Task 1

Dobra praktyką w przypadku zmian w motywach jest stworzenie motywu potomnego.

W związku z tym pliki css przechowywałbym w folderze motywu potomnego np:

/wp-content/themes/twentytwentyone-child/assets/css/custom-styles.css

plik ten importowany byłby następnie w głównym pliku ze stylami motywu potomnego:

/wp-content/themes/twentytwentyone-child/style.css

Task 2

Można to wykonać za pomocą następującej funkcji umieszczonej w pliku functions.php:

```
add_action( 'wp_footer', function() {
   wp_enqueue_script( 'my-script', get_theme_file_uri( '/assets/js/scripts.js' ), array( 'jquery' ), null, true );
});
```

Task 3

Poniżej znajduje się wymagany w tym zadaniu kod:

```
function register_cpt_library() {
	$labels = [
		"name" => __( "Books", "twentytwentyone" ),
		"singular_name" => __( "Book", "twentytwentyone" ),
	];
	$args = [
		"label" => __( "Books", "twentytwentyone" ),
		"labels" => $labels,
		"description" => "",
		"public" => true,
		"publicly_queryable" => true,
		"show_ui" => true,
```

```
                "show_in_rest" => true,
                "rest_base" => "",
                "rest_controller_class" => "WP_REST_Posts_Controller",
                "has_archive" => false,
                "show_in_menu" => true,
                "show_in_nav_menus" => true,
                "delete_with_user" => false,
                "exclude_from_search" => false,
                "capability_type" => "post",
                "map_meta_cap" => true,
                "hierarchical" => false,
                "rewrite" => [ "slug" => "library", "with_front" => true ],
                "query_var" => true,
                "supports" => [ "title", "editor", "thumbnail" ],
                "taxonomies" => [ "book-genre" ],
                "show_in_graphql" => false,
        ];
        register_post_type( "library", $args );
}
add_action( 'init', 'register_cpt_library' );

function register_cpt_taxes_book_genre() {
        $labels = [
                "name" => __( "Genres", "twentytwentyone" ),
                "singular_name" => __( "Genre", "twentytwentyone" ),
        ];
        $args = [
                "label" => __( "Genres", "twentytwentyone" ),
                "labels" => $labels,
                "public" => true,
                "publicly_queryable" => true,
```

```
                "hierarchical" => false,
                "show_ui" => true,
                "show_in_menu" => true,
                "show_in_nav_menus" => true,
                "query_var" => true,
                "rewrite" => [ 'slug' => 'book-genre', 'with_front' => true, ],
                "show_admin_column" => false,
                "show_in_rest" => true,
                "rest_base" => "book-genre",
                "rest_controller_class" => "WP_REST_Terms_Controller",
                "show_in_quick_edit" => false,
                "show_in_graphql" => false,
        ];
        register_taxonomy( "book-genre", [ "library" ], $args );
}
add_action( 'init', 'register_cpt_taxes_book_genre' );
```

Task 4.1

Utworzony został customowy template w pliku o nazwie: single-library.php

Aby dostać się do wskazanego widoku należy wejść z adresu: http://localhost/library/book-1/

Poniżej znajduje się kod szablonu:

```
<?php get_header(); ?>

<div id="main-content" class="main-content">
   <div id="primary" class="content-area">
      <div id="content" class="site-content" role="main">

         <?php
            echo the_title() .'<br>';
            echo get_the_post_thumbnail().'<br>';
            $terms = get_the_terms( $post->ID, 'book-genre' );
            foreach($terms as $term) {
               echo $term->name.'<br>';
            }
            echo get_the_date('Y-m-d').'<br>';
         ?>

      </div><!-- #content -->
   </div><!-- #primary -->
</div><!-- #main-content -->

<? php get_footer(); ?>
```

Task 4.2

Utworzony został customowy template w pliku o nazwie: taxonomy-book-genre.php

Aby dostać się do wskazanego widoku należy wejść z adresu: http://localhost/book-genre/genre-1/

Poniżej znajduje się kod szablonu archiwum:

```
<?php
   get_header();
   $term = get_queried_object();

   $paged = ( get_query_var( 'paged' ) ) ? get_query_var( 'paged' ) : 1;
   $args = array(
      'post_type' => 'library',
      'book-genre' => $term->slug,
      'paged' => $paged
   );
   $query = new WP_Query( $args );
?>

<div id="main-content" class="main-content">
   <div id="primary" class="content-area">
      <div id="content" class="site-content" role="main">
         <header class="archive-header">
            <h1 class="archive-title">
               <?php echo $term->name; ?>
            </h1>
         </header><!-- .archive-header -->
         <?php
            if ($query->have_posts()) {
               while ( $query->have_posts() ) : $query->the_post();
                  the_title( sprintf( '<h2 class="entry-title default-max-width"><a href="%s">', esc_url(
get_permalink() ) ), '</a></h2>' );
```

```
                twenty_twenty_one_post_thumbnail();
            endwhile;
        }
        $big = 999999999;
        echo paginate_links( array(
            'base' => str_replace( $big, '%#%', get_pagenum_link( $big ) ),
            'format' => '?paged=%#%',
            'current' => max( 1, get_query_var('paged') ),
            'total' => $query->max_num_pages,
            'prev_text' => '&laquo;',
            'next_text' => '&raquo;'
        ) );
        wp_reset_postdata();
      ?>
    </div><!-- #content -->
  </div><!-- #primary -->
</div><!-- #main-content -->
<?php get_footer(); ?>
```

Uwaga: Aby paginacja zadziałała należy w pliku function.php dodać następujący kod:

```
function custom_tax_query_change( $query ) {
    if ( ! is_admin() && $query->is_tax( 'book-genre' ) ) {
        $query->set( 'posts_per_page', 5 );
    }
}
add_action( 'pre_get_posts', 'custom_tax_query_change' );
```

Task 5.1

Poniższy kod należy dodać do pliku: functions.php

```
function recent_book_shortcode() {
	global $post;

	$args = array(
		'posts_per_page' => '1',
		'order'		=> 'DESC',
		'orderby'	=> 'post_date',
		'post_type'	=> 'library'
	);

	$output = '';

	$posts = get_posts($args);

	foreach($posts as $post) {
		setup_postdata($post);
		$output .= '<div>'. get_the_title() .'</div>';
	}

	wp_reset_postdata();

	return $output;

}
add_shortcode('recent_book', 'recent_book_shortcode');
```

Wywołanie shortcodu na stronie lub w poście: **[recent_book]**

Task 5.2

Poniższy kod należy dodać do pliku: functions.php

```
function books_by_genre_shortcode($atts) {
   global $post;
   extract(shortcode_atts(array(
      'genre' => '',
   ), $atts));
   $output = '';
   $args = array(
      'posts_per_page' => '5',
      'post_type'      => 'library',
      'tax_query'      => array(
         array(
            'taxonomy'  => 'book-genre',
            'field'     => 'term_id',
            'terms'     => array( $genre )
         )
   ));
   $books = NEW WP_Query($args);
   while($books->have_posts()) {
      $books->the_post();
      $output .= '<div>'.get_the_title().'</div>';
   };
   wp_reset_query();
   return $output;
}
add_shortcode('books_by_genre', 'books_by_genre_shortcode');
```

Wywołanie shortcodu na stronie lub w poście: **[books_by_genre genre="2"]**

W tym przypadku zostaną wyświetlone tytuły książek z taxonomią o term_id = 2

Task 6 - bonus

Aby zwrócić 20 książek (wymagane pola: name, date, genre, excerpt) przy pomocy AJAX należy umieścić poniższy kod w pliku functions.php

```
function get_ajax_posts() {
   global $post;
   $paged = ( get_query_var( 'paged' ) ) ? get_query_var( 'paged' ) : 1;
   $ajaxposts = [];
   $counter = 0;
   $args = array(
      'post_type' => array('library'),
      'post_status' => array('publish'),
      'posts_per_page' => 20,
      'paged' => $paged,
      'order' => 'DESC',
      'orderby' => 'date',
   );
   $posts = get_posts( $args );
   foreach($posts as $post) {
      $name = $post->post_name;
      $date = get_the_date();
      $excerpt = get_the_excerpt();
      $genre = '';
      $terms = get_the_terms( $post->ID, 'book-genre' );
         foreach($terms as $term) {
            $genre = $term->name;
         }
      $obj = [
         'name'     => $name,
         'date'     => $date,
         'excerpt'   => $excerpt,
```

```
            'genre'     => $genre
        ];
        $ajaxposts[$counter] = $obj;
        $counter++;
    }
    echo json_encode($ajaxposts);
    exit;
}
add_action('wp_ajax_get_ajax_posts', 'get_ajax_posts');
add_action('wp_ajax_nopriv_get_ajax_posts', 'get_ajax_posts');
```

Dodatkowo w pliku /assets/js/scripts należy dodać następującą funkcję zwracającą potrzebne dane przy pomocy AJAX:

```
jQuery(document).ready(function($) {
    $.ajax({
        type: 'POST',
        url: '/wp-admin/admin-ajax.php',
        dataType: "json",
        data: { action : 'get_ajax_posts' },
        success: function( response ) {
            $.each( response, function( key, value ) {
                console.log( key, value );
            } );
        }
    });
});
```