number" class="form-control"  name="listingSize"
id="listingSize" min="30" max="100000" required="required"/>
```

```
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="listingPrice">Listing price (in dollars)</label>
                <input type="number" class="form-control"  name="listingPrice"
id="listingPrice" min="1000" max="1000000000" required="required"/>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="listingBorough">Borough</label>
                <select name="listingBorough" class="form-control"
id="listingBorough" required>
                  <option value="0">Select a borough</option>
                </select>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="listingBuildingType">Building type</label>
                <select name="listingBuildingType" class="form-control"
id="listingBuildingType" required>
                  <option value="0">Select a building type</option>
                </select>
                <span class="error-msg hidden"></span>
              </div>
              <div class="col-12">
                <label for="listingRoomsList" class="d-block">Rooms</label>
                <select name="listingRoomsList" class="form-control w-50 d-
inline" id="listingRoomsList">
                  <option value="0">Select a room</option>
                </select>
                <span class="error-msg hidden"></span>
                <button id="addListingRoomButton" class="d-block btn btn-
success w-25 my-2 mx-auto" type="button">+</button>
                <div id="room-holder">
                  <!-- <label for="1" class="d-block">Livingroom</label>
                  <input type="number" min="1" class="form-control d-inline
listingRoom w-50" name="" id="1">
                  <label for="1" class="d-block">Livingroom</label>
                  <input type="number" min="1" class="form-control d-inline
listingRoom w-50" name="" id="1"> -->
                </div>
              </div>
              <div class="col-12 flex">
                <label for="listingPhoto">Choose listing image</label>
                </br>
```

```
                <input type="file" id="listingPhoto" name="listingPhoto"
required="required" accept="image/png, image/jpeg"/>
                <span class="error-msg hidden"></span>
                </br>
                <img src="..." id="main-photo-preview" class="img-fluid w-25"
alt="image preview">
              </div>
              <div class="col-12 my-2 p-2">
                <button id="listing-submit" class="btn btn-success my-1 col-6"
type="submit">Create new listing</button>
                <button id="listing-cancel" class="btn btn-danger my-1 col-6
close-button" type="button">Cancel</button>
                </div>
              <input type="hidden" id="listingId" name="listingId" value="0"/>
            </form>
          </div>
        </div>
        <div class="row d-flex justify-content-center w-100 position-relative
top-0">
            <div class="col-10" id="admin-panel-holder">
                <div class="col-12" id="whole-admin-table">
                  <h2 class="bg-dark my-0 p-2">Admin panel</h2>
                  <div class="d-flex gap-0 bg-dark" id="admin-tabs-holder">
                    <a href="#" class="btn btn-primary">Listings</a>
                    <a href="#" class="btn btn-info">Users</a>
                    <a href="#" class="btn btn-info">Links</a>
                    <a href="#" class="btn btn-info">Boroughs</a>
                    <a href="#" class="btn btn-info">Room types</a>
                  </div>
                  <div class="table-responsive">
                    <table class="table table-responsive table-hover"
id="element-table">
                      <thead>
                        <tr id="header-table-row">
                        <!--
                        <th>
                          Name
                        </th>
                        <th>
                          Price
                        </th>
                        <th>
                          Description
                        </th>
                        <th>
```

```
                        Address
                      </th>
                      <th>
                        Size
                      </th>
                      <th>
                        Options
                      </th> -->
                    </tr>
                    </thead>
                    <tbody id="table-result-holder">

                    </tbody>
                  </table>
                </div>
              </div>
            </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">


          </ul>
        </div>
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.3.3 author.html

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <script src="../resources/js/main.js"></script>
    <title>About author</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
    <div class="container-fluid mk-page mk-solo-page">
        <div class="row d-flex w-100 justify-content-center">
          <!--Start of author page-->
          <div class="col-md-8 col-8 p-2 row mk-fix" id="singleListingHolder">
            <!-- Start of author img-->
            <div class="col-12 col-md-2">
              <div class="listing-img w-100">
                <img src="../resources/imgs/author.jpg" alt="" class="img-fluid
mk-main-img portrait"/>
              </div>
            </div>
            <!--Start of author description-->
            <div class="col-12 col-md-10" id="listingHolder">
              <div class="listing-body">
                <h3 class="">Ilija Krstić 155/21</h3>
```

```
                <p class="">Sole developer of this and many other websites and
applications, student at the ICT College of Vocational Studies in Belgrade</p>
                <p>Professional status: Student</p>
                <p>More than three years of experience with web
technologies</p>
              </div>
              <div class="d-flex justify-content-between">
                <div class="listing-rooms listing-padding">
                  <h3>Skills</h3>
                  <p>Backend development</p>
                  <p>Database design</p>
                  <p></p>
                </div>
                <div class="contact listing-padding d-flex flex-column">
                  <a href="https://github.com/techbabette" class="btn soft-
blue">View Github</a>
                  <a href="https://rs.linkedin.com/in/ilija-krsti%C4%87-
b13428134" class="btn soft-blue my-2">View Linkedin</a>
                </div>
              </div>
            </div>
            <!--End of listing-->
          </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">
          </ul>
        </div>
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
```

```
</html>
```

3.3.4 contact.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <script src="../resources/js/main.js"></script>
    <title>Contact</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
    <div class="container-fluid mk-page mk-solo-page">
        <div class="row d-flex w-100 justify-content-center">
            <div class="col-6 bg-dark p-4">
                <form id="contactForm">
                    <div class="mb-3">
                        <label for="messageTitle" class="form-label">Message
title</label>
                        <input type="text" class="form-control"
name="messageTitle" id="messageTitle">
                        <span class="error-msg hidden"></span>
                    </div>
```

```
                    <div class="mb-3">
                        <label for="messageType">Message type</label>
                        <select name="messageType" class="form-control"
id="messageType" required>
                            <option value="0">Select a message type</option>
                        </select>
                        <span class="error-msg hidden"></span>
                    </div>
                    <div class="mb-3">
                        <label for="messageBody" class="form-label">Message
body</label>
                        <textarea class="form-control"  name="messageBody"
id="messageBody" required="required"></textarea>
                        <span class="error-msg hidden"></span>
                    </div>
                    <button type="submit" id="sendMessage" name="sendMessage"
class="btn btn-primary">Submit</button>
                </form>
            </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">


          </ul>
        </div>
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.3.5 favorites.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <script src="../resources/js/main.js"></script>
    <title>Favorites</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
    <div class="container-fluid mk-page mk-solo-page">
            <div class="col-7 p-2 row mk-fix justify-content-between text-
center">
                <h3 class="my-5">Your favorite listings</h3>
                <div class="col-12 row mk-fix justify-content-between text-
center" id="listingHolder">


                </div>
            </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
```

```
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">



          </ul>
        </div>
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.3.6 listing.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <script src="../resources/js/main.js"></script>
    <title>Listing</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
```

```
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
    <div class="container-fluid mk-page mk-solo-page">
        <div class="row d-flex w-100 justify-content-center">
          <!--Start of listing-->
          <div class="col-md-8 col-11 p-2 row mk-fix" id="singleListingHolder">


            <!--End of listing-->
          </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">
          </ul>
        </div>
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.3.7 login.html

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <script src="../resources/js/main.js"></script>
    <title>Login</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
      <div class="container-fluid mk-page mk-solo-page">
        <div class="row d-flex justify-content-center w-100">
            <div class="col-6 bg-dark p-4">
                <form id="loginForm">
                    <div class="mb-3">
                      <label for="exampleInputEmail1" class="form-label">Email
address</label>
                      <input type="email" class="form-control" id="emailInput"
aria-describedby="emailHelp">
                    </div>
                    <div class="mb-3">
                      <label for="exampleInputPassword1" class="form-
label">Password</label>
                      <input type="password" class="form-control"
id="loginPasswordInput">
                    </div>
```

```
                    <button type="submit" id="createNewUser"
name="createNewUser" class="btn btn-primary">Submit</button>
                </form>
            </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">


          </ul>
        </div>
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.3.8 register.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <script src="../resources/js/main.js"></script>
    <title>Register</title>
```

```
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
    <div class="container-fluid mk-page mk-solo-page">
        <div class="row d-flex w-100 justify-content-center">
            <div class="col-6 bg-dark p-4">
                <form id="registrationForm">
                    <div class="mb-3">
                      <label for="exampleInputEmail1" class="form-label">Email
address</label>
                      <input type="email" class="form-control" id="emailInput"
aria-describedby="emailHelp">
                      <span class="error-msg hidden"></span>
                      <div id="emailHelp" class="form-text">We'll never share
your email with anyone.</div>
                    </div>
                    <div class="mb-3">
                        <label for="nameInput" class="form-label">Name</label>
                        <input type="text" class="form-control"
name="firstName" id="nameInput">
                        <span class="error-msg hidden"></span>
                    </div>
                    <div class="mb-3">
                        <label for="lastNameInput" class="form-label">Last
name</label>
                        <input type="text" class="form-control" name="lastName"
id="lastNameInput">
                        <span class="error-msg hidden"></span>
                    </div>
                    <div class="mb-3">
```

```
                    <label for="exampleInputPassword1" class="form-
label">Password</label>
                    <input type="password" class="form-control"
id="registerPasswordInput">
                    <span class="error-msg hidden"></span>
                  </div>
                  <button type="submit" id="createNewUser"
name="createNewUser" class="btn btn-primary">Submit</button>
                </form>
            </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">


          </ul>
        </div>
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.3.9 survey.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
```

```
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <script src="../resources/js/main.js"></script>
    <title>Survey</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
    <div class="container-fluid mk-page mk-solo-page">
        <div class="row d-flex w-100 justify-content-center">
            <div class="col-md-6 col-10" id="surveyHolder">


            </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">


          </ul>
        </div>
```

```
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.3.10 listings.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
crossorigin="anonymous">
    <link rel="stylesheet" href="../resources/css/style.css">
    <script src="../resources/js/main.js"></script>
    <link rel="icon" type="image/x-icon" href="../resources/imgs/favicon.ico">
    <title>Listings</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg mk-background">
        <div class="container-fluid">
          <a class="navbar-brand" id="headerHolder" href="#">Navbar</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul id="navbarHolder" class="navbar-nav ms-auto mb-2 mb-lg-0">
            </ul>
          </div>
        </div>
      </nav>
    <div class="container-fluid mk-page mk-solo-page">
        <div class="row w-100 justify-content-center">
            <div class="col-md-3 col-12 p-2 mx-auto">
```

```
            <!--class="d-flex flex-column align-items-center text-center" for
form on mobile-->
            <form action="" class="text-dark d-flex flex-column mx-auto"
id="searchForm" method="post">
            <div>
              <label for="titleFilter" class="h3 text-dark text-center-on-
mobile w-100">Title filter</label>
              <input type="text" class="boroughFilter d-block"
name="titleFilter" id="titleFilter" value=""/>
            </div>
            <!--Start of borough filters-->
              <h3 class="text-dark">Borough filter</h3>
              </br>
              <div id="boroughFilters">


              </div>
            <!--End of borough filters-->
            <!--Start of building type filters-->
              <h3 class="text-dark">Building type filter</h3>
              </br>
              <div id="buildingTypeFilters">


              </div>
            <!--End of building type filters-->
            <!--Start of sort option-->
            <div>
              <label class="text-dark h3 text-center-on-mobile d-block"
for="listingsSort">Sort</label>
              <select name="" class="form-control" id="listingsSort">
                <option value="0">Newest first</option>
                <option value="1">By price (Ascending)</option>
                <option value="2">By price (Descending)</option>
                <option value="3">By size (Ascending)</option>
                <option value="4">By size (Descending)</option>
              </select>
            </div>
            <!--End of sort option-->
            <div>
              <button type="submit" id="searchListings" class="btn soft-blue
my-2">Search</button>
              <!-- <button type="reset" class="btn btn-danger">Reset</button>
-->
            </div>
```

```
            </form>
              <!--This is where the filters will go-->
            </div>
            <div class="col-md-7 col-12 p-2 row mk-fix justify-content-between
text-center" id="listingHolder">
            </div>
        </div>
        <div id="error-holder" class="d-flex flex-column">
        </div>
    </div>
    <footer class="container-fluid mk-background">
      <div class="row w-100" >
        <div class="mk-logo col-6 mk-flex-center">
          <a id="footerHeaderHolder" href="#">Tekst</a>
        </div>
        <div class="col-6 flex  flex-md-row">
          <ul id="footerHolder" class="row align-items-center list-unstyled p-
0">


          </ul>
        </div>
      </div>
    </footer>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-
w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="https://code.iconify.design/2/2.2.1/iconify.min.js"></script>
</body>
</html>
```

3.4 Javascript

3..4.1 main.js

```
let url = window.location.href.split("/");
let lastOfUrl = url[url.length - 1];
let currentPage = lastOfUrl.split("?")[0].toLowerCase().replace(/[^A-Za-z0-
9\.]/gi, "");
let mainPage = false;
let ajaxPath = "BusinessLogic/";
let data;
let globalData = {};
let success;
if (currentPage == "" || currentPage == "index.html") mainPage = true;
```

```
if (!mainPage) ajaxPath = "../BusinessLogic/";
window.onload = function(){
    if(currentPage == "") {
        currentPage == "index.html";
    }
    data = {currentPage};


    //Send current page to check if allwwed
    submitAjax("getLinks", generateNavbar, data, { newLocation : "index.html",
landing : true, redirectOnNotAllowed : true});


    if(currentPage == "register.html"){
        let registrationForm = document.querySelector("#registrationForm");
        registrationForm.addEventListener("submit", function(e){
            e.preventDefault();


            let emailField = document.querySelector("#emailInput");
            let nameField = document.querySelector("#nameInput");
            let lastNameField = document.querySelector("#lastNameInput");
            let passwordField =
document.querySelector("#registerPasswordInput");


            let email = emailField.value;
            let name = nameField.value;
            let lastName = lastNameField.value;
            let password = passwordField.value;


            //Check if data is valid
            let errors = 0;


            let reName = /^[A-Z][a-z]{1,14}(\s[A-Z][a-z]{1,14}){0,2}$/;


            let rePass1 = /[A-Z]/;
            let rePass2 = /[a-z]/;
            let rePass3 = /[0-9]/;
            let rePass4 = /[!\?\.]/;
            let rePass5 = /^[A-Za-z0-9!\?\.]{7,30}$/;
```

```
            let reEmail = /^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;


            let singleTests = [
                {re : reName, field: nameField, message : `Name doesn't match
format, eg "Nathan"`},
                {re : reName, field: lastNameField, message : `Last name does
not match format, eg "Smith"`},
                {re : reEmail, field: emailField, message : `Email not valid`},
            ]


            let passwordTests = [
            {re : rePass5, field: passwordField, message : `Password must be
between 7 and 30 characters long`},
            {re : rePass1, field: passwordField, message : `Password must
contain an uppercase letter`},
            {re : rePass2, field: passwordField, message : `Password must
contain a lowercase letter`},
            {re : rePass3, field: passwordField, message : `Password must
contain a digit`},
            {re : rePass4, field: passwordField, message : `Password must
contain ! or ? or .`}
            ]


            for(let test of singleTests){
                errors += reTestText(test.re, test.field, test.message);
            }


            for(let test of passwordTests){
                let result = reTestText(test.re, test.field, test.message);
                errors += result;


                if(result > 0) break;
            }


            let data = {"createNewUser" : true, "email" : email, "name" : name,
"lastName" : lastName, "password" : password};
```

```
            if(errors == 0){
                submitAjax("createNewUser", redirectSuccess, data, {newLocation
: "login.html", "landing" : false});
            }
        })
    }
    if(currentPage == "login.html"){
        let loginForm = document.querySelector("#loginForm");
        loginForm.addEventListener("submit", function(e){
            e.preventDefault();


            let emailField = document.querySelector("#emailInput");
            let passwordField = document.querySelector("#loginPasswordInput");


            let email = emailField.value;
            let password = passwordField.value;


            //Check if data is valid
            let errors = 0;


            let rePass1 = /[A-Z]/;
            let rePass2 = /[a-z]/;
            let rePass3 = /[0-9]/;
            let rePass4 = /[!\?\.]/;
            let rePass5 = /^[A-Za-z0-9!\?\.]{7,30}$/;


            let reEmail = /^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;


            errors += reTestText(reEmail, emailField, "");


            errors += reTestText(rePass1, passwordField, "");
            errors += reTestText(rePass2, passwordField, "");
            errors += reTestText(rePass3, passwordField, "");
            errors += reTestText(rePass4, passwordField, "");
            errors += reTestText(rePass5, passwordField, "");
```

```
            let data = {"attemptLogin" : true, "email" : email, "pass" :
password};


            if(errors == 0){
                submitAjax("attemptLogin", redirectSuccess, data, { newLocation
: "index.html", landing : true});
            }
            else{
                errorHandler("Incorrect email/password");
            }
        })
    }
    if(currentPage === "admin.html"){
        let currData;
        globalData.modalBackground = document.querySelector(".mk-modal");
        window.addEventListener("click", function(e){
            if(e.target === globalData.modalBackground){
                closeCurrentModal();
            }
        })
        let tables = [
                        {title : "Users", headers : ["Name", "Last name","Email",
"Date of creation", "Role"], target : "getAllUsers", edit : showUserModal},
                        {title : "Messages", headers : ["Sender", "Type","Title",
"Body", "Date sent"], target : "getAllMessages"},
                        {title : "Message types", headers : ["Title", "Number of
messages"], target : "getAllMessageTypesCount", edit : showMessageTypeModal,
createNew : showMessageTypeModal},
                        {title : "Listings", headers : ["Name",
"Price","Description", "Borough", "Building type", "Address", "Size"], target :
"getAllListings", createNew : showListingModal, edit: showListingModal},
                        {title : "Links", headers : ["Title", "Access
level","Link", "File location", "Location",  "Priority", "Parent", "Icon"],
target : "getAllLinks", createNew : showLinkModal, edit : showLinkModal},
                        {title : "Boroughs", headers : ["Title", "Number of
listings (Both active and deleted)"], target : "getAllBoroughsCount",
createNew: showBoroughModal, edit: showBoroughModal},
                        {title : "Building types", headers : ["Title", "Number of
listings (Both active and deleted)"], target : "getAllBuildingTypesCount",
createNew: showBuildingTypeModal, edit: showBuildingTypeModal},
```

```
                    {title : "Survey questions", headers : ["Title", "Number
of times answered"], target : "getAllQuestions", createNew : showQuestionModal,
edit: showQuestionModal, viewAnswers: "getSpecificQuestionAnswers"},
                    {title : "Deleted survey questions", headers : ["Title",
"Number of times answered"], target : "getAllDeletedQuestions", edit:
showQuestionModal, restore : "restoreQuestion", viewAnswers:
"getSpecificQuestionAnswers"},
                    {title : "Room types", headers : ["Title"], target :
"getAllRoomTypes", createNew: showRoomTypeModal, edit: showRoomTypeModal},
                    {title : "Deleted listings", headers : ["Name",
"Price","Description", "Borough", "Building type", "Address", "Size"], target :
"getAllDeletedListings", edit: showListingModal, restore : "restoreListing"},
                ];
        let table = document.querySelector("#element-table");
        let activeTable = 0;
        let html = "";
        let tabHolder = document.querySelector("#admin-tabs-holder");
        let active;
        globalData.rooms = new Array();
        //For each table, generate a button
        for(let table in tables){
            active = false;
            let id = table;
            let currTable = tables[id];
            if(table == activeTable) active = true;
            html += `<a href="#" data-id="${table}" class="btn ${active ?
"deep-blue" : "soft-blue"} admin-tab">${currTable["title"]}</a>`;
        }
        tabHolder.innerHTML = html;
        //To every button, add an event listener
        let adminTabs = document.querySelectorAll(".admin-tab");
        for(let tab of adminTabs){
            tab.addEventListener("click", function(e){
                e.preventDefault();
                activeTable = this.dataset.id;
                generateTable(this.dataset.id);
                applyCurrentTab(this.dataset.id);
            })
        }
        generateTable();
        //Set up all modals
        setUpModals();
        //Generates the structure of a table
        function generateTable(){
            let tableId = activeTable;
```

```
            generateHeaderTableRow(table, tableId)
            let target = tables[tableId].target;
            //Makas an AJAX request and fills table with resulting information
            readAjax(target, fillTable);
        }
        function applyCurrentTab(tableId){
            activeTable = tableId;
            for(let tab of adminTabs){
                if(tab.dataset.id != tableId){
                    tab.classList.remove("deep-blue");
                    tab.classList.add("soft-blue");
                }
                else{
                    tab.classList.add("deep-blue");
                    tab.classList.remove("soft-blue");
                }
            }
        }
        function fillTable(data){
            let html = "";
            let counter = 1;
            let tableResultHolder = document.querySelector("#table-result-
holder");
            if(data.length < 1){
                html += `<p class="text-center w-100 d-block text-dark">No rows
to display</p>`
                tableResultHolder.innerHTML = "";
                tableResultHolder.innerHTML += html;
            }
            currData = data;
            for(let row of data){
                html +=
                `
                <tr>
                <td>
                ${counter++}
                </td>
                `
                for(let i = 1; i < Object.keys(row).length / 2; i++){
                    html +=
                    `
                    <td>
                    ${row[i]}
                    </td>
                    `
```

```
                }
                html += `<td>`
                if(tables[activeTable].edit){
                    html += `<a href="#" data-id="${row["id"]}" class="btn btn-
light edit-button">Edit</button>`
                }
                if(tables[activeTable].viewAnswers){
                    html += `<a href="#" data-
table="${tables[activeTable].title}" data-id="${row["id"]}" class="btn btn-info
view-answers-button">View answers</a>`
                }
                if(tables[activeTable].restore){
                    html += `<a href="#"  data-
table="${tables[activeTable].title}" data-id="${row["id"]}" class="btn btn-
success restore-button">Restore</a>`
                }
                else{
                    html += `<a href="#" data-
table="${tables[activeTable].title}" data-id="${row["id"]}" class="btn btn-
danger delete-button">Delete</a>`
                }
                html += `</td></tr>`;
            }


            //Code for generating the "Insert new" button;
            //if the table should show a create new button
            let createNew = tables[activeTable].createNew;
            let edit = tables[activeTable].edit;
            let viewAnswers = tables[activeTable].viewAnswers;
            let restore = tables[activeTable].restore;
            if(createNew)
            {
                html +=
                `
                <tr>
                <a href="#" type="button" class="btn btn-success new-
button">Insert new</a>
                </tr>
                `
            }
            tableResultHolder.innerHTML = "";
            tableResultHolder.innerHTML += html;
            if(createNew){
                let newButton = document.querySelector(".new-button");
```

```
                newButton.addEventListener("click", function(e){
                    e.preventDefault();
                    createNew();
                })
            }
            if(edit){
                let editButtons = document.querySelectorAll(".edit-button");
                for(let button of editButtons){
                    button.addEventListener("click", function(e){
                        e.preventDefault();
                        let elemId = this.dataset.id;
                        edit(elemId);
                    })
                }
            }
            if(viewAnswers){
                let viewAnswerButtons = document.querySelectorAll(".view-
answers-button");
                for(let button of viewAnswerButtons){
                    addEventListenerOnce("click", button, function(e){
                        e.preventDefault();
                        let elemId = this.dataset.id;
                        let data = {questionId : elemId};
                        submitAjax(viewAnswers, showQuestionAnswers, data,
{closeModal : false});
                    })
                }
            }
            if(restore)
            {
                let restoreButtons = document.querySelectorAll(".restore-
button");
                for(let button of restoreButtons){
                    addEventListenerOnce("click", button, function(e){
                        e.preventDefault();
                        let elemId = this.dataset.id;
                        let data = {id : elemId};
                        submitAjax(restore, showResultGenerateTable, data,
{closeModal : false});
                    });
                }
            }
            let deleteButtons = document.querySelectorAll(".delete-button");
            for(let button of deleteButtons){
                button.addEventListener("click", function(){
```

```
                    let tab = this.dataset.table;
                    let elemId = this.dataset.id;
                    adminDeleteRequest(tab, elemId);
                })
            }
        }
        function generateHeaderTableRow(table, tableId){
            let headerTableRow = document.querySelector("#header-table-row");
            let headers = tables[tableId].headers;
            let html = "";
            html +=
            `
            <th>
            #
            </th>`
            for(let header of headers){
                html += `
              <th>
                ${header}
              </th>`
            }
            html +=
            `
            <th>
            Options
            </th>
            `
            headerTableRow.innerHTML = html;
        }
        function adminDeleteRequest(table, elementId){
            let data = {table, id : elementId};
            console.log(data);
            submitAjax("deleteFromTable", showResultGenerateTable, data,
{closeModal : false});
        }
        function showResultGenerateTable(data, args){
            generateTable();
            showResult(data, args);
        }
        function showUserModal(existingId = 0){
            setupUserModal();
            let modal = document.querySelector("#user-modal");
            globalData.currModal = modal;
            let type = existingId ? "edit" : "create";
```

```
            //Select all fields
            let userNameField = document.querySelector("#userName");
            let userLastNameField = document.querySelector("#userLastName");
            let userEmailField = document.querySelector("#userEmail");
            let userRoleField = document.querySelector("#userRole");
            let userPasswordField = document.querySelector("#userPassword");
            let userIdField = document.querySelector("#userId");


            let elems = new Array(userEmailField, userNameField,
userLastNameField, userRoleField, userPasswordField);


            //Remove success and error
            for(let elem of elems){
                removeError(elem, 1);
                removeSuccess(elem);
            }


            if(type == "edit"){
                let data = {id : existingId};
                userIdField.value = existingId;
                submitAjax("getSpecificUser", function(data){
                    let firstRow = data;


                    userNameField.value = firstRow.name;
                    userLastNameField.value = firstRow.lastName;
                    userEmailField.value = firstRow.email;
                    userRoleField.value = firstRow.role_id;
                }, data);
            }
            else{
                //Not implemented nor intended
            }
            openModal(modal, globalData.modalBackground)
        }
        function showLinkModal(existingId = 0){
            let modal = document.querySelector("#link-modal");
            globalData.currModal = modal;
            let type = existingId ? "edit" : "create";
            //Select all fields
```

```
            let linkTitleField = document.querySelector("#LinkTitle");
            let LinkHrefField = document.querySelector("#LinkHref");
            let LinkIconField = document.querySelector("#LinkIcon");
            let accessLevelSelect = document.querySelector("#LinkReqLevel");
            let LinkLocation = document.querySelector("#LinkLocation");
            let LinkRoot = document.querySelector("#LinkRoot");
            let linkIdField = document.querySelector("#linkId");
            let LinkPriorityField = document.querySelector("#linkPriority");


            let linkModalTitle = document.querySelector("#link-modal-title");
            let modalSubmitButton = document.querySelector("#link-submit");



            let elems = new Array(linkTitleField, LinkHrefField, LinkIconField,
accessLevelSelect, LinkLocation, LinkPriorityField);


            //Remove success and error
            for(let elem of elems){
                removeError(elem, 1);
                removeSuccess(elem);
            }
            if(type == "edit"){
                let data = {id : existingId};
                submitAjax("getSpecificLink", function(data){
                    let firstRow = data;


                    linkTitleField.value = firstRow.title;
                    LinkHrefField.value = firstRow.href;
                    LinkIconField.value = firstRow.icon ? firstRow.icon : "";
                    accessLevelSelect.value = firstRow.access_level_id;
                    LinkLocation.value = firstRow.location;
                    LinkRoot.checked = firstRow.landing > 0;
                    LinkPriorityField.value = firstRow.priority;
                }, data);


                linkIdField.value = existingId;
                modalSubmitButton.innerText = "Edit link";
                linkModalTitle.innerText = `Edit existing link`;
            }
            else{
```

```
                linkTitleField.value = "";
                LinkHrefField.value = "";
                LinkIconField.value = "";
                accessLevelSelect.value = 0;
                LinkLocation.value = 0;
                LinkRoot.checked = false;
                LinkPriorityField.value = 1;


                modalSubmitButton.innerText = "Create new link";
                linkModalTitle.innerText = "Create a new link";


                linkIdField.value = existingId;
            }
            openModal(modal, globalData.modalBackground)
        }
        function showBoroughModal(existingId = 0){
            let modal = document.querySelector("#borough-modal");


            let type = existingId ? "edit" : "create";


            let boroughNameField = document.querySelector("#boroughName");


            let boroughIdField = document.querySelector("#boroughId");


            boroughNameField.value = "";
            boroughIdField.value = existingId;


            let boroughTitle = document.querySelector("#borough-modal-title");
            let boroughSubmitButton = document.querySelector("#borough-
submit");


            if(type == "edit"){
                let data = {id : existingId};
                boroughTitle.innerText = "Edit borough";
                boroughSubmitButton.innerText = "Edit borough";
                submitAjax("getSpecificBorough", function(data){
                    console.log(data.title);
```

```
                    boroughNameField.value = data.title;
                }, data);
            }
            else{
                boroughTitle.innerText = "Create new borough";
                boroughSubmitButton.innerText = "Create borough";
            }



            let elems = new Array(boroughNameField);


            for(let elem of elems){
                removeError(elem, 1);
                removeSuccess(elem);
            }


            globalData.currModal = modal;
            openModal(modal, globalData.modalBackground);
        }
        function showMessageTypeModal(existingId = 0){
            let modal = document.querySelector("#messageType-modal");


            let type = existingId ? "edit" : "create";


            let messageTypeNameField =
document.querySelector("#messageTypeName");


            let messageTypeIdField = document.querySelector("#messageTypeId");


            messageTypeNameField.value = "";
            messageTypeIdField.value = existingId;


            let messageTypeTitle = document.querySelector("#messageType-modal-
title");
            let messageTypeSubmitButton = document.querySelector("#messageType-
submit");
```

```
            if(type == "edit"){
                let data = {id : existingId};
                messageTypeTitle.innerText = "Edit message type";
                messageTypeSubmitButton.innerText = "Edit message type";
                submitAjax("getSpecificMessageType", function(data){
                    console.log(data.title);
                    messageTypeNameField.value = data.title;
                }, data);
            }
            else{
                messageTypeTitle.innerText = "Create new message type";
                messageTypeSubmitButton.innerText = "Create message type";
            }


            let elems = new Array(messageTypeNameField);




            for(let elem of elems){
                removeError(elem, 1);
                removeSuccess(elem);
            }


            globalData.currModal = modal;
            openModal(modal, globalData.modalBackground);
        }
        function showBuildingTypeModal(existingId = 0){
            let modal = document.querySelector("#buildingType-modal");


            let type = existingId ? "edit" : "create";


            let buildingTypeNameField =
document.querySelector("#buildingTypeName");


            let buildingTypeId = document.querySelector("#buildingTypeId");
```

```
            buildingTypeNameField.value = "";
            buildingTypeId.value = existingId;


            let buildingTypeTitle = document.querySelector("#buildingType-
modal-title");
            let buildingTypeSubmitButton =
document.querySelector("#buildingType-submit");


            if(type == "edit"){
                let data = {id : existingId};
                buildingTypeTitle.innerText = "Edit building type";
                buildingTypeSubmitButton.innerText = "Edit building type";
                submitAjax("getSpecificBuildingType", function(data){
                    buildingTypeNameField.value = data.title;
                }, data);
            }
            else{
                buildingTypeTitle.innerText = "Create new building type";
                buildingTypeSubmitButton.innerText = "Create building type";
            }




            let elems = new Array(buildingTypeNameField);


            for(let elem of elems){
                removeError(elem, 1);
                removeSuccess(elem);
            }


            globalData.currModal = modal;
            openModal(modal, globalData.modalBackground);
        }
        function showRoomTypeModal(existingId = 0){
            let modal = document.querySelector("#roomType-modal");


            let type = existingId ? "edit" : "create";
```

```
            let roomTypeNameField = document.querySelector("#roomTypeName");


            let roomTypeId = document.querySelector("#roomTypeId");


            roomTypeNameField.value = "";
            roomTypeId.value = existingId;


            let roomTypeTitle = document.querySelector("#roomType-modal-
title");
            let roomTypeSubmitButton = document.querySelector("#roomType-
submit");


            if(type == "edit"){
                let data = {id : existingId};
                roomTypeTitle.innerText = "Edit room type";
                roomTypeSubmitButton.innerText = "Edit room type";
                submitAjax("getSpecificRoomType", function(data){
                    roomTypeNameField.value = data.title;
                }, data);
            }
            else{
                roomTypeTitle.innerText = "Create new room type";
                roomTypeSubmitButton.innerText = "Create room type";
            }




            let elems = new Array(roomTypeNameField);


            for(let elem of elems){
                removeError(elem, 1);
                removeSuccess(elem);
            }


            globalData.currModal = modal;
            openModal(modal, globalData.modalBackground);
```

```
        }
        function showListingModal(existingId = 0){
            setupListingModal();
            let modal = document.querySelector("#listing-modal");


            let type = existingId ? "edit" : "create";


            let listingTitleField = document.querySelector("#listingTitle");
            let listingDescriptionField =
document.querySelector("#listingDescription");
            let listingAddressField =
document.querySelector("#listingAddress");
            let listingSizeField = document.querySelector("#listingSize");
            let listingPriceField = document.querySelector("#listingPrice");
            let listingBoroughSelect =
document.querySelector("#listingBorough");
            let listingBuildingTypeSelect =
document.querySelector("#listingBuildingType");
            let listingPhotoField = document.querySelector("#listingPhoto");
            let listingIdField = document.querySelector("#listingId");
            let listingImagePreview = document.querySelector("#main-photo-
preview");


            let listingModalTitle = document.querySelector("#listing-modal-
title");
            let modalSubmitButton = document.querySelector("#listing-submit");


            let elems = new Array(listingTitleField, listingDescriptionField,
listingAddressField, listingSizeField, listingPriceField, listingBoroughSelect,
listingBuildingTypeSelect, listingPhotoField);


            for(let elem of elems){
                removeError(elem, 1);
                removeSuccess(elem);
            }


            listingIdField.value = existingId;
            removeAllRooms();
            listingImagePreview.src = "";
```

```
            if(type == "edit"){
                let data = {id : existingId};
                submitAjax("getSpecificListing", function(data){
                    console.log(data);
                    let core = data.main;
                    let photo = data.photo;
                    let rooms = data.rooms;
                    listingTitleField.value = core.listing_name;
                    listingAddressField.value = core.address;
                    listingDescriptionField.value = core.description;
                    listingSizeField.value = core.size;
                    listingPriceField.value = core.price;
                    listingBoroughSelect.value = core.borough_id;
                    listingBuildingTypeSelect.value = core.building_type_id;
                    listingImagePreview.src =
`../resources/imgs/${photo.path}`;

                    globalData.startingRooms = new Array();
                    for(let room of rooms){
                        addRoom(room.room_type_id, room.room_name,
room.numberOf);
                        globalData.startingRooms.push({roomId :
room.room_type_id, count : room.numberOf});
                    }
                }, data);
                listingModalTitle.innerText = "Edit listing";
                modalSubmitButton.innerText = "Edit listing";
            }
            else{
                listingTitleField.value = "";
                listingAddressField.value = "";
                listingDescriptionField.value = "";
                listingSizeField.value = 30;
                listingPriceField.value = 1000;
                listingBoroughSelect.value = 0;
                listingBuildingTypeSelect.value = 0;
                listingPhotoField.value = "";
                listingModalTitle.innerText = "Create new listing";
                modalSubmitButton.innerText = "Create listing";
            }


            globalData.currModal = modal;
```

```
            openModal(modal, globalData.modalBackground);
        }
        function showQuestionModal(existingId = 0){
            let modal = document.querySelector("#question-modal");
            let modalTitle = document.querySelector("#question-modal-title");
            let modalSubmitButton = document.querySelector("#question-submit");
            let answerHolder = document.querySelector("#question-answer-
holder");
            let questionNameField = document.querySelector("#questionName");
            let questionIdField = document.querySelector("#questionId");


            questionIdField.value = existingId;


            let type = existingId ? "edit" : "create";


            globalData.numberOfAnswers = 1;


            questionNameField.value = "";
            answerHolder.innerHTML = "";
            if(type == "edit"){
                modalTitle.innerText = "Edit survey question";
                modalSubmitButton.innerText = "Edit survey question";
                let data = {questionId : existingId};
                submitAjax("getSpecificQuestion", function(data){
                    let question = data["name"];
                    questionNameField.value = question;
                    let answers = data["answers"];
                    for(let answer of answers){
                        addAnswer(globalData.numberOfAnswers++,
answer["answer"], answer["answer_id"]);
                    }
                }, data);
            }
            else{
                modalTitle.innerText = "Create new survey question";
                modalSubmitButton.innerText = "Create new question";
            }



            globalData.currModal = modal;
```

```
            openModal(modal, globalData.modalBackground);
        }
        function showQuestionAnswers(data){
            let modal = document.querySelector("#question-results-modal");


            let questionTitleField = document.querySelector("#question-title");
            let questionResultsField = document.querySelector("#question-
results");


            let questionTitle = data["name"];
            let questionAnswers = data["answers"];


            questionTitleField.innerText = `Question: ${questionTitle}?`;
            questionResultsField.innerHTML = "";
            for(let answer of questionAnswers){
                questionResultsField.innerHTML += addResult(answer["answer"],
answer["count"]);
            }


            globalData.currModal = modal;
            openModal(modal, globalData.modalBackground);
        }
        //Setup functions
        function setUpModals(){
            let cancelButtons = document.querySelectorAll(".close-button");
            for(let button of cancelButtons){
                button.addEventListener("click", closeCurrentModal);
            }
            setupLinkModal();
            setupUserModal();
            setupListingModal();
            setupBoroughModal();
            setupBuildingTypeModal();
            setupRoomTypeModal();
            setupQuestionModal();
            setupMessageTypeModal();
        }
        function setupUserModal(){
            let userRoleSelect = document.querySelector("#userRole");
            readAjax("getAllRoles", fillDropdown, [userRoleSelect]);
```

```
            let modalSubmitButton = document.querySelector("#user-submit");
            addEventListenerOnce("click", modalSubmitButton, function(e){
                e.preventDefault();
                submitUserForm();
            })
        }
        function setupListingModal(){
            let boroughSelect = document.querySelector("#listingBorough");
            let listingBuildingType =
document.querySelector("#listingBuildingType");
            let listingRoomsList = document.querySelector("#listingRoomsList");


            readAjax("getAllBoroughs", fillDropdown, [boroughSelect]);
            readAjax("getAllBuildingTypes", fillDropdown,
[listingBuildingType]);
            readAjax("getAllRoomTypes", fillDropdown, [listingRoomsList]);



            let fileReader = new FileReader();
            let previewHolder = document.querySelector("#main-photo-preview");
            fileReader.onload = function (e) {previewHolder.src = this.result;}
            let listingPhotoField = document.querySelector("#listingPhoto");
            addEventListenerOnce("change", listingPhotoField, function(e){
                fileReader.readAsDataURL(listingPhotoField.files[0]);
            })


            let addListingRoomButton =
document.querySelector("#addListingRoomButton");
            addEventListenerOnce("click", addListingRoomButton, function(e){
                e.preventDefault();
                let roomId =
listingRoomsList.options[listingRoomsList.selectedIndex].value;
                let roomText =
listingRoomsList.options[listingRoomsList.selectedIndex].text;
                if(roomId==0){
                    addError(listingRoomsList, "Must select a room before
adding");
                    removeSuccess(listingRoomsList);
                    return;
                }
```

```
                removeError(listingRoomsList, 1);
                addRoom(roomId, roomText);
            });


            let modalSubmitButton = document.querySelector("#listing-submit");
            addEventListenerOnce("click", modalSubmitButton, function(e){
                e.preventDefault();
                submitListingForm();
            });
        }
        function setupLinkModal(){
            let accessLevelSelect = document.querySelector("#LinkReqLevel");
            readAjax("getAllAccessLevels", fillDropdown, [accessLevelSelect]);


            let locationSelect = document.querySelector("#LinkLocation");
            readAjax("getAllNavigationLocations", fillDropdown,
[locationSelect, true]);


            let modalSubmitButton = document.querySelector("#link-submit");
            addEventListenerOnce("click", modalSubmitButton, function(e){
                e.preventDefault();
                submitLinkForm();
            });
        }
        function setupBoroughModal(){
            let modalSubmitButton = document.querySelector("#borough-submit");
            addEventListenerOnce("click", modalSubmitButton, function(e){
                e.preventDefault();
                submitBoroughForm();
            });
        }
        function setupBuildingTypeModal(){
            let modalSubmitButton = document.querySelector("#buildingType-
submit");
            addEventListenerOnce("click", modalSubmitButton, function(e){
                e.preventDefault();
                submitBuildingTypeForm();
            })
        }
        function setupRoomTypeModal(){
            let modalSubmitButton = document.querySelector("#roomType-submit");
            addEventListenerOnce("click", modalSubmitButton, function(e){
```

```
                e.preventDefault();
                submitRoomTypeForm();
            })
        }
        function setupMessageTypeModal(){
            let modalSubmitButton = document.querySelector("#messageType-
submit");
            addEventListenerOnce("click", modalSubmitButton, function(e){
                e.preventDefault();
                submitMessageTypeForm();
            })
        }
        function setupQuestionModal(){
            let questionAddAnswerButton =
document.querySelector("#questionAddAnswer");
            addEventListenerOnce("click", questionAddAnswerButton, function(e){
                e.preventDefault();
                addAnswer(globalData.numberOfAnswers++, "");
            })


            let modalSubmitButton = document.querySelector("#question-submit");
            addEventListenerOnce("click", modalSubmitButton, function(e){
                e.preventDefault();
                submitQuestionForm();
            })
        }
        function submitUserForm(){
            let userNameField = document.querySelector("#userName");
            let userLastNameField = document.querySelector("#userLastName");
            let userEmailField = document.querySelector("#userEmail");
            let userRoleField = document.querySelector("#userRole");
            let userPasswordField = document.querySelector("#userPassword");
            let userIdField = document.querySelector("#userId");


            let errors = 0;


            let reName = /^[A-Z][a-z]{1,14}(\s[A-Z][a-z]{1,14}){0,2}$/;


            let rePass1 = /[A-Z]/;
            let rePass2 = /[a-z]/;
            let rePass3 = /[0-9]/;
```

```
            let rePass4 = /[!\?\.]/;
            let rePass5 = /^[A-Za-z0-9!\?\.]{7,30}$/;


            let reEmail = /^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;


            let singleTests = [
                {re : reName, field: userNameField, message : `Name doesn't
match format, eg "Nathan"`},
                {re : reName, field: userLastNameField, message : `Last name
does not match format, eg "Smith"`},
                {re : reEmail, field: userEmailField, message : `Email not
valid`},
            ]


            let passwordTests = [
                {re : rePass5, field: userPasswordField, message : `Password
must be between 7 and 30 characters long`},
                {re : rePass1, field: userPasswordField, message : `Password
must contain an uppercase letter`},
                {re : rePass2, field: userPasswordField, message : `Password
must contain a lowercase letter`},
                {re : rePass3, field: userPasswordField, message : `Password
must contain a digit`},
                {re : rePass4, field: userPasswordField, message : `Password
must contain ! or ? or .`}
            ]


            for(let test of singleTests){
                errors += reTestText(test.re, test.field, test.message);
            }

            if(userPasswordField.value.length > 0){
                for(let test of passwordTests){
                    let result = reTestText(test.re, test.field, test.message);
                    errors += result;

                    if(result > 0) break;
                }
            }
```

```
            errors += testDropdown(userRoleField, 0, "You must select a role");


            if(userIdField.value < 1) errors++;


            if(errors != 0) return;


            data = {};


            target = "editUser";


            data.userId = userIdField.value;


            data.name = userNameField.value;
            data.lastName = userLastNameField.value;
            data.email = userEmailField.value;
            data.password = userPasswordField.value;
            data.roleId = userRoleField.value;


            submitAjax(target, showResultGenerateTable, data, {closeModal :
true});
        }
        function submitListingForm(){
            let formData = new FormData();


            let listingTitleField = document.querySelector("#listingTitle");
            let listingDescriptionField =
document.querySelector("#listingDescription");
            let listingAddressField =
document.querySelector("#listingAddress");
            let listingSizeField = document.querySelector("#listingSize");
            let listingPriceField = document.querySelector("#listingPrice");
            let listingBoroughSelect =
document.querySelector("#listingBorough");
            let listingBuildingTypeSelect =
document.querySelector("#listingBuildingType");
            let listingPhotoField = document.querySelector("#listingPhoto");
            let listingIdField = document.querySelector("#listingId");
```

```
            let listingId = listingIdField.value;

            let type = listingId > 0 ? "edit" : "create";

            let target = "createNewListing";

            let errors = 0;

            if(type == "edit"){
                formData.append("listingId", listingId);
                target = "editListing";
            }

            let reTitle = /^[A-Z][a-z]{2,15}(\s[A-Za-z][a-z]{2,15}){0,2}$/;
            let reAddress = /^(([A-Z][a-z\d']+)|([0-9][1-9]*\.?))(\s[A-Za-
z\d][a-z\d']+){0,7}\s(([1-9][0-9]{0,5}[\/-]?[A-Z])|([1-9][0-
9]{0,5})|(NN))\.?$/;
            let reDescription = /^[A-Z][a-z']{0,50}(\s[A-Za-z][a-z']{0,50})*$/;

            //tests

            if(reTestText(reTitle, listingTitleField, `Title does not match
format, eg "Great New Listing" (One to three words)`, 1)) errors++;

            if(reTestText(reDescription, listingDescriptionField, `Description
does not match format, only words are allowed in descriptions`, 1)) errors++;

            if(reTestText(reAddress, listingAddressField, `Address does not
match format, eg "First Street 20"`, 1)) errors++;

            if(testGeneric(listingSizeField, listingSizeField.value < 30, "Size
cannot be below 30 feet", 1)) errors++;
```

```
            if(testGeneric(listingSizeField, listingSizeField.value > 100000,
"Size cannot be above 100000 feet", 1)) errors++;


            if(testGeneric(listingPriceField, listingPriceField.value < 1000,
"Price cannot be below 1000$", 1)) errors++;


            if(testGeneric(listingPriceField, listingPriceField.value >
1000000000, "Price cannot be above 1000000000$", 1)) errors++;


            if(testDropdown(listingBoroughSelect, 0, "You must select a
borough", 1)) errors++;


            if(testDropdown(listingBuildingTypeSelect, 0, "You must select a
building type", 1)) errors++;


            if(type === "create"){
                if(testImage(listingPhotoField, "")) {
                    console.log("Here");
                    errors++
                };
            }
            let roomSelects = document.querySelectorAll(".listingRoom");
            let arrayOfRooms = new Array();
            for(let roomElement of roomSelects){
                if(testGeneric(roomElement, roomElement.value < 1, "Number of
rooms cannot be less than one", 2)) {
                    errors++;
                    continue;
                }
                arrayOfRooms.push({roomId : parseInt(roomElement.dataset.id),
count : parseInt(roomElement.value)});
            }


            let listingRooms = JSON.stringify(arrayOfRooms);


            console.log(`Step one, errors = ${errors}`);
```

```
            //On success
            if(errors !== 0) return;


            if(listingPhotoField.value != ""){
                formData.append("listingPhoto", listingPhotoField.files[0]);
            }
            formData.append("listingTitle", listingTitleField.value);
            formData.append("listingDescription",
listingDescriptionField.value);
            formData.append("listingAddress", listingAddressField.value);
            formData.append("listingSize", listingSizeField.value);
            formData.append("listingPrice", listingPriceField.value);
            formData.append("listingBorough", listingBoroughSelect.value);
            formData.append("listingBuildingType",
listingBuildingTypeSelect.value);


            if(arrayOfRooms.length > 0){
                formData.append("listingRooms", listingRooms);
            }
            console.log("Step two");


            submitFormDataAjax(target, showResultGenerateTable, formData,
{closeModal : true});
        }
        function submitLinkForm(){
            let LinkIdField = document.querySelector("#linkId");
            let LinkTitleField = document.querySelector("#LinkTitle");
            let LinkHrefField = document.querySelector("#LinkHref");
            let LinkIconField = document.querySelector("#LinkIcon");
            let accessLevelSelect = document.querySelector("#LinkReqLevel");
            let LinkLocationSelect = document.querySelector("#LinkLocation");
            let LinkPriorityField = document.querySelector("#linkPriority");
            let LinkRootCheck = document.querySelector("#LinkRoot");



            let linkId = LinkIdField.value;
            let linkTitle = LinkTitleField.value;
            let LinkHref = LinkHrefField.value;
            let LinkIcon = LinkIconField.value;
```

```
            let accessLevel = accessLevelSelect.value;
            let LinkLocation = LinkLocationSelect.value;
            let LinkRoot = LinkRootCheck.checked;
            let linkPriority = LinkPriorityField.value;


            let reTitle = /^[A-Z][a-z]{2,15}(\s[A-Za-z][a-z]{2,15}){0,2}$/;
            let reHref = /(https?:\/\/(www\.)?[-a-zA-Z0-9@:%._\+~#=]{1,256}\.[-
a-zA-Z0-9()]{1,6}\b([-a-zA-Z0-9()@:%_\+.~#?&\/=]*))||(^[a-z]{3,40}\.[a-
z]{2,5}$)/;
            let reIcon = /(^$)|(^[a-z:-]{5,30}$)/;


            let submitType = linkId > 0 ? "edit" : "create";


            let data = {};


            let target = "createNewLink";


            if(submitType === "edit"){
                data.linkId = linkId;
                target = "editLink";
            }

            let errors = 0;


            if(reTestText(reTitle, LinkTitleField, "Title does not match
format", 1)) errors++;


            if(reTestText(reHref, LinkHrefField, "Link href does not match
format", 1)) errors++;


            if(reTestText(reIcon, LinkIconField, "Link icon does not match
format", 1)) errors++;


            if(testDropdown(accessLevelSelect, 0, "You must select an access
level", 1)) errors++;
```

```
            if(testDropdown(LinkLocationSelect, 0, "You must select a
location", 1)) errors++;


            if(testGeneric(LinkPriorityField, linkPriority < 1, "Link priority
cannot be lower than 1", 1)) errors++;


            if(testGeneric(LinkPriorityField, linkPriority > 99, "Link priority
cannot be higher than 99", 1)) errors++;


            if(errors != 0){
                return;
            }

            data.title = linkTitle;
            data.href = LinkHref;
            data.icon = LinkIcon;
            data.aLevel = accessLevel;
            data.location = LinkLocation;
            data.main = LinkRoot;
            data.priority = linkPriority;


            submitAjax(target, showResultGenerateTable, data, {closeModal :
true});
        }
        function submitBoroughForm(){
            let boroughNameField = document.querySelector("#boroughName");
            let boroughIdField = document.querySelector("#boroughId");


            let boroughId = boroughIdField.value;


            let reBoroughName = /^[A-Z][a-z']{1,50}(\s[A-Za-z][a-
z']{1,50}){0,3}$/;


            let errors = 0;
```

```
            let type = boroughId > 0 ? "edit" : "create";


            let target = "createNewBorough";


            let data = {};


            if(type == "edit"){
                target = "editBorough";
                data.id = boroughId;
            }


            if(reTestText(reBoroughName, boroughNameField, `Borough name does
not match format, eg "The Queens"`)) errors ++;


            if(errors != 0){
                return;
            }


            data.boroughName = boroughNameField.value;


            submitAjax(target, showResultGenerateTable, data, {closeModal :
true});
            // submitAjax(target, callMultipleFunctions, data,
[showResultGenerateTable, setupListingModal]);
        }

  function submitBuildingTypeForm(){
            let buildingTypeNameField =
document.querySelector("#buildingTypeName");
            let buildingTypeIdField =
document.querySelector("#buildingTypeId");


            let buildingTypeId = buildingTypeIdField.value;


            let reBuildingTypeName = /^[A-Z][a-z']{1,50}(\s[A-Za-z][a-
z']{1,50}){0,3}$/;
```

```
            let errors = 0;


            let type = buildingTypeId > 0 ? "edit" : "create";


            let target = "createNewBuildingType";


            let data = {};


            if(type == "edit"){
                target = "editBuildingType";
                data.id = buildingTypeId;
            }


            if(reTestText(reBuildingTypeName, buildingTypeNameField, `Building
type name does not match format, eg "House"`)) errors ++;


            if(errors != 0){
                return;
            }


            data.buildingTypeName = buildingTypeNameField.value;


            submitAjax(target, showResultGenerateTable, data, {closeModal :
true});
        }
        function submitRoomTypeForm(){
            let roomTypeNameField = document.querySelector("#roomTypeName");
            let roomTypeIdField = document.querySelector("#roomTypeId");


            let roomTypeId = roomTypeIdField.value;


            let reRoomTypeName = /^[A-Z][a-z']{1,50}(\s[A-Za-z][a-
z']{1,50}){0,3}$/;
```

```
            let errors = 0;


            let type = roomTypeId > 0 ? "edit" : "create";


            let target = "createNewRoomType";


            let data = {};


            if(type == "edit"){
                target = "editRoomType";
                data.id = roomTypeId;
            }


            if(reTestText(reRoomTypeName, roomTypeNameField, `Room type name
does not match format, eg "Livingroom"`)) errors ++;


            if(errors != 0){
                return;
            }


            data.roomTypeName = roomTypeNameField.value;


            submitAjax(target, showResultGenerateTable, data, {closeModal :
true});
        }
        function submitQuestionForm(){
            let questionNameField = document.querySelector("#questionName");
            let questionIdField = document.querySelector("#questionId");


            let reQuestion = /^[A-Z][a-z']{1,20}(\s[A-Za-z][a-
z']{0,20}){2,10}$/;
            let reAnswer = /^[A-Z][a-z']{0,20}(\s[A-Za-z][a-z']{0,20}){2,10}$/;
```

```
            let type = questionIdField.value > 0 ? "edit" : "create";


            let target = "createNewQuestion";


            let data = {};


            if(type == "edit"){
                data.questionId = questionIdField.value;
                target = "editQuestion"
            }


            let providedAnswers = document.querySelectorAll(".questionAnswer");


            let errors = 0;


            if(reTestText(reQuestion, questionNameField, `Question does not fit
format, eg: "Do you like this website"`)) errors++;


            let arrayOfAnswers = new Array();
            for(let answer of providedAnswers){
               if(reTestText(reAnswer, answer, `Answer does not fit format, eg:
"Yes I do" (at least three words)`, 2)){
                errors++;
                continue;
               };
               if(type == "create"){
                arrayOfAnswers.push(answer.value);
               }
               else{
                arrayOfAnswers.push({answerId : parseInt(answer.dataset.id),
text : answer.value});
               }
            };


            if(errors != 0){
                return;
            }
```

```
            let addAnswerButton = document.querySelector("#questionAddAnswer");
            if(arrayOfAnswers.length < 2){
                addError(addAnswerButton, "Must provide at least two answers
before submitting", 1);
                errors++;
            }
            else{
                removeError(addAnswerButton, 1);
                addSuccess(addAnswerButton);
            }


            if(errors != 0){
                return;
            }


            data.questionName = questionNameField.value;
            if(type == "create"){
                data.questionAnswers = arrayOfAnswers;
            }
            else{
                data.newAnswers = arrayOfAnswers.filter(answer =>
answer.answerId === 0);
                data.modifiedAnswers = arrayOfAnswers.filter(answer =>
answer.answerId != 0);
            }


            submitAjax(target, showResultGenerateTable, data, {closeModal :
true});
        }
        function submitMessageTypeForm(){
            let messageTypeNameField =
document.querySelector("#messageTypeName");
            let messageTypeIdField = document.querySelector("#messageTypeId");


            let messageTypeId = messageTypeIdField.value;


            let reMessageTypeName = /^[A-Z][a-z]{1,19}$/;
```

```
            let errors = 0;


            let type = messageTypeId > 0 ? "edit" : "create";


            let target = "createNewmessageType";


            let data = {};


            if(type == "edit"){
                target = "editMessageType";
                data.id = messageTypeId;
            }


            if(reTestText(reMessageTypeName, messageTypeNameField, `Message
type name has to be one capitalized word`)) errors ++;


            if(errors != 0){
                return;
            }


            data.messageTypeName = messageTypeNameField.value;


            submitAjax(target, showResultGenerateTable, data, {closeModal :
true});
        }
        function addResult(text, count){
            return `<tr><td>${text}</td><td>${count}</td></tr>`
        }
        function addAnswer(num, answerText, answerId = 0){
            let answerHolder = document.querySelector("#question-answer-
holder");
            let newAnswerHolder = document.createElement("div");


            let html =
```

```
            `
            <label for="answer${num}" class="d-block">Answer ${num}</label>
            <input type="text" value="${answerText}" class="form-control d-
inline questionAnswer w-50" data-id="${answerId}" name="answer${num}"
id="answer${num}">
            <a href="#" class="btn btn-danger d-inline removeAnswer">Remove</a>
            <span class="error-msg hidden d-block"></span>
            `
            newAnswerHolder.innerHTML += html;
            answerHolder.appendChild(newAnswerHolder);
            let removeButtons = document.querySelectorAll(".removeAnswer");


            for(let elem of removeButtons){
                addEventListenerOnce("click", elem, function(e){
                    e.preventDefault();
                    let parentElement = this.parentElement;
                    parentElement.remove();
                })
            }
        }
        function addRoom(roomId, roomText, count = 1){
            let html = "";
            let roomHolder = document.querySelector("#room-holder");
            let existingInputField = document.querySelector(`#room${roomId}`);
            if(existingInputField){
                let currValue = existingInputField.value;
                existingInputField.value = parseInt(currValue) + 1;
                return;
            }
            let newRoomHolder = document.createElement("div");
            html +=
            `<label for="room${roomId}" class="d-block">${roomText}</label>
            <input type="number" value="${count}" min="1" class="form-control
d-inline listingRoom w-50" data-id="${roomId}" name="listingRoom${roomId}"
id="room${roomId}">
            <a href="#" class="btn btn-danger d-inline removeRoom"
id="removeButton${roomId}">Remove</a>
            <span class="error-msg hidden d-block"></span>`
            newRoomHolder.innerHTML += html;
            roomHolder.appendChild(newRoomHolder);
            let removeButtons = document.querySelectorAll(`.removeRoom`);
            for(let elem of removeButtons){
                addRemoveParentOnClickListener(elem);
            }
```

```
        }
        function removeAllRooms(){
            let roomHolder = document.querySelector("#room-holder");
            roomHolder.innerHTML = "";
        }
        function addRemoveParentOnClickListener(element){
            element.addEventListener("click", function(e){
            e.preventDefault();
            let parentElement = this.parentElement;
            parentElement.remove();
        });
        }
    }
    if(currentPage === "contact.html"){
        let messageTypeSelect = document.querySelector("#messageType");
        let messageTitleField = document.querySelector("#messageTitle");
        let messageBodyField = document.querySelector("#messageBody");
        readAjax("getAllMessageTypes", fillDropdown, [messageTypeSelect]);


        let sendMessageButton = document.querySelector("#sendMessage");
        addEventListenerOnce("click", sendMessageButton, function(e){
            e.preventDefault();
            sendMessage();
        })


        function sendMessage(){
            let data = {};


            let errors = 0;


            let reTitle = /^[A-Z][a-z']{0,19}(\s[A-Za-z][a-z']{0,20}){1,4}$/;
            let reBody = /^[A-Z][a-z']{0,19}(\s[A-Za-z][a-z']{0,20}){2,14}$/;


            if(reTestText(reTitle, messageTitleField, "Message title does not
match format (Between two and five words, no punctuation)", 1)) errors++;


            if(reTestText(reBody, messageBodyField, "Message body does not
match format (Between three and fifteen words, no punctuation)", 1)) errors++;
```

```
            if(testDropdown(messageTypeSelect, 0, "You must select a message
type", 1));


            if(errors != 0) return;


            data.message_type_id = messageTypeSelect.value;
            data.title = messageTitleField.value;
            data.body = messageBodyField.value;


            submitAjax("createNewMessage", showResult, data);
        }
    }
    if(currentPage === "survey.html"){
        getQuestionsForUser();
    }
    if(currentPage === "listings.html"){
        let data = {};


        let args = {};


        let boroughArgs = {checkboxHolder :
document.querySelector("#boroughFilters"), checkboxName : "boroughFilter"};


        let buildingTypeArgs = {checkboxHolder :
document.querySelector("#buildingTypeFilters"), checkboxName :
"buildingTypeFilter"};


        args.listingHolder = document.querySelector("#listingHolder");


        let queryString = window.location.search;


        let urlParams = new URLSearchParams(queryString);
```

```
        let boroughid = urlParams.get("boroughid");


        let buildingTypeid = urlParams.get("buildingtypeid");


        if(boroughid != null){
            let id = Number(boroughid);
            let boroughFilter = [id];
            saveToLocalStorage(boroughFilter, "boroughFilter");
        }


        if(buildingTypeid != null){
            let buildingFilter = [Number(buildingTypeid)];
            saveToLocalStorage(buildingFilter, "buildingTypeFilter");
        }


        let readBoroughs = readFromLocalStorage("boroughFilter");


        if(readBoroughs){
            data.boroughFilter = readBoroughs;
        }


        let readBuildingType = readFromLocalStorage("buildingTypeFilter");


        if(readBuildingType){
            data.buildingTypeFilter = readBuildingType;
        }


        let selectedSort = readFromLocalStorage("selectedSort");


        if(selectedSort){
            data.sortType = parseInt(selectedSort);
            document.querySelector("#listingsSort").value = selectedSort;
        }


        readAjax("getAllBoroughsWithListings", fillCheckbox, boroughArgs);
```

```
        readAjax("getAllBuildingTypesWithListings", fillCheckbox,
buildingTypeArgs);


        //Get preexisting filters from local storage, then query for listings


        submitAjax("getListingsForFilter", displayListings, data, args);


        let searchButton = document.querySelector("#searchListings");
        addEventListenerOnce("click", searchButton, function(e){
            e.preventDefault();
            sendFiltersDisplayListings();
        })
    }
    if(currentPage === "favorites.html"){
        let data = {};


        let args = {};


        data.onlyFavorite = true;


        args.listingHolder = document.querySelector("#listingHolder");


        args.noListingsMessage = "No listings saved as favorite"


        submitAjax("getListingsForFilter", displayListings, data, args);
    }
    if(currentPage === "listing.html"){
        let queryString = window.location.search;


        let urlParams = new URLSearchParams(queryString);


        let id = urlParams.get("listing_id");
```

```
        console.log(id);

        if(!id) redirect({ newLocation : "listings.html", landing : false});

        data = {};

        data.listing_id = id;

        args = {};

        args.errorFunction = showListingNotFound;

        submitAjax("getSpecificDetailedListing", showSingleListing, data,
args);
    }
    if(currentPage === "index.html"){
        let boroughSelect = document.querySelector("#landing-input");
        readAjax("getAllBoroughsWithListings", fillDropdown, [boroughSelect]);

        let searchButton = document.querySelector("#landing-search");
        addEventListenerOnce("click", searchButton, function(e){
            e.preventDefault();
            let selectedId = boroughSelect.value;

            if(selectedId == 0){
                redirect({newLocation : "listings.html", landing : false});
            }
            else{
                redirect({ newLocation :
`listings.html?boroughid=${selectedId}`, landing : false});
            }
        })
    }
}
```

```
function showSingleListing(data){
    let listingHolder = document.querySelector("#singleListingHolder");


    let body = data["body"];
    let img = data["img"];
    let rooms = data["rooms"];
    let number = data["number"];
    let favorite = data["body"]["favorite"] > 0;


    html = "";


//     <div class="col-12 col-md-4">
//     <div class="listing-img w-100">
//       <img src="../resources/imgs/168217066120666745526443e325515b3.jpg"
alt="" class="img-fluid mk-main-img"/>
//         <a href="#" data-id="4" class="mk-favorite-icon-holder">
//           <span class="iconify mk-favorite-icon" data-icon="mdi:cards-heart-
outline">Heart</span>
//         </a>
//     </div>
//   </div>
//   <div class="col-12 col-md-8" id="listingHolder">
//     <div class="listing-body">
//       <h3 class="">Listing title</h3>
//       <p class="">Lorem ipsum dolor sit amet consectetur, adipisicing elit.
Tempore velit magni necessitatibus voluptates minus nostrum sunt vel, beatae
delectus eligendi.</p>
//       <p>Borough: Queens</p>
//       <p>Building type: Duplex</p>
//       <p>Size: 3000 feet</p>
//     </div>
// <div class="d-flex justify-content-between">
    //   <div class="listing-rooms listing-padding">
    //     <h3>Listing rooms</h3>
    //     <p class="d-inline">Livingroom: 5</p>
    //     <p class="d-inline">Bedroom: 5</p>
    //     <p class="d-inline">Bathroom: 5</p>
    //   </div>
//       <div class="contact listing-padding">
```

```
//         <a href="#" class="btn btn-success">Copy phone number</a>
//         <p>Price: 3000$</p>
//       </div>
//     </div>
//   </div>


    html +=
    `
    <div class="col-12 col-md-4">
    <div class="listing-img w-100 listing-img-height">
      <img src="../resources/imgs/${img}" alt="${body["listing_name"]} img"
class="img-fluid mk-main-img"/>
    <a href="#" data-id="${body["id"]}" class="mk-favorite-icon-holder">
        <span class="iconify mk-favorite-icon" data-icon="${favorite ?
"mdi:cards-heart": "mdi:cards-heart-outline"}">Heart</span>
    </a>
    </div>
    </div>
    <div class="col-12 col-md-8" id="listingHolder">
    <div class="listing-body">
      <h3 class="">${body["listing_name"]}</h3>
      <p class="">${body["description"]}</p>
      <a href="listings.html?boroughid=${body["borough_id"]}">Borough:
${body["borough"]}</a>
      </br>
      <a href="listings.html?buildingtypeid=${body["type_id"]}">Building type:
Duplex</a>
      <p>Size: ${body["size"]} feet</p>
    </div>
    <div class="d-flex justify-content-between">
    `
    if(rooms.length > 0){
        html +=
        `
        <div class="listing-rooms listing-padding">
        <h3>Rooms:</h3>
        `
        for(let room of rooms){
            html+=
            `
            <p class="d-inline">${room["room_name"]}: ${room["numberOf"]}</p>
            `
        }
        html +=
```

```
        `
        </div>
        `
    }
    html +=
    `
    <div class="contact listing-padding">
    <a href="#" class="btn soft-blue" id="copy-number-button" data-
number="${number}">Copy phone number</a>
    <p>Price: ${body["price"]}$</p>
  </div>
</div>
</div>
    `

    listingHolder.innerHTML = html;


    addFavoriteFunctionality();


    let copyNumberButton = document.querySelector("#copy-number-button");


    addEventListenerOnce("click", copyNumberButton, function(e){
        e.preventDefault();
        let num = this.dataset.number;
        navigator.clipboard.writeText(num);
        showSuccess("Successfully copied phone number " + num);
    })


    console.log(data);
}


function addFavoriteFunctionality(){
    let favoriteButtons = document.querySelectorAll(".mk-favorite-icon-
holder");
    for(let button of favoriteButtons){
        addEventListenerOnce("click", button, function(e){
            e.preventDefault();
            let idOfListing = this.dataset.id;
            let iconHolder = this.firstElementChild;
```

```
            let newIcon = iconHolder.dataset.icon === "mdi:cards-heart" ?
"mdi:cards-heart-outline" : "mdi:cards-heart";


            let data = {};


            data.listingId = idOfListing;


            args = {};


            args.additionalFunctions = new Array(flipListingFavoriteIcon)


            args.additionalFunctionArgs = {idOfListing};


            if(newIcon === "mdi:cards-heart"){
                submitAjax("addToFavoriteListings", showResult, data, args);
            }
            else{
                submitAjax("removeFromFavoriteListings", showResult, data,
args);
            }
        })
    }
}


function showListingNotFound(data){
    errorHandler(data);
    let listingHolder = document.querySelector("#singleListingHolder");
    listingHolder.innerHTML =  `<p class="h3">${data}</p>`;
}


function sendFiltersDisplayListings(){
    let data = {};


    let args = {};
```

```
    args.listingHolder = document.querySelector("#listingHolder");


    let selectedBoroughs = new Array();
    let boroughFilters = document.querySelectorAll(".boroughFilter");


    for(let borough of boroughFilters){
        if(borough.checked){
            selectedBoroughs.push(parseInt(borough.value));
        }
    }


    data.boroughFilter = selectedBoroughs;


    let selectedBuildingTypes = new Array();
    let buildingTypeFilters = document.querySelectorAll(".buildingTypeFilter");


    for(let buildingType of buildingTypeFilters){
        if(buildingType.checked){
            selectedBuildingTypes.push(parseInt(buildingType.value));
        }
    }

    let selectedSort = parseInt(document.querySelector("#listingsSort").value);


    saveToLocalStorage(selectedSort, "selectedSort");


    saveToLocalStorage(selectedBoroughs, "boroughFilter");


    saveToLocalStorage(selectedBuildingTypes, "buildingTypeFilter");


    data.buildingTypeFilter = selectedBuildingTypes;


    data.sortType = selectedSort;
```

```
    let titleFilter = document.querySelector("#titleFilter");


    data.titleFilter = titleFilter.value;


    submitAjax("getListingsForFilter", displayListings, data, args);
}


function saveToLocalStorage(data, name){
    localStorage.setItem(name, JSON.stringify(data));
}


function readFromLocalStorage(name){
    return JSON.parse(localStorage.getItem(name));
}


function fillCheckbox(data, args){
    let checkboxHolder = args.checkboxHolder;
    let checkboxName = args.checkboxName;
    let localStoragePresence = readFromLocalStorage(args.checkboxName);
    let html = "";
    let checked = "";
    for(let row of data){
    if(localStoragePresence){
        checked = "";
        for(let elem of localStoragePresence){
            if(elem == row["id"]) checked = `checked="checked"`;
        }
    }
    html +=
    `
    <span class="custom-check">
        <input type="checkbox" ${checked} class="${checkboxName}"
name="${checkboxName}" id="${checkboxName}${row["id"]}" value="${row["id"]}">
        <label class="text-dark"
for="${checkboxName}${row["id"]}">${row["title"]}</label>
        <span class="custom-check-target"></span>
    </span>
    `
    //     <div>
```

```
    //     <input type="checkbox" class="boroughFilter" name="borough" id="The
Bronx" value="The Bronx">
    //     <label for="The Bronx">Apartment</label>
    //   </div>
    }
    checkboxHolder.innerHTML = html;
}


function displayListings(data, args){
    console.log(data);


    let listingHolder = args.listingHolder;


    let html = "";


    listingHolder.innerHTML = "";


    if(data.length < 1){
        let errorText = "No listings found for filters provided";
        if(args.noListingsMessage){
            errorText = args.noListingsMessage;
        }
        listingHolder.innerHTML = `<p class="h3">${errorText}</p>`;
        return;
    }


    for(let row of data){
        let body = row["body"];
        let img = row["img"];
        let rooms = row["rooms"];


        let favorite = body["favorite"] > 0;


    // <!--Start of custom listing-->
    // <div class="listing">
    //   <div class="listing-main">
    //     <div class="listing-img w-100 listing-img-height">
```

```
    //       <img src="../resources/imgs/16821706612066674552644 3e325515b3.jpg"
alt="" class="img-fluid mk-img-fluid">
    //     </div>
    //     <div class="listing-body">
    //       <h3 class="h5 listing-title">New listing</h3>
    //       <p class="listing-desc">Lorem, ipsum dolor sit amet consectetur
adipisicing elit. Mollitia ex architecto quaerat officia assumenda, veritatis
dolorem tempora eos nostrum libero?</p>
    //       <p class="listing-size">Size: 3000 feet</p>
    //       <p class="listing-price">Price: 3000$</p>
    //     </div>
    //   </div>
    //   <div class="listing-cover">
    //     <p class="h5 text-center">Additional information</p>
    //     <ul class="list-group list-group-flush w-75">
    //       <li class="list-group-item">Bedrooms: 5</li>
    //       <li class="list-group-item">Livingrooms: 3</li>
    //       <li class="list-group-item">Bathrooms: 2</li>
    //     </ul>
    //   </div>
    //   <div class="listing-footer w-100">
    //     <a href="#" class="card-link  w-100 listing-read-more text-
center">Read more</a>
    //   </div>
    // </div>
    // <!-- End of custom listing-->


        html +=
        `
        <div class="listing">
        <a href="#" data-id="${body["id"]}" class="mk-favorite-icon-holder">
            <span class="iconify mk-favorite-icon" data-icon="${favorite ?
"mdi:cards-heart": "mdi:cards-heart-outline"}">Heart</span>
        </a>
        <div class="listing-main">
          <div class="listing-img w-100 listing-img-height">
            <img src="../resources/imgs/${img}" alt="${body["listing_name"]}
img" class="img-fluid mk-img-fluid">
          </div>
          <div class="listing-body">
            <h3 class="h5 listing-title">${body["listing_name"]}</h3>
            <p class="listing-desc">${body["description"]}</p>
            <p class="listing-size">Size: ${body["size"]} feet</p>
            <p class="listing-price">Price: ${body["price"]}$</p>
```

```
          </div>
        </div>
        <div class="listing-cover">
        <p class="h5 text-center">Additional information</p>
        <ul class="list-group list-flush w-75">
          <li class="list-group-item">Building type: ${body["Type"]}</li>
          <li class="list-group-item">Borough: ${body["borough"]}</li>
        `
        for(let room of rooms){
            html += `<li class="list-group-item">${room["room_name"]}:
${room["numberOf"]}</li>`
        }
        html +=
        `
        </ul>
        </div>
        <div class="listing-footer w-100">
        <a href="listing.html?listing_id=${body["id"]}" class="card-link  w-100
listing-read-more text-center">Read more</a>
        </div>
    </div>
        `
    }


    listingHolder.innerHTML = html;


    addFavoriteFunctionality();
}


function flipListingFavoriteIcon(args){
    let favoriteButtons = document.querySelectorAll(".mk-favorite-icon-
holder");
    let listingId = args.idOfListing;
    for(let button of favoriteButtons){
        if(button.dataset.id == listingId){
            let iconHolder = button.firstElementChild;
            iconHolder.dataset.icon = iconHolder.dataset.icon === "mdi:cards-
heart" ? "mdi:cards-heart-outline" : "mdi:cards-heart";
            console.log("Here");
        }
    }
}
```

```
function getQuestionsForUser(){
    readAjax("getQuestionsForUser", displaySurveyQuestions);
}


function setGlobal(data, args){
    globalData[args.name] = data;
}


function displaySurveyQuestions(data){
    // <!--One form-->
    // <form id="surveyQuestion1" class="my-2 question-form">
    //     <div class="mb-3">
    //         <p class="h2">Question: How do you like this website?</p>
    //     </div>
    //     <div class="mb-3">
    //         <div class="form-check">
    //             <input class="form-check-input" type="radio"
name="flexRadioDefault" id="flexRadioDefault1">
    //             <label class="form-check-label" for="flexRadioDefault1">
    //               Default radio
    //             </label>
    //         </div>
    //           <div class="form-check">
    //             <input class="form-check-input" type="radio"
name="flexRadioDefault" id="flexRadioDefault2" checked>
    //             <label class="form-check-label" for="flexRadioDefault2">
    //               Default checked radio
    //             </label>
    //         </div>
    //     </div>
    //     <button type="submit" id="sendMessage" name="sendMessage" class="btn
btn-primary">Submit answer</button>
    //   </form>
    // <!--End of one form-->
    let html = ``;
    let surveyHolder = document.querySelector("#surveyHolder");
    surveyHolder.innerHTML = "";
    let question;
    let answers;
```

```
    if(data.length < 1){
        html += `<p class="h2 text-center">No more questions left</p>`;
        surveyHolder.innerHTML = html;
        return;
    }


    for(let row of data){
        question = row["question"];
        answers = row["answers"];


        html+= `
        <form id="surveyQuestion${question["id"]}" class="my-2 question-form
bg-dark">
        <div class="mb-3">
            <p class="h2">Question: ${question["question"]}?</p>
        </div>
        <div class="mb-3">
        `
        for(let answer of answers){
        html +=
        `
        <div class="form-check">
            <input class="form-check-input" type="radio"
value="${answer["answer_id"]}" required="required"
name="questionAnswers${question["id"]}" id="answer${answer["answer_id"]}">
            <label class="form-check-label" for="answer${answer["answer_id"]}">
                ${answer["answer"]}
            </label>
        </div>
        `
        }
        html += `
        </div>
        <button type="submit" data-id="${question["id"]}"
id="submitForm${question["id"]}" name="submitForm${question["id"]}" class="btn
btn-primary submit">Submit answer</button>
        <span class="error-msg hidden"></span>
        </form>`
    }
    surveyHolder.innerHTML = html;
    let submitButtons = document.querySelectorAll(".submit");
    for(let button of submitButtons){
        addEventListenerOnce("click", button, function(e){
```

```
            e.preventDefault();
            let questionId = this.dataset.id;
            submitQuestionAnswer(questionId);
        })
    }
}


function submitQuestionAnswer(questionId){
    let answerCheckboxes =
document.getElementsByName(`questionAnswers${questionId}`);
    let selectedAnswerId = 0;


    for(let checkbox of answerCheckboxes){
        if(checkbox.checked){
            selectedAnswerId = checkbox.value;
            break;
        }
    }
    let submitButton = document.querySelector(`#submitForm${questionId}`);
    if(selectedAnswerId === 0){
        addError(submitButton, "You must select an answer before submitting",
1);
        return;
    }
    else{
        removeError(submitButton, 1);
    }
    data = {};
    data.answerId = selectedAnswerId;
    args = {};
    args.additionalFunctions = new Array(getQuestionsForUser);
    submitAjax("submitAnswer", showResult, data, args);
}


function showSuccess(data){
    let messageHolder = document.querySelector("#error-holder");
    let displayMessage = true;
    if(displayMessage){
        let message = document.createElement("span");
        message.classList.add("error-message");
        message.classList.add("alert");
        message.classList.add("alert-success");
```

```
        message.innerText = data;
        messageHolder.append(message);
        setTimeout(function(){
            hideElement(message);
        }, 2000);
    }
}


function showResult(data, args){
    let messageHolder = document.querySelector("#error-holder");
    let displayMessage = true;
    if(displayMessage){
        let message = document.createElement("span");
        message.classList.add("error-message");
        message.classList.add("alert");
        message.classList.add("alert-success");
        message.innerText = data;
        messageHolder.append(message);
        setTimeout(function(){
            hideElement(message);
        }, 2000);
    }
    if(args.closeModal){
        closeCurrentModal();
    };
}



function openModal(modal, modalBackground) {
    showElement(modalBackground, "hidden");
    showElement(modal, "hidden")
}
function closeModal(modal, modalBackground) {
    hideElement(modalBackground, "hidden");
    hideElement(modal, "hidden");
}


function closeCurrentModal(){
    hideElement(globalData.modalBackground, "hidden");
    hideElement(globalData.currModal, "hidden");
}
```

```
function addEventListenerOnce(event, element, onEvent, listenerMark = ""){
    let listenerMarker = event + listenerMark;


    if(element.classList.contains(listenerMarker)){
        return;
    }


    element.classList.add(listenerMarker);
    element.addEventListener(event, onEvent);
}


function fillDropdown(data, args){
    let selectElement = args[0];
    let simple = args[1];
    let value;
    let title;
    let newElement;
    let identifier = selectElement.id;
    let children = document.querySelectorAll(`.${identifier}`);
    for(let child of children){
        child.remove();
    }
    for(let row of data){
        if(simple){
            value = row;
            title = row;
        }
        else{
            value = row.id;
            title = row.title;
            if(row.Count){
                title += ` (${row.Count})`;
            }
        }
        let firstLetter = title.charAt(0);


        let firstLetterCap = firstLetter.toUpperCase();
```

```
        let remainingLetters = title.slice(1);


        title = firstLetterCap + remainingLetters;


        newElement = document.createElement("option");


        newElement.innerText = title;


        newElement.setAttribute("value", value);


        newElement.classList.add(identifier);


        selectElement.appendChild(newElement);
    }
}



function generateNavbar(response){
    let url;


    data = response.links;
    accessLevel = response.accessLevel;


    let headerHolder = document.querySelector("#headerHolder");
    let footerHeaderHolder = document.querySelector("#footerHeaderHolder");
    let navbarHolder = document.querySelector("#navbarHolder");
    let footerHolder = document.querySelector("#footerHolder");


    let headerElement = data.filter(el => el.location == "head")[0];
    let navbarElements = data.filter(el => el.location == "navbar");
    let footerElements = data.filter(el => el.location == "footer");


    url = generateUrl(headerElement, "pages/");
```

```
    headerHolder.href = url;
    headerHolder.text = headerElement.link_title;


    footerHeaderHolder.href = url;
    footerHeaderHolder.text = headerElement.link_title;


    for(let navbarElement of navbarElements){
        navbarHolder.innerHTML += generateLinkElement(navbarElement, "pages/");
    }


    if(accessLevel > 1){
        navbarHolder.innerHTML +=
        `
          <li class="nav-item">
            <a class="nav-link" id="logoutButton" aria-current="page"
href="#">Log out</a>
          </li>
        `;
        let logoutButton = document.querySelector("#logoutButton");
        logoutButton.addEventListener("click", function(e){
            e.preventDefault();
            readAjax("logout", redirectSuccess, { newLocation : "login.html",
landing : false});
        })
    }


    for(let footerElement of footerElements){
        footerHolder.innerHTML += generateLinkElement(footerElement);
    }
}


function callMultipleFunctions(data, functions){
    let firstFunction = functions[0];
    firstFunction(data);
    for(let i = 1; i < functions.length; i++){
        let currFunc = functions[i];
        currFunc();
    }
```

```
}


function toggleShowElement(element){
    let hidden = element.classList.contains("hidden");
    if(hidden){
        showElement(element);
    }
    else{
        hideElement(element);
    }
}


function showElement(element, type="hide"){
    element.classList.remove(type);
}


function hideElement(element, type="hide"){
    element.classList.add(type);
}


function errorHandler(error){
    let errorHolder = document.querySelector("#error-holder");
    let errorMessage = document.createElement("span");
    errorMessage.classList.add("error-message");
    errorMessage.classList.add("alert");
    errorMessage.classList.add("alert-danger");
    errorMessage.innerText = error;
    errorHolder.append(errorMessage);
    // showElement(errorHolder);
    setTimeout(function(){
        hideElement(errorMessage);
    }, 2000);
}


function createRequest(){
    let request = false;
    try{
        request = new XMLHttpRequest();
    }
    catch{
```

```
        try{
            request = new ActiveXObject("Msxml2.XMLHTTP");
        }
        catch{
            try{
                request = new ActiveXObject("Microsoft.XMLHTTP");
            }
            catch{
                console.log("Ajax is not supported");
            }
        }
    }
    return request;
}


function testGeneric(field, statement, errorMessage, errorHolderDistance = 1){
    if(statement){
        removeSuccess(field);
        addError(field, errorMessage, errorHolderDistance);
        return 1;
    }
    removeError(field, errorHolderDistance);
    addSuccess(field);
    return 0;
}


function testNumericBounds(field, minimalValue, errorMessage,
errorHolderDistance = 1){
    let value = field.value;
    if(value < minimalValue){
        removeSuccess(field);
        addError(field, errorMessage, errorHolderDistance);
        return 1;
    }
    removeError(field, errorHolderDistance);
    addSuccess(field);
    return 0;
}


function testDropdown(field, negativeValue, errorMessage, errorHolderDistance =
1){
    let value = field.value;
```

```
    //On success
    if(value != negativeValue){
        removeError(field, errorHolderDistance);
        addSuccess(field);
        return 0;
    }
    //On fail
    else{
        removeSuccess(field);
        addError(field, errorMessage, errorHolderDistance);
        return 1;
    }
}


function testImage(field, errorMessage, errorHolderDistance = 1){
    let value = field.value;
    if(value == ""){
        removeSuccess(field);
        addError(field, errorMessage, errorHolderDistance);
        return 1;
    }
    removeError(field, errorHolderDistance);
    addSuccess(field);
}


function reTestText(regex, field, errorMessage, errorHolderDistance = 1){
    let textValue = field.value;
    let passes = regex.test(textValue);
    if(passes){
        if(errorMessage != ""){
            removeError(field, errorHolderDistance);
            addSuccess(field);
        }
        return 0;
    }
    else{
        if(errorMessage != ""){
            removeSuccess(field);
            addError(field, errorMessage, errorHolderDistance);
        }
        return 1;
    }
}
```

```
function getNthNextElement(field, n){
    let elem = field;
    for(let i = 0; i < n; i++){
        elem = elem.nextElementSibling;
    }
    return elem;
}


function addSuccess(field){
    field.classList.add("success-outline");
}


function removeError(field, errorHolderDistance = 0 ){
    if(errorHolderDistance > 0){
        let errorBox = getNthNextElement(field, errorHolderDistance);
        errorBox.innerText = "";
        errorBox.classList.add("hidden");
    }
    field.classList.remove("error-outline");
}


function removeSuccess(field){
    field.classList.remove("success-outline");
}


function addError(field, msg, errorHolderDistance){
    if(msg != ""){
    let errorBox = getNthNextElement(field, errorHolderDistance);
        console.log(errorBox);
        errorBox.innerText = msg;
        errorBox.classList.remove("hidden");
    }
    field.classList.add("error-outline");
}


function redirect(args){
    let newLocation = args.newLocation;
    let landing = args.landing;
```

```
    let additionalText = ""
    if(window.location.hostname === "localhost"){
        additionalText = "/nycestatee";
    }


    let newLink = window.location.hostname + additionalText + (landing ?
`/${newLocation}` : `/pages/${newLocation}`);
    window.location.assign("https://" + newLink);
}


//Wrapper for the redirect function
function redirectSuccess(data, args){
    redirect(args);
}


function readAjax(url, resultFunction, args = {}){
    let request = createRequest();
    request.onreadystatechange = function(){
        if(request.readyState == 4){
            handleServerResponse(resultFunction, args, request);
        }
    }
    request.open("GET", ajaxPath+url+".php");
    request.send();
}


function submitAjax(url, resultFunction, data, args = {}){
    let request = createRequest();
    request.onreadystatechange = function(){
        if(request.readyState == 4){
            handleServerResponse(resultFunction, args, request);
        }
    }


    request.open("POST", ajaxPath+url+".php");
    request.setRequestHeader("Content-type", "application/json");
    console.log(data);
    request.send(JSON.stringify(data));
}
```

```
function submitFormDataAjax(url, resultFunction, data, args = {}){
    let request = createRequest();
    request.onreadystatechange = function(){
        if(request.readyState == 4){
            handleServerResponse(resultFunction, args, request);
        }
    }
    request.open("POST", ajaxPath+url+".php");
    console.log(data);
    request.send(data);
}


function handleServerResponse(resultFunction, args, request){
    if(request.status >= 200 && request.status < 300){
        console.log(request.responseText);
        let data = JSON.parse(request.responseText);
        if(args != {}){
            resultFunction(data.general, args);
            if(args.additionalFunctions){
                let additionalArgs = args.additionalFunctionArgs ? true :
false;
                for(let func of args.additionalFunctions){
                    if(additionalArgs){
                        func(args.additionalFunctionArgs);
                        continue;
                    }
                    func();
                }
            }
        }
        else{
            resultFunction(data.general);
        }
    }
    else if(request.status >= 300 && request.status < 400){
        redirect(args);
    }
    else if(args.redirectOnNotAllowed && (request.status === 401 ||
request.status === 403)){
        redirect(args);
    }
    else{
```

```
        let data = JSON.parse(request.responseText);
        if(args.errorFunction){
            args.errorFunction(data["error"]);
            return;
        }
        errorHandler(data["error"]);
        console.log(data["error"]);
    }
}



function generateUrl(object, redirect = ""){
    let url = "";
    if(object.landing == 1){
       url += mainPage? '' : '../';
    }
    else{
       url += mainPage? `${redirect}` : '';
    }
    url += object.href;
    return url;
 }


 function generateLinkElement(object, redirect = ""){
    let html;
    let url = generateUrl(object, redirect);
    let text = object.link_title;
    let icon = object.icon;
    if(icon == null){
        html =
        `
        <li class="nav-item">
          <a class="nav-link ${currentPage == object.href ? "active" : ""}"
id=${text} aria-current="page" href="${url}">${text}</a>
        </li>
      `
    }
    else{
        html = `
        <li class="col-md-3 col-6 icon-holder">
            <a class="" href="${url}">
                <span class="iconify" id="${text}-icon" data-
icon="${icon}"></span>
```

```
            </a>
        </li>
        `
    }
    return html;
}
```