LOMBA KOMPETENSI SISWA
SEKOLAH MENENGAH KEJURUAN
TINGKAT PROVINSI JAWA BARAT
TAHUN 2019

NASKAH SOAL
(Terbuka)

Bidang Lomba
WEB TECHNOLOGIES

SMK Bisa !
Siap Kerja ~ Cerdas ~ Kompetitif

SPEED TEST MODULE

INTRODUCTION

In this module, you are given 4 hours to complete mini tasks. There are 5 parts of mini tasks, they are design, layout, front-end development, back-end development and content management system.

PART A: WEBSITE DESIGN

A1: Blur Picture Image (Easy)

Given a picture image, please use graphical software to make this on greyscale, put a blur effect, and save as a1.jpg.

A2: Transparent Background (Easy)

Given an image on format jpg. Please use graphical software to turn the background into transparent, and save as a2.png.

A3: Pattern (Easy)

Make a background image with any line pattern that repeat every 100x100 pixels, and save with 1920x1080 px as result.jpg.

A4: Loading GIF (Normal)

Create a GIF animation with a word "Loading" that appears on the left edge of image and moves outside of the right edge of image. The animation should be within 3 seconds to complete.

A5: Walking Gif (Normal)

Create a GIF animation from the given walk.gif file, so that the character moves in from the left edge of image and moves right and moves out of the right edge of screen.

A7: Messenger (Difficult)

Draw 2 screen mockups for a messenger app with size 360x640.

1 screen - a list of dialogs (avatar, full name, message preview).

2 screen - chat with the user (avatar, full name, menu (three points), messages, text field, send button)

All elements should exist.

PART B: LAYOUT

B1: HTML tags (Easy)

A customer wants you to make his writings into HTML so that it displays well in web browser. He has written the text. Your job is to use HTML to define the content for web browser to properly display it. Please consider semantic tags and accessibility.

B2: CSS Grid (Normal)

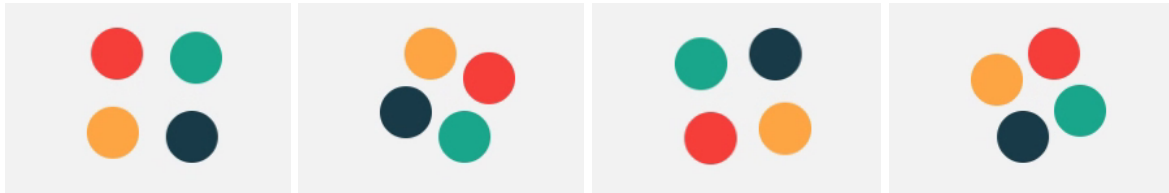Please implement the following layout in CSS grid.
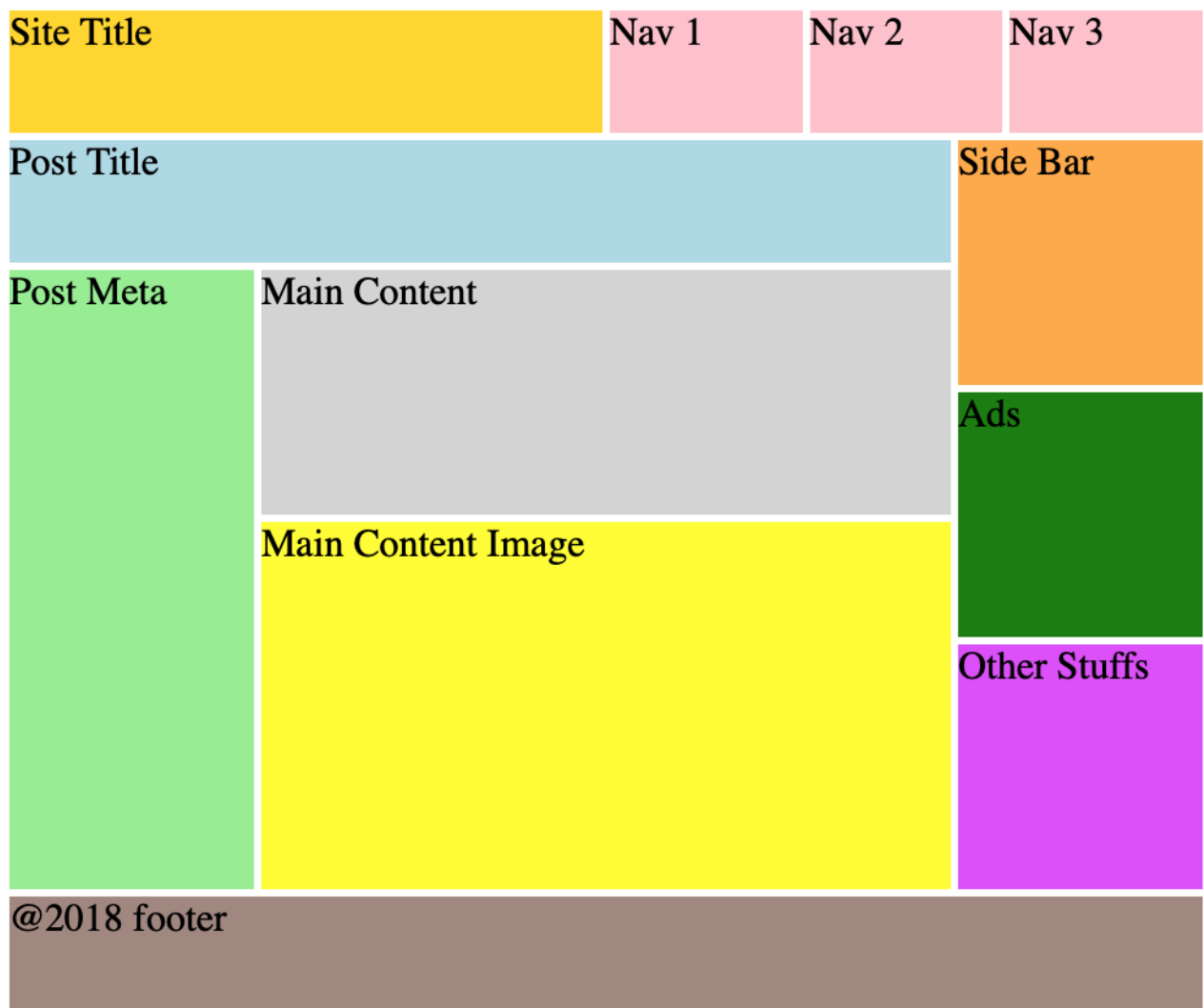
B3 Loading Screen (Normal)

Create an animated loader using only HTML and CSS same as video.mp4. The loader animation should be looped. 4 different colors (#19A68C, #F63D3A, #FDA543, #193B48) should be used.

Screen shots:

B5 Yellow Square (Easy)

Make an animation that starts with a big yellow square, and this square turns a small red circle, the animation needs to be centered on the view and the animation needs to loop;

B8 Jumper Ball (Easy)

- Make a loop animation of a jumper ball respecting all states from a animation (only css);

B9: HTML/CSS – Contact Form (Normal)

An HTML and CSS implementation of a not yet finished form are given.

Modify the CSS and HTML so that the form looks as depicted in the screenshot in the media files.
It's responsive behaviour should follow the screencast in the media files.

The use of JavaScript is NOT allowed.

B13: Gradient animation (Easy)

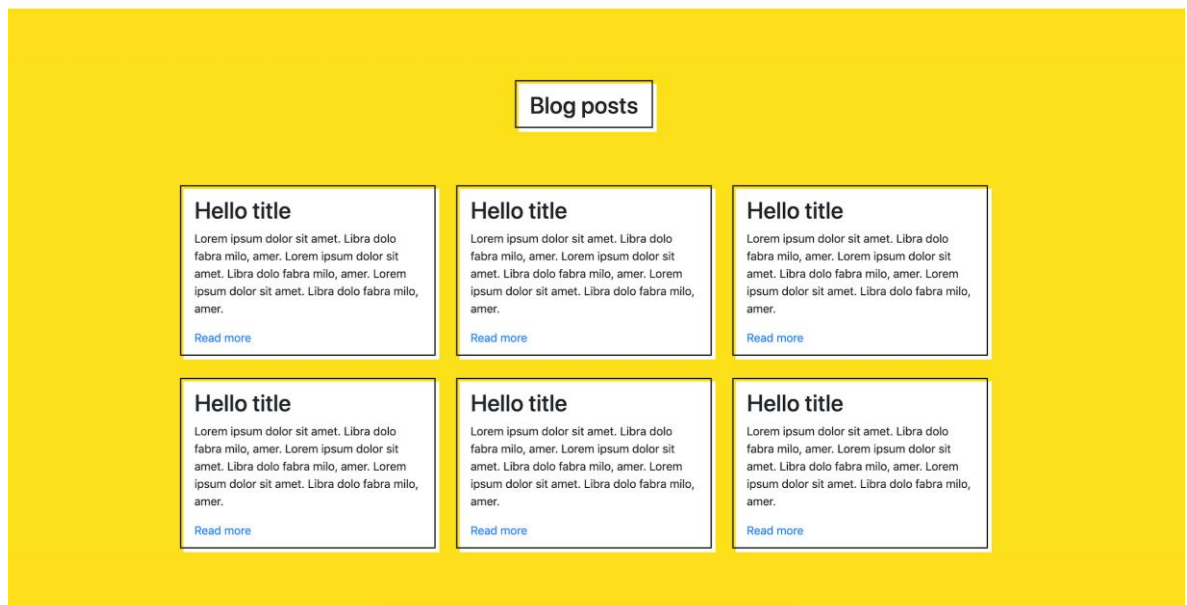By using HTML and CSS, create a gradient shift animation on the page. The gradient may move into any direction.

B14: Modern layout (Easy)

Your task is to create the following layout, the posts' width is ⅓ of the container's on desktop screen size (>=1440px) and full width below this.

PART C: FRONT-END DEVELOPMENT

C2: Loading Animation (Normal)

We want to create the following CSS loading animation.
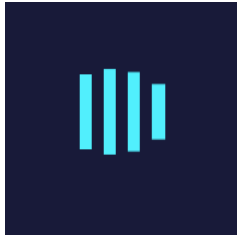
C3: Drawing Canvas (Normal)

Please try to use canvas to build a drawing pad with the following features:

- Basic drawing with mouse down and move
- 3 color switching

C5: JS – Calculation (Easy)

Implement the function that calculate the sum of the squares of the odd numbers in a given array.

C10: Simple Image Filter (Difficult)

For this speed Project you must create a function that allows you to add a filter in a certain image using canvas. This speed Project is divided into two parts.
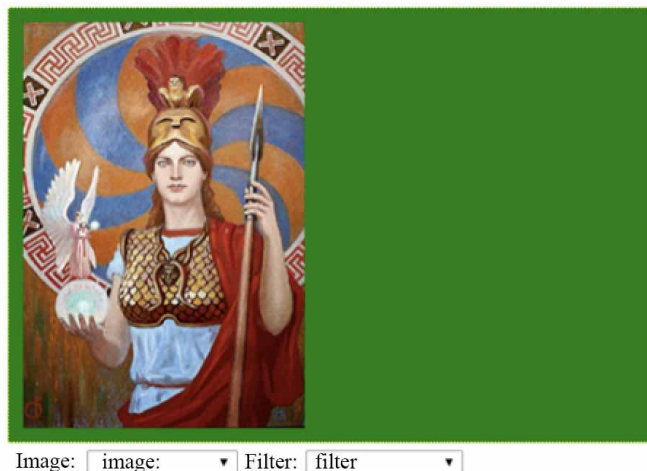
Part 1
● Develop the following design:

● The size of the canvas should be 460 pixels wide and 320 pixels high.
● Create a drop-down list, with the image options: Athena.jpg, theKiss.jpg, young-pearl.jpg and mona-lisa.jpg. There must be a first option called image.
● Load the images on the canvas depending on the option chosen in the drop-down list.
● The image of athena.jpg is loaded by default.

Part 2
● Create a drop-down list, called filter with the options: Darken and Lighten. There must be a first option called filter.
● When choosing a specific image in the dropdown list and applying a filter, the result should appear on the right side of the canvas according to the chosen option.

Image: image: ▼ Filter: filter ▼

C11: Canvas Animation (Easy)

Draw a circular figure in a container of 320px height and 400px width using canvas. When the browser window reloads, the circular object must move from left to right continuously.

PART D: BACK-END DEVELOPMENT

D3: PHP Array (Easy)

Given the code with some PHP arrays in media files, please implement a PHP function that compare the given two arrays and return a new array that contains the common elements from both given arrays.

D4: PHP Verification captcha (Difficult)

Please use PHP to produce verification code image (captcha), which must comply the following rules (please see the provided images):

a. It randomly generates four characters, consisting of English Alphabet (A-Z) or digit numbers (0-9).

b. The four characters must be slightly rotated.

c. The four characters must not be located at the same row.

d. The image contains at least 3 random lines.

e. At least 3 mixed noise points.

D5: Watermark (Easy)

Put a watermark (png file) on the image provided. The watermark should be in the lower right corner.

D11: A Pagination Code (Nromal)

After reading data.json, the data in the current page must be shown..

5 data must be shown in one page.

5 page links to move to another page must be shown in one page

Create buttons to move 5 pages at once, previous and next.

Page link of current page must be located in the center

Take a look at the provided screen shots and data.json.

You can only use HTML & CSS & PHP, but for reading json data and the pagination code must use only PHP. You can use bootstrap and JavaScript is not allowed.

Screen shots:

D13: Remove duplicates (Easy)

Given an array, remove the duplicate values, and show the new array;

PART E: CONTENT MANAGEMENT SYSTEMS

E1: Animated Social Button Plugin (Normal)

Create social buttons plugin for Wordpress using Fontawesome icons and add hover effects for the buttons.

When the plugin activated, social buttons must be shown in the footer area.

There should be 4 social buttons and the icon fonts are below.

   facebook - <i class="fab fa-facebook-f"></i>

   pinterest - <i class="fab fa-pinterest"></i>

   twitter - <i class="fab fa-twitter"></i>

   thumblr - <i class="fab fa-tumblr"></i>

The color of hover effects are below.

   facebook - #3d5998

   pinterest - #e50122

   twitter - #54acec

   thumblr - #314258

The effects should be same with thie example video but you can change the duration time of the hover effects.

You can only use HTML and CSS, no JS(including Libries and Framewors) are allowed.

Screen Shots(See the video.mp4 in folder named "2")
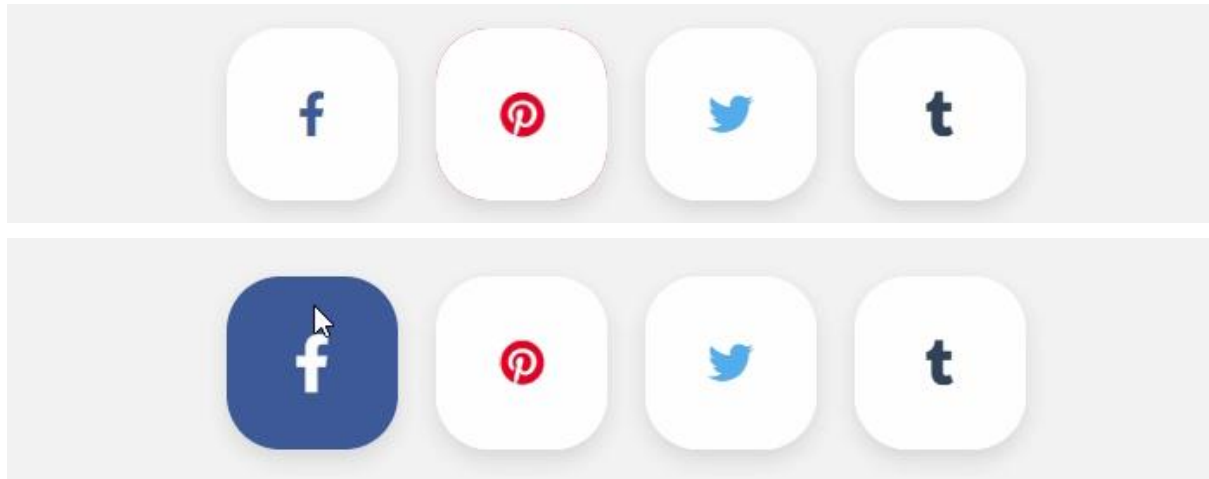
E2: Dropdown menu plugin (Difficult)

Create a Plugin that shows the menu. It must be able to be used in the theme through shortcode.

Admin can add/delete menus. When mouse is over one of the menu items, delete icon or button should be shown so that admin can delete the menu. The menus should not be static.

Menus shown in the CMS main page

DROPDOWN MENUS

Food   Sports

Pizza

Hamburger

Chicken
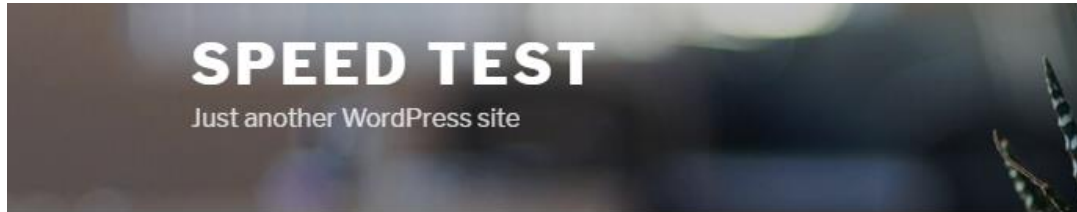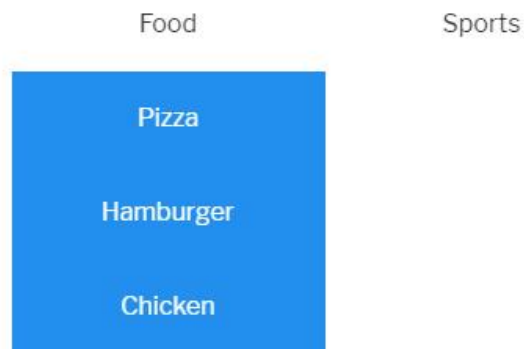
E3: CountDown Kazan WorldSkills plugin (Normal)

Create a plugin that tells the time left until 22d of August.

When the plugin is activated, widget is created.

If widget is added in sidebar, remaining time must be shown.

The date should not a static data.

E4: Plugin works (Easy)

- Make a plugin that when activated, insert a new menu item on admin, and when accessed, shows a Title: "Plugin works!";

E7: Widget (Easy)

- Make a widget that list of title of all pages created, with their permalinks;

INSTRUCTIONS TO THE COMPETITOR

You do not need to implement all the tasks. There will be instruction on C1 to clarify which tasks are selected to be implemented.

MARKING SCHEME

SECTION	CRITERION	JUDGEMENT MARKS	MEASUREMENT MARKS	TOTAL
A1	Design	0	4	4
A2	Layout	0	4	4
A3	Front-end	0	4	4
A4	Back-end	0	4	4
A5	CMS	0	4	4

CMS MODULE

INTRODUCTION

You are a freelancer working in the field of Web Technologies and you have been asked to develop the brand-new website for Kazan's most beautiful museums. Your client has heard that you are good at building content management system.

DESCRIPTION OF PROJECT AND TASKS

This module involves knowledge about website design, website layout techniques, client-side scripting, and server-side scripting, all combined in one CMS project. You will be using one of the most popular content management systems - WordPress.

Your job is to develop a website backed by content management system. Please create your own theme as a child-theme of a given starter WordPress theme. Please name your theme **Kazan_MuseumTour**.

The goal of this website is to present museums in Kazan, with some selected meseums that contain extra news information. It also gives a list of upcoming events. The goal of this website is to present an overview of all available museums in Kazan and to give a list of upcoming events, which could be of interest to all tourists visiting Kazan. The target audience for this page are tourists visiting Kazan. This can be a wide range, therefore you will define your interpretation of the target audience (based on "tourists visiting Kazan"). Also please make sure you have the title and slogan deinfed in the content management system.

The admin and editor need to have the possibility to add and modify every entry for each museum to either update existing information or add new museums to the list.

Content Management System

For administration tasks, we need two user profiles, Admin and Editor:

- The Admin user - access to the complete WordPress main dashboard.
    - Username: admin
    - Password: admin
- The Editor user – access as editor role in the CMS.
    - Username: editor
    - Password: editor

We want the CMS login page to be white-label. That means the login page should not show the CMS default logo. The background should be a fullscreen museum photo from one of the museums.

Also the login page should not include any "wp" wordings for white-label reasons. Please make the backend admin URL as following: <host>/admin/

Managing Museums and pages

The client wants to be able to add, edit, update, delete museums records. There are selected museums to have custom museum page.

Selected (highlighted) Museums

The client wants to highlight the following selected museums. This list is static. In the future, the client may want to add or remove museums from the selected list. But this is not in the current scope of work.

- Soviet Lifestyle Museum
- National Museum of the Republic of Tatarstan
- Museum of National Culture
- Chak-Chak Museum

For museum pages that are listed above, they have different layout than general pages. The client would like to have large full-page background with the meseum photo. This photo is selected by featured photo of the page.

Also, we need news posts from this specific museum to be displayed in the page.

All other museums

For museum pages that are not on the selected list, general page template is enough. There is no need to show museum specific news post in these museum pages. This general page should have a large photo banner as header. This photo is selected by featured photo of the page.

News Blog Post

From time to time, there will be news from the website as well as from specific museum. These blog posts should be categorized.

The client wants to show news posts from all categories in the home page. While in the specific museum page, the client wants to show news posts from only the category for the museum.

Post Categories
- Site Updates
- Seasonal Events
- Each selected museums

Pages and URL Strategy

Accessibility is very important to our website. We would like to lavarage different levels of accessibility optimization.

URL level of accessibility

All museums page should have its own URL permalink. For example, the page for museum of national culture may have URL: <host>/museum-of-national-culture/

News posts should have following URL structure:

- All news: <host>/news/
- News from a category: <host>/news/<category name>

Dashboard Configuration

The client wants to have At a Glance, Activity, Quick Draft in the dashboard. Please configure these dashboard widgets and remove all others in the dashboard.

Contact Form

The client does not have budget to build a contact form right now. We choose to use free static form to email service for the contact form. This is done by a static form with action pointing to the following URL. Please configure to send the email to **admin@example.com**.

POST https://formspree.io/email@domain.tld

The form should have the following input field: name, email, content.

You may find a form example code in the media file.

Menus

The menu should stick to the top when scrolling down the page. The menu has the following items.

- Home

- Museums (Drop down menu)

- Seasonal Events (List posts from categories "seasonal events" from all museums)

Front-end Requirement

You will need to create your own theme, based on the given starter themes. You will need to implement the following elements for your web page:

- A footer with copyrights and social media links
  - "**Copyright © 2019 - All rights reserved**" where the year should be the current year dynamically based on server time.
  - Footer plugins should be used for footer links to Twitter, Facebook and Instagram. User is able to change the social links.

Feel free to add and change as many elements as you like, but the elements in the list need to be present and your design should blend in with the given design.

The client does not like refreshing the whole page. Page transition should be loaded via JavaScript with animated transition effect. (This does not apply to admin area)

Home Page Layout Columns strategy and mockup

The layout should be responsive for different screens including mobile and desktop.

Home page in desktop screen should have a multiple columns layout. Left half is the news and right half is some kinds of cover and images.

Home page in mobile screen should have a top-down layout strategy. Each column take the full width of the screen, except for the list of selected museum. They should have a left and right 2 columns layout.

Further optimization tasks

The page needs to be optimized for search engines.

You can choose one of the provided plugins or implement your own SEO.

Remember to optimize certain parts of your website (e.g. urls, sitemap, …).

Secure your page by installing and configuring a security plugin.

Note: If the plugin you would like to choose is broken or does not work as expected, you need to choose another one or feel free to fix this plugin.

INSTRUCTIONS TO THE COMPETITOR

The page should follow the Web Content Accessibility Guide.

Publish the finished website on: http://localhost/XX_lksp2019_cms_module

XX is your competitor number.

EQUIPMENT, MACHINERY, INSTALLATIONS, AND MATERIALS REQUIRED

ITEM	DESCRIPTION
advanced-custom-fields.5.8.0.zip	Plugin
all-in-one-wp-security-and-firewall.zip	Plugin
autodescription.3.2.4.zip	Plugin
blankslate.2019.1.zip	Theme
contact-form-7.5.1.3.zip	Plugin
contact-form-plugin.4.1.5.zip	Plugin
contact-forms-builder.1.5.2.zip	Plugin
elegant-responsive-content-slider.1.0.1.zip	Plugin
jetpack-6.9.zip	Plugin
sample_underscores.zip	Theme
seo-by-rank-math.1.0.24.zip	Plugin
wonderplugin-carousel-free.zip	Plugin
wordfence.7.3.2.zip	Plugin
wordpress-seo.11.2.1.zip	Plugin
wp-responsive-jquery-slider.1.7.zip	Plugin

MARKING SCHEME

SECTION	CRITERION	JUDGEMENT MARKS	MEASUREMENT MARKS	TOTAL
B1	CMS Theme	0	3.0	3.0
B2	CMS Configuration	0	3.75	3.75
B3	CMS Design	1.5	1.5	3
B4	CMS Layout	3.0	3.5	6.5
B5	CMS Category	0.5	3.25	3.75
B6	CMS Admin	0.25	2.25	2.5

FRONTEND WEB TECHNOLOGIES

INTRODUCTION

Our world has changed a lot over the past 10 years, the borders of country are blurred, it becomes easier to travel, people learn new worlds and become travellers. But this leads to an increase in the information processed and may be the reason for the loss of the most important and interesting aspects of the life of the studied worlds for travellers.

This problem is solved by various electronic helpers, simple and affordable Internet services that help to systematize the data, follow the notes of other travelers, create new resources and databases about interesting places and things.

Your task will be to develop a working prototype of one of these services – Interactive Presenter. And a one-page marketing site for this presentation on the Internet.

The main idea of that service is to create a huge database of everything. And to give everyone the opportunity to get acquainted with it in an interactive way, allowing you to concentrate only on this interesting information.

But the company understands that this is a voluminous task, so it suggested that you develop a prototype of a web application for one narrow area - tourist company of Kazan, whose goal is to acquaint visitors of the city with the history, culture and atmosphere of the thousand-year-old capital of Tatarstan in an interactive presentation format. The visitor can choose the route of acquaintance and receipt of information in accordance with his contextual choice on each slide.

In the future, it is planned to open for everyone to create their own interactive routes and publish them on the company's website. Therefore, special attention is paid to the route editor in terms of convenience and ease of use for the mass user.

DESCRIPTION OF PROJECT AND TASKS

This task is to be completed in 4 hours. The result should work in modern browsers.

You will create the worked prototype of application using JavaScript ensuring the app works correctly in different web browsers, following the requirements described below.

The main feature of the application is its work in two modes: "Route Editor" for those who want to create and share their impressions of Kazan tourist information and "View Mode" those who study new tourist material. This feature should be reflected in the design of the website.

Because the service will be used as presentation tools to promote tourism in Kazan the users of the route editor can be segmented as adult aged 25 - 40 years old. But, for the presentation format it will be targeted for all ages, so that both children and adults can understand about history of Kazan easily.

Glossary

- Route Editor - a mode in the service that can be used to link different element using links visually and can be controlled using mouse and keyboard
- View Mode - a mode in the service that can be used to see the end result of all linked elements and their transition (like a presentation of slides)
- Element - a node in the route editor that is used to store the content and link it with other element (like a slide)
- Link - a line in the route editor that is used to store information about the next element to transition and the transition property itself
- Transition - an animation that runs in view mode between moving from one element into other element
- Content – it is user input data to be demonstrated.

*Front-end programming module*

Route editor
The editor should work on desktop. In this scope of work, we don't consider supporting touch devices for editor mode.

You need to make the route editor that can be used by the user to link several created elements into a sequence. The functionality of the route editor that you need to make is listed below:

1. When creating a new presentation or clearing the editor, one root element initially appears in the center of the screen. An element can be look like anything (circle, square, star, blob, etc.).
2. The element has 4 interaction areas, (1, 2, 3, 4), which are hidden by default and are displayed only after **hovering** on the element.

Original State   hover   Interactive Areas are displayed
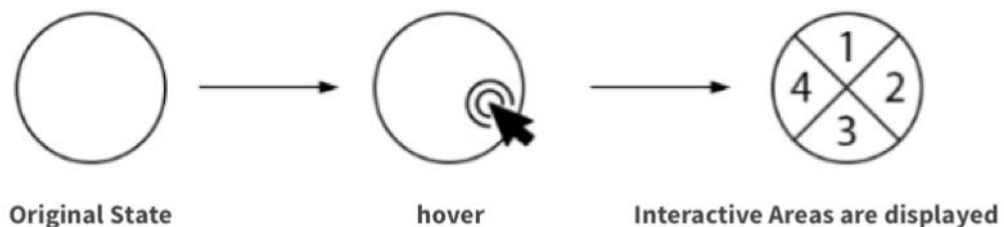
3. When you click on one of the areas, a related element should appear on the side of the areas node on which the click was made. Link between the elements should be created as well.
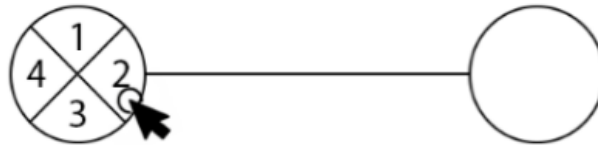
4. Alongside the 4-interactive-areas, there should be "edit" button and "delete" button for the element which are shown only when element is on hover.
5. The following features should be implemented for the elements:
    a. Editing the content of the element (slide). To edit you need to use one of the wysiwyg editors, which will be given to you.
    b. Editing captions for each of the sections (1,2,3,4), which will be shown in view mode.
    c. Deleting an element by clicking the delete button.
6. When deleting an element, all links of this element must also be deleted.
7. The user can delete a link by clicking on the link and pressing the **Delete/Back Space** key.
8. Elements can be moved around the editor by mouse drag. Links should also follow the element.
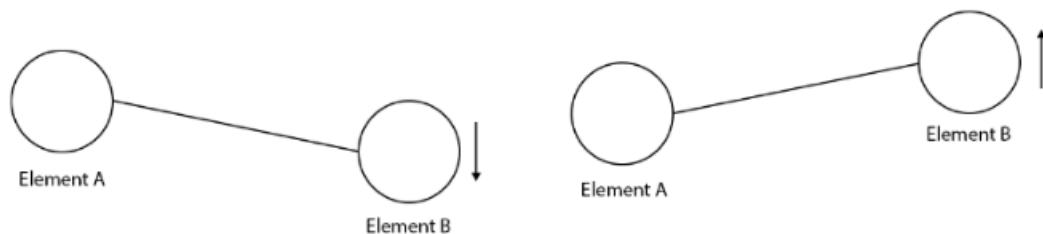
9. To link two unrelated elements (for example, after removing a connection), with the **Shift** key held down, move a section of one element to a section of another element.
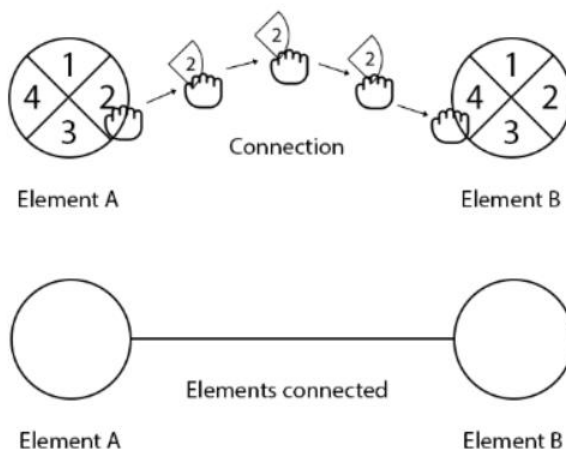
10. The possibility of looping routes is allowed.

LOMBA KOMPETENSI SISWA
SEKOLAH MENENGAH KEJURUAN
TINGKAT PROVINSI JAWA BARAT
TAHUN 2019

LKS SMK
TINGKAT JAWA BARAT

11. Any changes done inside slide editor is saved automatically and will be restored upon refresh.

The presented examples are only a sketch, you have to choose the appearance, size, animation, reaction time to user actions and other characteristics to improve the usability of the application.

It is expected that you will create additional interface elements and or user interaction mechanism to improve the convenience of the specified functionality.

View mode
The view mode should work in both desktop and tablet.

1. You need to create a view mode that open the preview of the slides starting from the root element with an option for moving between content based on linked section that is already modified in the route editor.
2. You also need to implement a transition when there is a movement between element.
3. The animation of transition must correspond to the direction of movement and movement along the main path or an additional one (For example, left, right, down, up, in a circle).
4. The view mode needs to be seen in full screen mode.
5. The user should be able to understand where he is at the moment and go to the desired element.
6. For navigation, this mode should provide movement controls that should be easy to use, including on touchscreen devices. And show the title of that control, which helps the user with the choice of the next step to move.
7. For navigation in desktop, the viewer allows using number keys (1,2,3,4) on keyboard to navigate to corresponding linked element.

You need to consider the user experience and design the apps accordingly to be as easy as possible when used and as clear as possible when viewed. Even if you cannot implement a full application, you may implement a view mode as a proof-of-concept to show the client how it works.

Your viewer should follow the design that you created at design module and correspond to Web Content Accessibility Guidelines (WCAG) 2.0.

INSTRUCTIONS TO THE COMPETITOR

Please follow the following instructions to deliver your work.

1. The media files are available in the ZIP file. You can modify the supplied files and create new media files to ensure the correct functionality and improve the application. You can use any supplied JavaScript framework if you find it necessary.

2. Save the working apps in a directory on the server named "XX_ lksp2019_front_end_module". Be sure that your main file is called index.html.

* XX is your country code

MARKING SCHEME

SECTION	CRITERION	JUDG. MARKS	MEAS. MARKS	TOTAL
E1	Editor functionality for Element	0	3.75	3.75
E2	Editor functionality for Connections	1	2	3
E3	Viewer requirements	0	4	4
E4	Animation	1.5	0.75	2.25
E5	Code Quality	3.5	0	3.5

SERVER MODULE

INTRODUCTION

Trilu, task management website, ask you to make their minimum viable product. Your task is to implement the backend with Laravel PHP Framework and frontend with JavaScript Framework (VueJS, AngularJS, or ReactJS). The front-end design skeleton is provided. The detail description and tools that you can use will be described below.

DESCRIPTION OF PROJECT AND TASKS

API List:

Use provided ERD to make your database. Create dummy users on users table (password is hashed using bcrypt):

username	password	First name	Last name
john.doe	12345	John	Doe
richard.roe	12345	Richard	Roe
jane.poe	12345	Jane	Poe

These are the list of web service endpoint that requested by **Trilu**:

1. Authentication

a. Register

URL: [domain]/v1/auth/register

Description: For client to register new user

Method: POST

Request Parameter:

- Body:
    - First Name
    - Last Name
    - Username
    - Password

Validation:

- First Name must be alphabet only, length between 2 – 20
- Last name must be alphabet only, length between 2 – 20
- Username only consist of alphanumeric, underscore '_', or dot '.', length between 5 – 12
- Username must be unique
- Password length between 5 - 12

Response:

- If register success, registered user automatically logged in:

  Header: response status: 200

  Body:

  - Token (authorization token generated by the system from logged in user id with bcrypt hashing method)
  - Role

- If input validation failed:

  Header: response status: 422

  Body: message: invalid field

b. Login

URL: [domain]/v1/auth/login

Description: For client to generate and get login token using username and password. Username and password must be valid.

Method: POST

Request Parameter:

- Body:
  - Username
  - Password

Response:

- If login success:

  Header: response status: 200

  Body:

  - Token (authorization token generated by the system from logged in user id with bcrypt hashing method)
  - Role

- If login failed (username or password do not match or empty):

  Header: response status: 401

  Body: message: invalid login

c. **Logout**

URL: [domain]/v1/auth/logout?token={*AUTHORIZATION_TOKEN*}

Description: For server to make token invalid

Method: GET

Response:

- If logout success:

  Header: response status: 200

  Body:

  - message: logout success

- If logout failed (token invalid):

  Header: response status: 401

  Body: message: unauthorized user

2. **Board**

   a. **Create new board**

   URL: [domain]/v1/board?token={*AUTHORIZATION_TOKEN*}

   Description: For client to create new board. Assign creator as team member when board created.

   Method: POST

   Request Parameter:

   - Body:
     o Name

   Validation:

   - name must be filled

   Response:

   - If success:
     Header: response status: 200

     Body: message: create board success

   - If input validation failed:
     Header: response status: 422

     Body: message: invalid field

   - If unauthorized user access it (only logged in user can access this endpoint):
     Header: response status: 401

     Body: message: unauthorized user

b. Update board

URL: [domain]/v1/board/{*board_id*}?token={*AUTHORIZATION_TOKEN*}

Description: For client to update existing board by id

Method: PUT

Request Parameter:

- Body:
  - Name

Validation:

- name must be filled

Response:

- If success:

  Header: response status: 200

  Body: message: update board success

- If input validation failed:

  Header: response status: 422

  Body: message: invalid field

- If unauthorized user access it (only team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

c. Delete board

URL: [domain]/v1/board/{*board_id*}?token={*AUTHORIZATION_TOKEN*}

Description: For client to delete existing board by id

Method: DELETE

Response:

- If success:

  Header: response status: 200

  Body: message: delete board success

- If unauthorized user access it (only creator can access this endpoint):
  Header: response status: 401

  Body: message: unauthorized user

d. Get all boards

URL: [domain]/v1/board?token={*AUTHORIZATION_TOKEN*}

Description: For client to get all boards data based on logged in user, as a member or creator.

Method: GET

Response:

- If success:
  Header: response status: 200

  Body: all board data in json (consists of id, name, creator_id)

- If unauthorized user access it (only logged in user can access this endpoint):
  Header: response status: 401

e. Open board

URL: [domain]/v1/board/{*board_id*}?token={*AUTHORIZATION_TOKEN*}

Description: For client to get board detail based on board_id. Lists and cards sorted by column order.

Method: GET

Response:

- If success:
  Header: response status: 200

  Body: eager load all board data in json (consists of all team members, all lists of board and cards inside list)

Format:

```
{
  "id": [board_id],
  "name": [board_name],
  creator_id: [creator_id],
  "members": [
    {
      "id": [user_id],
      "first_name": [first name],
      "last_name": [last name],
      "initial": [generated from first name and last name]
    },
    …
  ],
  "lists": [
    {
      "id": [list_id],
      "name": [list_name],
      "order": [order],
      "cards": [
        {
          "card_id": [card_id],
          "task": [card_task],
          "order", [order]
        },
        …
```

```
        ]

      },

      …

    ]

  }
```

- If unauthorized user access it (only team member can access this endpoint):

    Header: response status: 401

f. Add team member

URL:

[domain]/v1/board/*{board_id}*/member?token={*AUTHORIZATION_TOKEN*}

Description: For client to add team member to the board.

Method: POST

Request Parameter:

- Body:
  - Username

Validation:

- username must be existing in user table

Response:

- If success:

    Header: response status: 200

    Body: message: add member success

- If input validation failed:

    Header: response status: 422

    Body: message: user did not exist

- If unauthorized user access it (only team member can access this endpoint):

    Header: response status: 401

Body: message: unauthorized user

g. Remove team member

URL:
[domain]/v1/board/*{board_id}*/member/*{user_id}*?token={*AUTHORIZATION_TOKEN*}

Description: For client to delete team member from the board using user id.

Method: DELETE

Response:

- If success:

  Header: response status: 200

  Body: me   ssage: remove member success

- If unauthorized user access it (only team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

3. List

a. Create new list

URL: [domain]/v1/board/*{board_id}*/list?token={*AUTHORIZATION_TOKEN*}

Description: For client to create new list to the board

Method: POST

Request Parameter:

- Body:
  - Name

Validation:

- name must be filled

Response:

- If success:

  Header: response status: 200

  Body: message: create list success

- If input validation failed:

  Header: response status: 422

  Body: message: invalid field

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

b. Update list

URL:
[domain]/v1/board/{*board_id*}/list/{*list_id*}?token={*AUTHORIZATION_TOKEN*}

Description: For client to update existing list by id

Method: PUT

Request Parameter:

- Body:
    - Name

Validation:

- name must be filled

Response:

- If success:

  Header: response status: 200

  Body: message: update list success

- If input validation failed:

  Header: response status: 422

  Body: message: invalid field

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

c. Delete list

URL:

[domain]/v1/board/{*board_id*}/list/{*list_id*}?token={*AUTHORIZATION_TOKEN*}

Description: For client to delete existing list by id

Method: DELETE

Response:

- If success:

  Header: response status: 200

  Body: message: delete list success

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

d. Move list to right

URL:

[domain]/v1/board/{*board_id*}/list/{*list_id*}/right?token={*AUTHORIZATION_TOKEN*}

Description: For client to switch the order of the selected list with the list on the right

Method: POST

Response:

- If success:

  Header: response status: 200

  Body: message: move success

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

e. Move list to left

URL:
[domain]/v1/board/{*board_id*}/list/{*list_id*}/left?token={*AUTHORIZATION_TOKEN*}

Description: For client to switch the order of the selected list with the list on the left

Method: POST

Response:

- If success:

  Header: response status: 200

  Body: message: move success

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

4. Card

a. Create new card

URL:
[domain]/v1/board/{*board_id*}/list/{*list_id*}/card?token={*AUTHORIZATION_TOKEN*}

Description: For client to create new card to the list

Method: POST

Request Parameter:

- Body:
  - Task

Validation:

- Task must be filled

Response:

- If success:

  Header: response status: 200

Body: message: create card success

- If input validation failed:

  Header: response status: 422

  Body: message: invalid field

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

b. Update card

URL:
[domain]/v1/board/{*board_id*}/list/{*list_id*}/card/{*card_id*}?token={*AUTHORIZATION_TOKEN*}

Description: For client to update existing card by id

Method: PUT

Request Parameter:

- Body:
  - o Task

Validation:

- Task must be filled

Response:

- If success:

  Header: response status: 200

  Body: message: update card success

- If input validation failed:

  Header: response status: 422

  Body: message: invalid field

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

c. Delete card

URL:
[domain]/v1/board/{*board_id*}/list/{*list_id*}/card/{*card_id*}?token={*AUTHORIZATION_TOKEN*}

Description: For client to delete existing card by id

Method: DELETE

Response:

- If success:

  Header: response status: 200

  Body: message: delete card success

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

d. Move up card

URL: [domain]/v1/card/{*card_id*}/up?token={*AUTHORIZATION_TOKEN*}

Description: For client to switch the order of the selected card with the card above

Method: POST

Response:

- If success:

  Header: response status: 200

  Body: message: move success

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

e. Move down card

URL: [domain]/v1/card/*{card_id}*/down?token={*AUTHORIZATION_TOKEN*}

Description: For client to switch the order of the selected card with the card below

Method: POST

Response:

- If success:

  Header: response status: 200

  Body: message: move success

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

f. Move card to another list

URL:
[domain]/v1/card/*{card_id}*/move/*{list_id}*?token={*AUTHORIZATION_TOKEN*}

Description: For client to move card to another list. List must be in the same board.

Method: POST

Response:

- If success:

  Header: response status: 200

  Body: message: move success

- If list id is not in the same board:

  Header: response status: 422

  Body: message: move list invalid

- If unauthorized user access it (only board team member can access this endpoint):

  Header: response status: 401

  Body: message: unauthorized user

The complete minimum viable product for **Trilu** system should cover the following requirement:

| Menu	Detail
Login & Register	• User can register into the system on the register page • User can login (and logout) into the system on the start page of the application • After login, user directed to "home" menu • Links to the Home and Logout are always visible on the top part of the page while the user is logged in
Home	• On "home" menu, user can add new board or access board if the user is the team member • User can update board's name by clicking the board name, type new name, then press Enter to submit • Creator of the board can delete a board by deleting its entire name, then press Enter to submit • Make sure your system is preventing users to view and access unauthorized board
Board	• On "board" menu, user can manage board: members, lists, and cards **Member** • User can add new member using username • User can remove member by clicking its initial then click confirmation button **List** • User can add new list to the board • User can update list's name by clicking the list name, type new name, then press Enter to submit • User can delete a list by deleting its entire name, then press Enter to submit • User can move list left and right **Card** • User can add new card to a list • User can update card's task by clicking the card, type new task, then press Enter to submit • User can delete a card by deleting its entire task, then press Enter to submit • User can move card up and down in the list • User can move card to another list by click the card, then click another list.

ERD

You can use and improve ERD below:
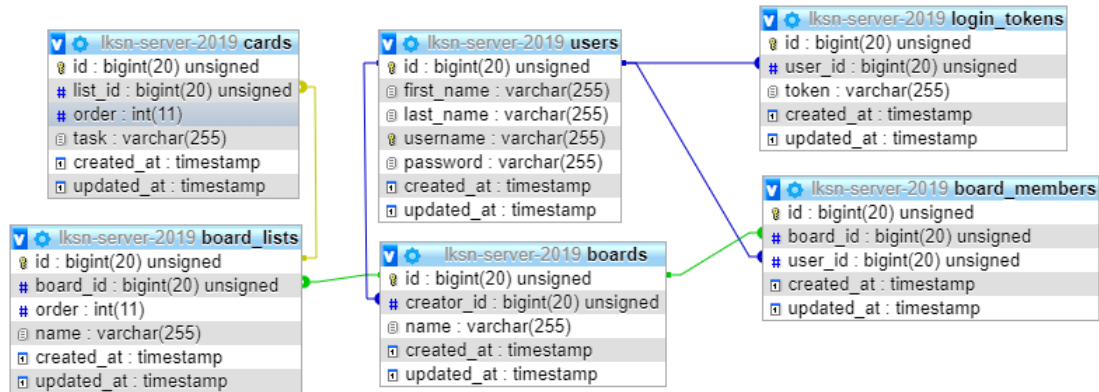
INSTRUCTIONS TO THE COMPETITOR

- Save your files in your root directory on the server called "**XX_LKSP2019_SERVER_MODULE**" where XX is your computer number.

- Generate your database dump called **XX_LKSP2019_DB_DUMP.sql** and store in your root application folder,

- *Create/generate a db-diagram named "db-diagram.xxx" (xxx is the extension/type of the file eg. Pdf or jpg) and put it into the directory mentioned above. Example:*
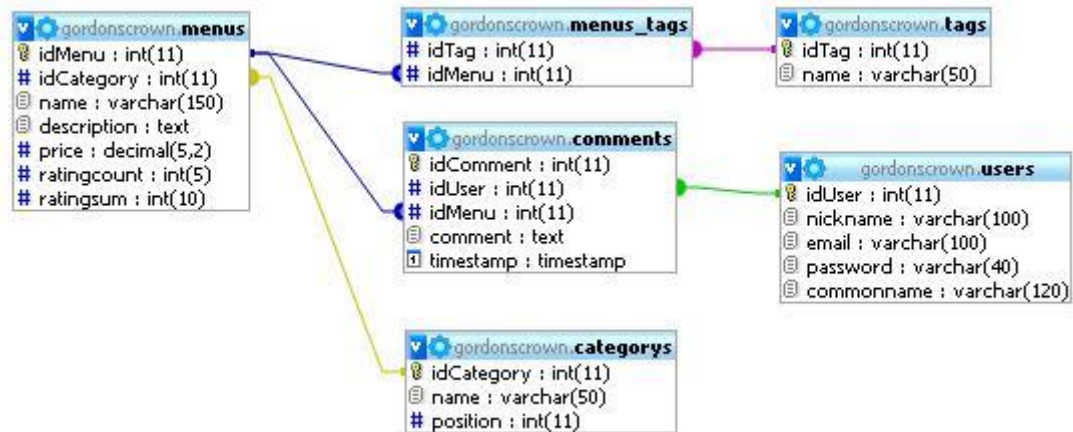
- *For this module, you must use one of the three available frameworks provided. Applications developed without use of any of these frameworks will not be considered. You should take advantage of the framework as much as possible.*

SECTION	CRITERION	JUDG. MARKS	MEAS. MARKS	TOTAL
C1	Server API Framework	1.5	0	1.5
C2	Server API Database	0	3.5	3.5
C3	Server API Login	0	2.5	2.5
C4	Server API Board	0	4.5	4.5
C5	Server API Board Member	0	1.5	1.5
C6	Server API List	0	3.0	3.0
C7	Server API Card	0	4.0	4.0

SECTION	CRITERION	JUDG. MARKS	MEAS. MARKS	TOTAL
D1	Server Front End Framework	1.5	1	2.5
D2	Server Front End Template	2.0	0	2.0
D3	Server Front End Authentication	3.0	0	3.0
D4	Server Front End Board	0	4.0	4.0
D5	Server Front End List	0	4.0	4.0
D6	Server Front End Card	0	5.0	5.0