MURUGAPPA POLYTECHNIC COLLEGE

Sathyamurthy Nagar, Chennai-600 062.

(An Academically Autonomous Institution)

DEPARTMENT OF COMPUTER ENGINEERING

PROJECT REPORT

ON

WEB PAGE BUILDER

Submitted By

Name	Register No
JOHN AUGUSTIN N	1912256
MADHAN RAJ D	1912264
MIRTHUN K	1912268
SANTHOSH P	1912289
YUGENDIRAN G	1912299

In partial fulfilment of the requirements of the award of

DIPLOMA IN COMPUTER ENGINEERING (REGULAR)

Under the guidance of

Mr. S. THIRUMALAI, M.E.
Lecturer, Department of Computer Engineering

APRIL 2022

MURUGAPPA POLYTECHNIC COLLEGE

Sathyamurthy Nagar, Chennai-600 062.

(An Academically Autonomous Institution)

DEPARTMENT OF COMPUTER ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report entitled **“WEB PAGE BUILDER”** is the bonafide record of work done by Selvan _____________________ Registration No__________, student of Sixth Semester Computer Engineering (Regular) Diploma Course, during MARCH 2021 – APRIL 2022 in fulfillment of the requirements for the award of **DIPLOMA IN COMPUTER ENGINEERING (REGULAR).**

Project Guide	**Head of the Department**
Name: Mr. S. THIRUMALAI, M.E	Name: Mr. B. RAJENDRAN, M.E
Date:	Date:

Submitted for the End Semester Examination held on……………………………………

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

Any work completed consists not only the experience and skill of an individual but also it includes an organized body of people. As such, many kind hearts lay behind the completion of our project.

First, we wish our grateful gratitude to our parents who dedicated their life for our studies. We express our deep sense of gratitude to our beloved principal **Dr.K.SUDHAGAR, Ph.D.** whose motivation has helped us to develop this project.

We wish to express our sincere thanks to **HOD** of Computer Engineering **Mr.B.RAJENDRAN, M.E** and our guide **Mr. S. THIRUMALAI, M.E**. The valuable guidance and encouragement given by guide have helped us in successfully completing this project.

We also wish to thank all staff members of Computer Engineering Department for their guidance and encouragement.

We place in record all our profound gratitude to helping hands for their courtesy and guidance.

The team members are very thankful to their friends and fellow students for helping us during the course of this project Work.

SYNOPSIS

There are number of coding languages available in order to build web pages. So, there is need to know any one coding language to build a web page. This innovative software application allows users to build web pages without knowing any coding language.

It is specifically designed for the internal use for companies. This software helps to build/design graphical Web pages. This software is embedded in a website for professional use, and will be available for customers to tailor web pages as per their need. This system helps the user to build web page with effective graphical user interface. User doesn’t have to work specifically for GUI. System will display various webpage template user can select any template according to his preference.

This system will save time of the user and will help the user to concentrate on main functionality required in their webpage. This system will provide user with professional webpage templates they can select the templates based on the requirement and functionality. Here in this system user can build whole website by just selecting the content and images required in their webpage. User can design each and every web page in their websites according to their preference by selecting the content and images. User can even specify position of the content to be placed.

When user logins to the system, system will display various templates to the user. User has to select the template. He must specify the content and images that needs to be placed in the webpage. User can place the content and can view the template simultaneously. Once the user clicks onto the submit button system will generate the website. System will generate code and .zip file to the user. User can build custom websites easily using this application. User doesn’t have to know any coding languages to build web pages. With the help of this system user can work on main functionality required in their web page.

S.NO	CONTENTS	PAGE NO
01.	**INTRODUCTION**	**01**
	1.1 AIM OF THE PROJECT	
	1.2 OBJECTIVE OF THE PROJECT	
02.	**SYSTEM ANALYSIS**	**02**
	2.1 EXISTING SYSTEM	
	2.2 PROPOSED SYSTEM	
03.	**SYSTEM REQUIREMENTS**	**03**
	3.1 HARDWARE REQUIREMENTS	
	3.2 SOFTWARE REQUIREMENTS	
04.	**SYSTEM DESIGN**	**04**
	4.1 SYSTEM ARCHITECTURE	
	4.2 DATABASE DESIGN	
05.	**MODULES DESCRIPTION**	**07**
06.	**LANGUAGE SPECIFICATION**	**09**
07.	**SAMPLE CODING**	**21**
08.	**SYSTEM TESTING AND MAINTENANCE**	**31**
09.	**SCREEN SHOTS**	**34**
10.	**COST ESTIMATION**	**39**
11.	**CONCLUSION**	**40**
12.	**BIBLOGRAPHY**	**41**
13.	**USER MANUAL**	**42**

1. INTRODUCTION

1.1 AIM OF THE PROJECT

A webpage-builder's aim to help the average user to build a site without the need or at least with minimal intervention from a webmaster or designer. Using drag-and-drop tools, users can build a site by arranging elements in blocks or replacing templates and themes with their own copy and images.

The result is non-technical users, especially small business owners, can now build, run and update their sites even without coding skills. This bodes well to their ability to feed their customers' content consumption.

1.2 OBJECTIVES

This software helps to build/design graphical Web pages. This software is embedded in a website for professional use, and will be available for customers to tailor web pages as per their need.

User doesn't have to work specifically for GUI. System will display various webpage template user can select any template according to his preference.

Here in this system user can build whole website by just selecting the content and images required in their webpage.

System will generate code and .zip file to the user. User can build custom websites easily using this application.

2. SYSTEM ANALYSIS

FEASIBILITY STUDY

Feasibility studies to aim to objectively and rationally identification systems Uncover the strengths and weakness of the existing business or proposed venture, opportunities and threats as presented by the environment, the resources required to carry through and ultimately the prospects for success.

2.1 EXISTING SYSTEM

In the old days building a website required a lot of coding and a lot of work. And it’s very logical and slow. Those websites are only static and poor in design. It cannot contain images, videos, etc. And now there are some website builders now in the current technological development. Websites can be designed and hosted within some clicks. Most of the people cannot understand the work flow of many website builders.

2.2 PROPOSED SYSTEM

Now there are much more efficient solutions. For example, using a website builder to have your website up and running within minutes. Whether you’re a freelancer, blogger or own a small business, a website builder makes it so you can create your website from the comfort of your home without needing to hire external help. In this project user’s work flow is very clean and designing is very simple so that users can understand the system easily. Here users can download the web page template also.

3. SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

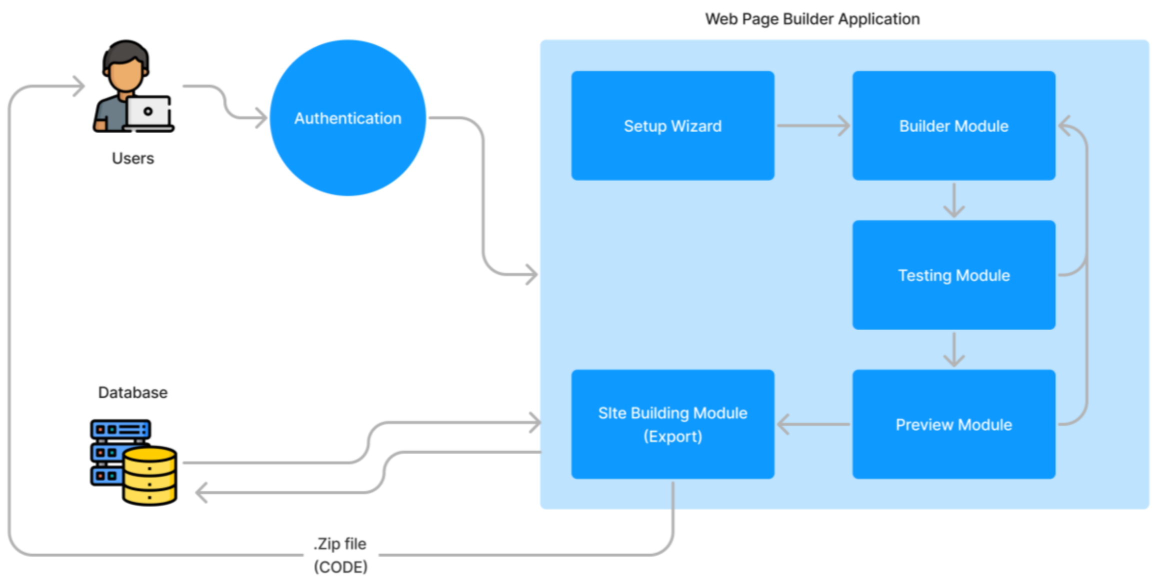
➢ Processor	:	Intel I3
➢ RAM	:	4GB
➢ Hard Disk	:	250GB

3.2 SOFTWARE REQUIREMENTS

➢ Operating System	:	Windows 10
➢ Languages	:	HTML, CSS, JAVASCRIPT, PHP
➢ Frameworks	:	GrapesJS, Bootstrap
➢ Preposessor	:	Sass
➢ Database	:	MySQL
➢ Applications	:	Xampp, Visual Studio Code, Sass Compiler, NodeJS, NPM, Chrome, Command Prompt.

4. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE



4.2 DATABASE DESIGN

Themes:

S.No	Name	Type	Collation	Null	Default	Extra
1	theme_id	int(11)		No	None	A_I
2	themes_name	varchar(255)	latin1_swedish_ci	No	None	
3	themes_html	text	latin1_swedish_ci	No	None	

Users:

S.No	Name	Type	Collation	Null	Default	Extra
1	user_id	int(11)		No	None	A_I
2	user_name	varchar(255)	utf8mb4_general_ci	No	None	
3	user_email	varchar(255)	utf8mb4_general_ci	No	None	
4	user_pass	text	utf8mb4_general_ci	No	None	
5	user_status	varchar(30)	utf8mb4_general_ci	No	None	

Users_link:

S.No	Name	Type	Collation	Null	Default	Extra
1	users_link_id	int(11)		No	None	A_I
2	users_link_email	varchar (255)	utf8mb4_general_ci	No	None	
3	users_link_token	text	utf8mb4_general_ci	No	None	
4	users_link_status	varchar (50)	utf8mb4_general_ci	No	None	
5	users_link_status	varchar (50)	utf8mb4_general_ci	No	None	
6	users_link_exp_dt	varchar (50)	utf8mb4_general_ci	No	None	
7	users_link_module	varchar (255)	utf8mb4_general_ci	No	None	

5. MODULES DESCRIPTION

1. Setup Wizard
2. Builder Module
3. Testing & Preview
4. Exporting
5. Database & Authentication

1. Setup Wizard

Setup wizard is the most 1st step in this application which allows the users to choose the site name and site template as per their requirements.

2. Builder Module

Builder is the important section in this application where the user chooses the site name and template then they can customize the template as per their expectations and requirements. This includes texts, headings, picture, etc.… everything can be customized by the user.

3.Testing & Preview

This module is responsible for testing the customized template such as responsibility, glitches, empty content, size, etc.… If any of the unsatisfied section found automatically user will redirect back to fix the problem in the template.

4.Exporting

Once the user customized the template as per the requirements they are finally now to export the site to code. The algorithm gets all the user's customization and customize the template and save the archive of the site in the database.

5. Database & Authentication:

Once the code exported to the database the user needs to be authenticated by the system to login the application. This is a process where the exporting module downloads the source code and the user can extract the file and use it.

6. LANGUAGE SPECIFICATION

HTML

The Hypertext Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

A form of HTML, known as HTML5, is used to display video and audio, primarily using the <canvas> element, in collaboration with JavaScript.

History

In 1980, physicist Tim Berners-Lee, a contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents.

In 1989, Berners-Lee wrote a memo proposing an Internet-based hypertext system. Berners-Lee specified HTML and wrote the browser and server software in late Robert Cillian collaborated on a joint request for funding, but the project was not formally adopted by CERN. In his personal notes from 1990 he listed "some of the many areas in which hypertext is used" and put an encyclopedia first.

The first publicly available description of HTML was a document called "HTML Tags", first mentioned on the Internet by Tim Berners-Lee in late 1991. It describes 18 elements comprising the initial, relatively simple design of HTML.

Except for the hyperlink tag, these were strongly influenced by SGMLguid, an in-house Standard

Generalized Markup Language (SGML)-based documentation format at CERN.

Eleven of these elements still exist in HTML 4.

HTML version timeline

➢ **HTML 2**

November 24, 1995

HTML 2.0 was published as RFC 1866. Supplemental RFCs added capabilities:

- November 25, 1995: RFC 1867 (form-based file upload)
- May 1996: RFC 1942 (tables)
- August 1996: RFC 1980 (client-side image maps)
- January 1997: RFC 2070 (internationalization)

➢ **HTML 3**

January 14, 1997

HTML 3.2 was published as a W3C Recommendation. It was the first version developed and standardized exclusively by the W3C, as the IETF had closed its HTML Working Group on September 12, 1996.

Initially code-named "Wilbur", HTML 3.2 dropped math formulas entirely, reconciled overlap among various proprietary extensions and adopted most of Netscape's visual markup tags. Netscape's blink element and Microsoft's marquee element were omitted due to a mutual agreement between the two companies. A markup for mathematical formulas similar to that in HTML was not standardized until 14 months later in MathML.

➢ **HTML 4**

December 18, 1997

HTML 4.0 was published as a W3C Recommendation. It offers three variations:

- Strict, in which deprecated elements are forbidden
- Transitional, in which deprecated elements are allowed
- Frameset, in which mostly only frame related elements are allowed.

Initially code-named "Cougar", HTML 4.0 adopted many browser-specific element types and attributes, but at the same time sought to phase out Netscape's visual markup features by marking them as deprecated in favor of style sheets. HTML 4 is an SGML application conforming to ISO 8879 – SGML.

April 24, 1998

HTML 4.0 was reissued with minor edits without incrementing the version number.

December 24, 1999

HTML 4.01 was published as a W3C Recommendation. It offers the same three variations as HTML 4.0 and its last errata were published on May 12, 2001.

May 2000

ISO/IEC 15445:2000 ("ISO HTML", based on HTML 4.01 Strict) was published as an ISO/IEC international standard. In the ISO these standard falls in the domain of the ISO/IEC JTC1/SC34 (ISO/IEC Joint Technical Committee 1, Subcommittee 34 – Document description and processing languages).

After HTML 4.01, there was no new version of HTML for many years as development of the parallel, XML-based language XHTML occupied the W3C's HTML Working Group through the early and mid-2000s.

HTML 5

Main article: HTML5

October 28, 2014

HTML5 was published as a W3C Recommendation.

November 1, 2016

HTML 5.1 was published as a W3C Recommendation.

December 14, 2017

HTML 5.2 was published as a W3C Recommendation.

CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .CSS file, which reduces complexity and repetition in the structural content; and enable the .CSS file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as onscreen, in print, by voice (via speech-based browser or screen reader).

Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

SYNTAX

CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties.

A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors, and a declaration block.

Selector

In CSS, *selectors* declare which part of the markup a style applies to by matching tags and attributes in the markup itself.

Selectors may apply to the following:

➢ All elements of a specific type, e.g. the second-level headers h2 ➢ Elements specified by attribute, in particular:
  - id: an identifier unique within the document, identified with a hash prefix e.g., #id
  - class: an identifier that can annotate multiple elements in a document, identified with a period prefix e.g. classname

➢ elements depending on how they are placed relative to others in the document tree.

Classes and IDs are case-sensitive, start with letters, and can include alphanumeric characters, hyphens, and underscores. A class may apply to any number of instances of any elements. An ID may only be applied to a single element.

Pseudo-classes are used in CSS selectors to permit formatting based on information that is not contained in the document tree. One example of a widely used pseudo-class is: however, which identifies content only when the user "points to" the visible element, usually by holding the mouse cursor over it. It is appended to a selector as in a: however, or #elementid: hover. A pseudo-class classifies document elements, such as: link or: visited, whereas a pseudo-element makes a selection that may consist of partial elements, such as: first-line or: first-letter.

Selectors may be combined in many ways to achieve great specificity and flexibility. Multiple selectors may be joined in a spaced list to specify elements by location, element type, id, class, or any combination thereof. The order of the selectors is important. For example, div .my Class {color: red;} applies to all elements of class my Class that are inside div elements, whereas

Class div {color: red;} applies to all div elements that are inside elements of class.

This is not to be confused with concatenated identifiers such as div. my Class {color: red;} which applies to div elements of class my Class.

Use

Before CSS, nearly all presentational attributes of HTML documents were contained within the HTML markup. All font colors, background styles, element alignments, borders and sizes had to be explicitly described, often repeatedly, within the HTML. CSS lets authors move much of that information to another file, the style sheet, resulting in considerably simpler HTML.

For example, headings (h1 elements), sub-headings (h2), sub-sub-headings (h3), etc., are defined structurally using HTML. In print and on the screen, choice of font, size, color and emphasis for these elements is presentational.

Before CSS, document authors who wanted to assign such typographic characteristics to, say, all h2 headings had to repeat HTML presentational markup for each occurrence of that heading type. This made documents more complex, larger, and more error-prone and difficult to maintain. CSS allows the separation of presentation from structure. CSS can define color, font, text alignment, size, borders, spacing, layout and many other typographic characteristics, and can do so independently for on-screen and printed views. CSS also defines non-visual styles, such as reading speed and emphasis for aural text readers. The W3C has now deprecated the use of all presentational HTML markup.

JavaScript

JavaScript often abbreviated JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client side for web page behavior, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices.

JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

The ECMAScript standard does not include any input/output (I/O), such as networking, storage, or graphics facilities. In practice, the web browser or other runtime system provides JavaScript APIs for I/O.

JavaScript engines were originally used only in web browsers, but are now core components of some servers and a variety of applications. The most popular runtime system for this usage is Node.js.

Although Java and JavaScript are similar in name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

History

Creation at Netscape

The first web browser with a graphical user interface, Mosaic, was released in 1993. Accessible to non-technical people, it played a prominent role in the rapid growth of the nascent World Wide Web. The lead developers of Mosaic then founded the Netscape corporation, which released a more polished browser, Netscape Navigator, in 1994. This quickly became the most-used.

Adoption by Microsoft

Microsoft debuted Internet Explorer in 1995, leading to a browser war with Netscape. On the JavaScript front, Microsoft reverse-engineered the Navigator interpreter to create its own, called JScript.

The rise of JScript

In November 1996, Netscape submitted JavaScript to Emma International, as the starting point for a standard specification that all browser vendors could conform to. This led to the official release of the first ECMAScript language specification in June 1997.

The standards process continued for a few years, with the release of ECMAScript 2 in June 1998 and ECMAScript 3 in December 1999. Work on ECMAScript 4 began in 2000.

Meanwhile, Microsoft gained an increasingly dominant position in the browser market. By the early 2000s, Internet Explorer's market share reached 95%. This meant that JScript became the de facto standard for client-side scripting on the Web.

Growth and standardization

During the period of Internet Explorer dominance in the early 2000s, clientside scripting was stagnant. This started to change in 2004, when the successor of Netscape, Mozilla, released the Firefox browser. Firefox was well received by many, taking significant market share from Internet Explorer.

In 2005, Mozilla joined ECMA International, and work started on the ECMAScript for XML (E4X) standard. This led to Mozilla working jointly with Macromedia (later acquired by Adobe Systems), who were implementing E4X in their ActionScript 3 language, which was based on an ECMAScript 4 draft. The goal became standardizing ActionScript 3 as the new ECMAScript 4. To this end, Adobe Systems released the Tamarin implementation as an open source project. However, Tamarin and ActionScript 3 were too different from established client-side scripting, and without cooperation from Microsoft, ECMAScript 4 never reached fruition.

Reaching maturity

Ambitious work on the language continued for several years, culminating in an extensive collection of additions and refinements being formalized with the publication of ECMAScript 6 in 2015.

The creation of Node.js in 2009 by Ryan Dahl sparked a significant increase in the usage of JavaScript outside of web browsers. Node combines the V8 engine, an event loop, and I/O APIs, thereby providing a stand-alone JavaScript runtime system. As of 2018, Node had been used by millions of developers, and name had the most modules of any package manager in the world.

PHP

PHP is a general-purpose scripting language geared toward web development. It was originally created by Danish-Canadian programmer RasmusLerdorf in 1994. The PHP reference implementation is now produced by The PHP Group. PHP originally stood for Personal Home Page, but it now stands for the recursive initialism PHP: Hypertext Preprocessor.

PHP code is usually processed on a web server by a PHP interpreter implemented as a module, a daemon or as a Common Gateway Interface (CGI) executable. On a web server, the result of the interpreted and executed PHP code – which may be any type of data, such as generated HTML or binary image data – would form the whole or part of an HTTP response. Various web template systems, web content management systems, and web frameworks exist which can be employed to orchestrate or facilitate the generation of that response. Additionally, PHP can be used for many programming tasks outside the web context, such as standalone graphical applications and robotic drone control. PHP code can also be directly executed from the command line.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on a variety of operating systems and platforms.

The PHP language evolved without a written formal specification or standard until 2014, with the original implementation acting as the de facto standard which other implementations aimed to follow. Since 2014, work has gone on to create a formal PHP specification.

W3Techs reports that, as of January 2022, "PHP is used by 78.1% of all the websites whose server-side programming language we know." PHP version 7.4 is the most used version. Support for version 7.3 was dropped on 6 December 2021.

HISTORY

PHP development began in 1994 when Rasmus Lerdorf wrote several Common Gateway Interface (CGI) programs in C, which he used to maintain his personal homepage. He extended them to work with web forms and to communicate with databases, and called this implementation "Personal Home Page/Forms Interpreter" or PHP/FI.

PHP/FI could be used to build simple, dynamic web applications. To accelerate bug reporting and improve the code, Lerdorf initially announced the release of PHP/FI as "Personal Home Page Tools (PHP Tools) version 1.0" on the Usenet discussion group comp.infosystems.www.authoring.cgi on June 8, 1995. This release already had the basic functionality that PHP has today. This included Perl-like variables, form handling, and the ability to embed HTML.
The syntax resembled that of Perl, but was simpler, more limited and less consistent.

Zeev Suraski and Andi Gutmans rewrote the parser in 1997 and formed the base of PHP 3, changing the language's name to the recursive acronym PHP: Hypertext Preprocessor. Afterwards, public testing of PHP 3 began, and the official launch came in June 1998. Suraski and Gutmans then started a new rewrite of PHP's core, producing the Zend Engine in 1999. They also founded Zend Technologies in Ramat Gan, Israel.

On 1 July 2004, PHP 5 was released, powered by the new Zend Engine II. PHP 5 included new features such as improved support for object-oriented programming, the PHP Data Objects (PDO) extension (which defines a lightweight and consistent interface for accessing databases), and numerous performance enhancements. In 2008, PHP 5 became the only stable version under development. Late static binding had been missing from previous versions of PHP, and was added in version 5.3.

PHP received mixed reviews due to lacking native Unicode support at the core language level. In 2005, a project headed by Andrei Zmievski was initiated to bring native Unicode support throughout PHP, by embedding the International Components for Unicode (ICU) library, and representing text strings as UTF16 internally. Since this would cause major changes both to the internals of the language and to user code, it was planned to release this as version 6.0 of the language, along with other major features then in development.

During 2014 and 2015, a new major PHP version was developed, PHP 7. The numbering of this version involved some debate among internal developers. While the PHP 6 Unicode experiment had never been released, several articles and book titles referenced the PHP 6 name, which might have caused confusion if a new release were to reuse the name. After a vote, the name PHP 7 was chosen.

PHP 8 was released on November 26, 2020. PHP 8 is a major version and has breaking changes from previous versions.

PHP 8.1 was released on November 25, 2021. It included several improvements, such as enumerations (also called "enums"), readonly properties and array unpacking with string keys.

MySQL

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of cofounder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter, and YouTube.

HISTORY

MySQL was created by a Swedish company, MySQL AB, founded by Swedes David Ax mark, Allan Larsson and Finland Swede Michael "Monty" Widenius. Original development of MySQL by Widenius and Ax mark began in 1994.[22] The first version of MySQL appeared on 23 May 1995. It was initially created for personal usage from MySQL based on the low-level language ISAM, which the creators considered too slow and inflexible. They created a new SQL interface, while keeping the same API as MySQL. By keeping the API consistent with the MySQL system, many developers were able to use MySQL instead of the (proprietarily licensed) MySQL antecedent.

7. SAMPLE CODINGS

DATABASE CONNECTION

```
<?php

ob_start();
session_start();
date_default_timezone_set("Asia/Kolkata");

$connection = mysqli_connect("localhost", "root", "", "site");

if(!$connection){
    alertBox("Database Not Connected");
}else{
    if(isset($_SESSION['login_user_id'])){
        $login_user_id = $_SESSION['login_user_id'];
    }else{
        header("location: login/index.php");
    }
}

function alertBox($msg){
    echo "<script>alert($msg);</script>";
}

?>
```

SCRIPTS:

```
var siteHolder = document.getElementById("site_loader");


var pageRenderContainer = document.getElementById("page_render_container");


var buildBtn = document.getElementById("buildBtn");


function openFrame(theme, siteName){


    siteHolder.style.display = "block";


    pageRenderContainer.setAttribute("src", "page_render.php?theme="+theme);


    buildBtn.setAttribute("href",
"builder.php?template="+theme+"&site_name="+siteName);


    window.onclick = function(event) {
        if (event.target == siteHolder) {
            siteHolder.style.display = "none";
        }
    }
}
var editor = grapesjs.init({
  height: '100%',
  noticeOnUnload: 0,
  storageManager: { autoload: 0 },
  container: '#gjs',
  fromElement: true,


  plugins: [
    "gjs-preset-webpage",
    "grapesjs-tabs",
    "grapesjs-lory-slider",
    "gjs-blocks-flexbox",
    "grapesjs-custom-code",
    "grapesjs-tooltip",
    "grapesjs-typed",
    "grapesjs-tui-image-editor",
    "grapesjs-blocks-bootstrap4"
  ],
  pluginsOpts: {
    "gjs-preset-webpage": {},
    "grapesjs-tabs": {},
    "grapesjs-lory-slider": {},
    "gjs-blocks-flexbox": {},
    "grapesjs-custom-code": {},
    "grapesjs-tooltip": {},
    "grapesjs-typed": {},
    'grapesjs-tui-image-editor': {
      config: {
        includeUI: {
          initMenu: 'filter',
        },
      },
      script: [
        'https://cdnjs.cloudflare.com/ajax/libs/fabric.js/1.6.7/fabric.min.js',
        'https://uicdn.toast.com/tui.code-snippet/v1.5.2/tui-code-snippet.min.js',
        'https://uicdn.toast.com/tui-color-picker/v2.2.7/tui-color-picker.min.js',
        'https://uicdn.toast.com/tui-image-editor/v3.15.2/tui-image-editor.min.js'
      ],
      style: [
        'https://uicdn.toast.com/tui-color-picker/v2.2.7/tui-color-picker.min.css',
        'https://uicdn.toast.com/tui-image-editor/v3.15.2/tui-image-editor.min.css',
      ],
    },
    "grapesjs-blocks-bootstrap4": {
      blocks: {},
      blockCategories: {},
      labels: {},
      gridDevicesPanel: true,
      formPredefinedActions: [
          {name: 'Contact', value: '/contact'},
          {name: 'landing', value: '/landing'},
      ]
    }
  },
  canvas: {
    styles: [
        'https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css'
    ],
    scripts: [
        'https://code.jquery.com/jquery-3.5.1.slim.min.js',
        'https://unpkg.com/@popperjs/core@2',
        'https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js'
    ],
}
});
window.editor = editor;
```

BUILDER MODULE:

```
<?php

include "db/conn.php";

if(isset($_GET['template']) && isset($_GET['site_name'])){
  $template_name = $_GET['template'];
  $site_name = $_GET['site_name'];

  $select_theme_query = "SELECT * FROM themes WHERE theme_id =
$template_name";
  $select_theme_result = mysqli_query($connection, $select_theme_query);
  $theme_count = mysqli_num_rows($select_theme_result);

  if($theme_count >= 1){
    while($row = mysqli_fetch_assoc($select_theme_result)){
        $db_theme_id = $row['theme_id'];
        $db_themes_name = $row['themes_name'];
        $themes_html = $row['themes_html'];
    }
  }else{
    header("location: index.php");
  }
}else{
  header("location: index.php");
}
?>
<!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" />
    <title>Web Page Builder</title>
    <script src="js/lib/grapes.min.js"></script>
    <link rel="stylesheet" href="css/lib/grapes.min.css" />

    <!-- Add Style and Script for Preset Webpage Builder -->
    <script src="js/lib/grapesjs-preset-webpage.min.js"></script>
    <link rel="stylesheet" href="css/lib/grapesjs-preset-webpage.min.css" />

    <!-- Plugins -->
    <script src="js/lib/grapesjs-tabs.min.js"></script>
    <script src="js/lib/grapesjs-lory-slider.min.js"></script>
    <script src="js/lib/grapesjs-blocks-flexbox.min.js"></script>
    <script src="js/lib/grapesjs-custom-code.min.js"></script>
    <script src="js/lib/grapesjs-tooltip.min.js"></script>
    <script src="js/lib/grapesjs-typed.min.js"></script>
    <script src="js/lib/grapesjs-tui-image-editor.min.js"></script>

    <!-- Modules -->
    <script src="../dist/grapesjs-blocks-bootstrap4.min.js"></script>

    <!-- Add-on -->
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.1.1/css/all.min.css" integrity="sha512-
KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13u
QjoXtEKNosOWaZqXgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer"
/>

    <link rel="stylesheet" href="css/main.css" />
  </head>
  <body>

    <div id="gjs" style="overflow:hidden">
    <?php
echo $themes_html;
```

```
    ?>
      <style>

      </style>
    </div>


    <script src="js/main.js"></script>
  </body>
</html>
```

LOGIN AUTHENTICATION:

```
<?php
if(isset($_POST['login'])){
        $login_email = $_POST['email'];
        $login_pass = $_POST['pass'];
        $select_users_query = "SELECT * FROM users WHERE user_email =
'$login_email'";
        $select_users_result = mysqli_query($connection, $select_users_query);
        $select_users_row = mysqli_num_rows($select_users_result);
        alertBox("There is no user with this email.");
        if($select_users_row >= 1){
                while($row = mysqli_fetch_assoc($select_users_result)){
                        $db_user_id = $row['user_id'];
                        $db_user_name = $row['user_name'];
                        $db_user_email = $row['user_email'];
                        $db_user_pass = $row['user_pass'];
                        $db_user_status = $row['user_status'];
                }
                if($db_user_status == 'activated'){
                        if($db_user_email == $login_email && $db_user_pass ==
md5($login_pass)){
                                $_SESSION['login_user_id'] = $db_user_id;
                                header("location: ../index.php");
                        }else{
                                alertBox("Incorrect Password. Please try again");
                        }
                }else{
                        alertBox("Please activate your account. Check your mail inbox");
                }
        }else{
                alertBox("There is no user with this email.");
        }
}
```

PACKAGE JSON:

```
{
  "name": "grapesjs-plugin-ckeditor",
  "version": "0.0.10",
  "description": "Replace the built-in RTE with CKEditor",
  "main": "dist/grapesjs-plugin-ckeditor.min.js",
  "scripts": {
    "lint": "eslint src",
    "v:patch": "npm version --no-git-tag-version patch",
    "build": "npm run v:patch && webpack --env.production",
    "start": "webpack-dev-server --open --progress --colors",
    "scss": "sass --no-source-map --style compressed --watch scss:public/css"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/Yugendiran/site_builder.git"
  },
  "keywords": [
    "grapesjs",
    "plugin",
    "ckeditor",
    "wysiwyg"
  ],
  "author": "Artur Arseniev",
  "license": "BSD-3-Clause",
  "babel": {
    "presets": [
      "env"
    ],
    "plugins": [
      "transform-object-rest-spread"
    ]
  },
  "peerDependencies": {
    "ckeditor": "4.x",
    "grapesjs": "0.x"
  },
  "devDependencies": {
    "babel-core": "^6.26.0",
    "babel-loader": "^7.1.2",
    "babel-plugin-transform-object-rest-spread": "^6.26.0",
    "babel-preset-env": "^1.6.1",
    "eslint": "^4.1.1",
    "html-webpack-plugin": "^2.30.1",
    "webpack": "^5.72.1",
    "webpack-cli": "^4.9.2",
    "webpack-dev-server": "^4.9.0"
  },
  "dependencies": {
    "grapesjs-aviary": "^0.1.2",
    "grapesjs-blocks-bootstrap4": "^0.2.3",
    "grapesjs-plugin-ckeditor": "^0.0.10",
    "sass": "^1.51.0"
  }
}
```

8. SYSTEM TESTING AND MAINTENANCE

The various levels of testing are

1. White Box Testing
2. Black Box Testing
3. Unit Testing
4. Functional Testing
5. Performance Testing
6. Integration Testing
7. Configuring testing
8. Validation Testing
9. Database Testing

White Box Testing

White-box testing (also known as clear box testing, glass box testing, and transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application.

In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases.

Black Box Testing

Black-box testing is a method of software testing that examines the functionality of an application (e.g. what the software does) without peering into its internal structures or workings (see white-box testing).

This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher level testing, but can also dominate unit testing as well

Unit testing

In computer programming, **unit testing** is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use.

Functional testing

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test.

Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing).

Performance testing

In software engineering, performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload.

It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage.

Configuration testing

Rather than testing for performance from the perspective of load, tests are created to determine the effects of configuration changes to the system's components on the system's performance and behavior.

A common example would be experimenting with different methods of load-balancing.

Integration testing

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing.

Verification and validation

Verification and Validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it full fills its intended purpose.

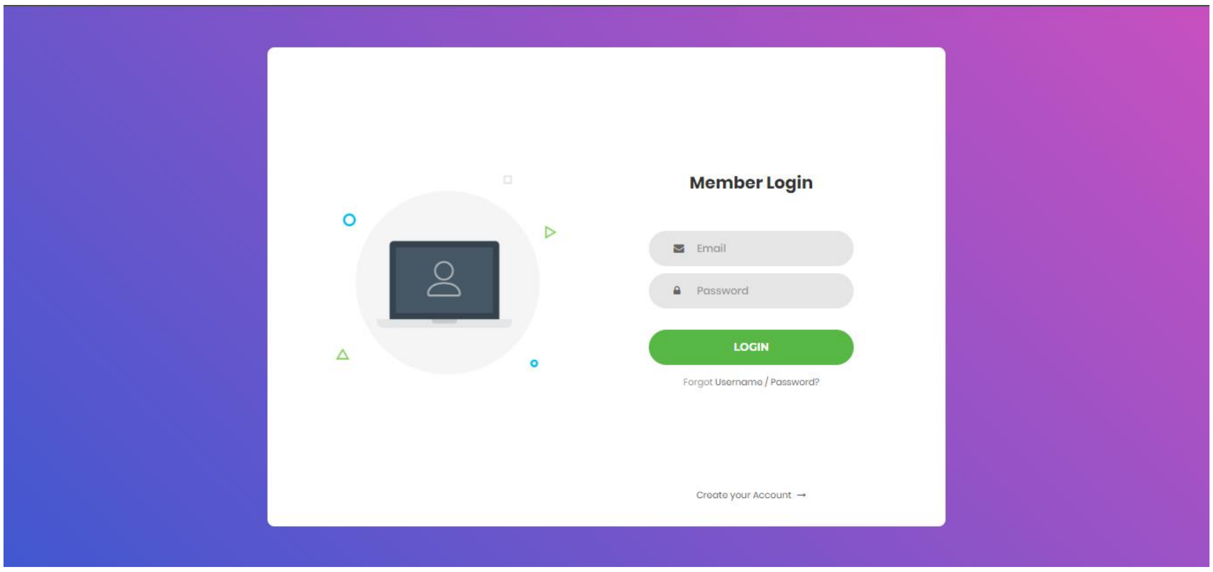
Database Testing

In Database testing backend records are tested which have been inserted through the web or desktop applications. The data which is displaying in the web application should match with the data stored in the Database.

- The tester should understand the functional requirements, business logic, application flow and database design thoroughly.
- The tester should figure out the tables, triggers, store procedures, views and cursors used for the application.
- The tester should understand the logic of the triggers, store procedures, views and cursors created.
- The tester should figure out the tables which get affected when insert update and delete (DML) operations are performed through the web or desktop applications.

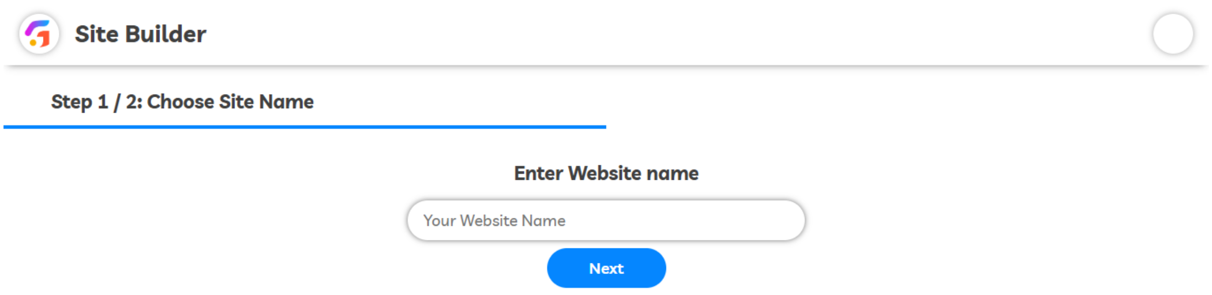
9. SCREEN SHOTS

Step 1: Open the project location. Page will be redirected to login page.

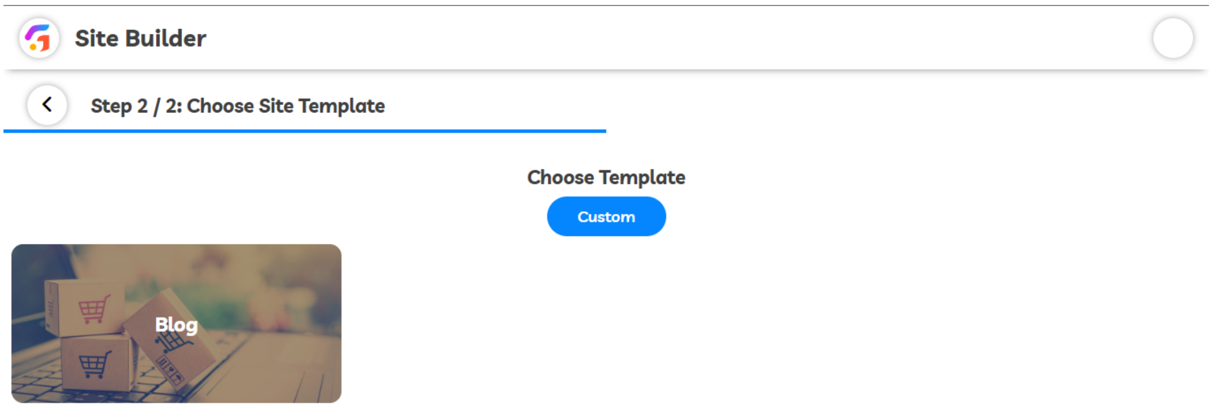


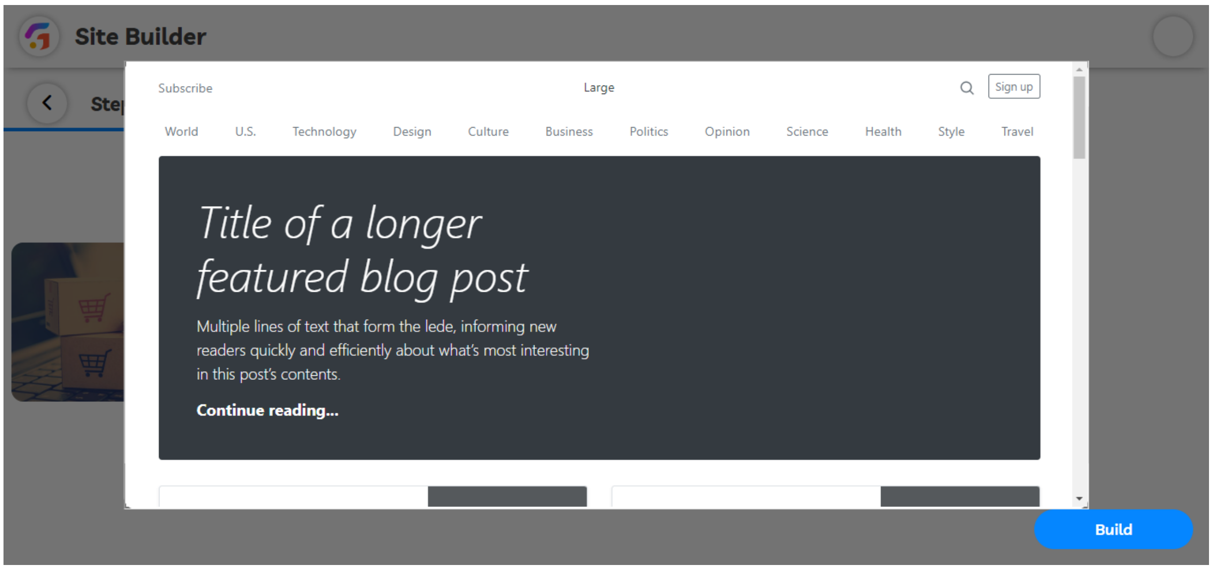
Step 2: Enter the credentials and login. You will be redirected to setup wizard.

Step 3: Type the site name and submit.



Step 4: After choosing site name. Choose the template.



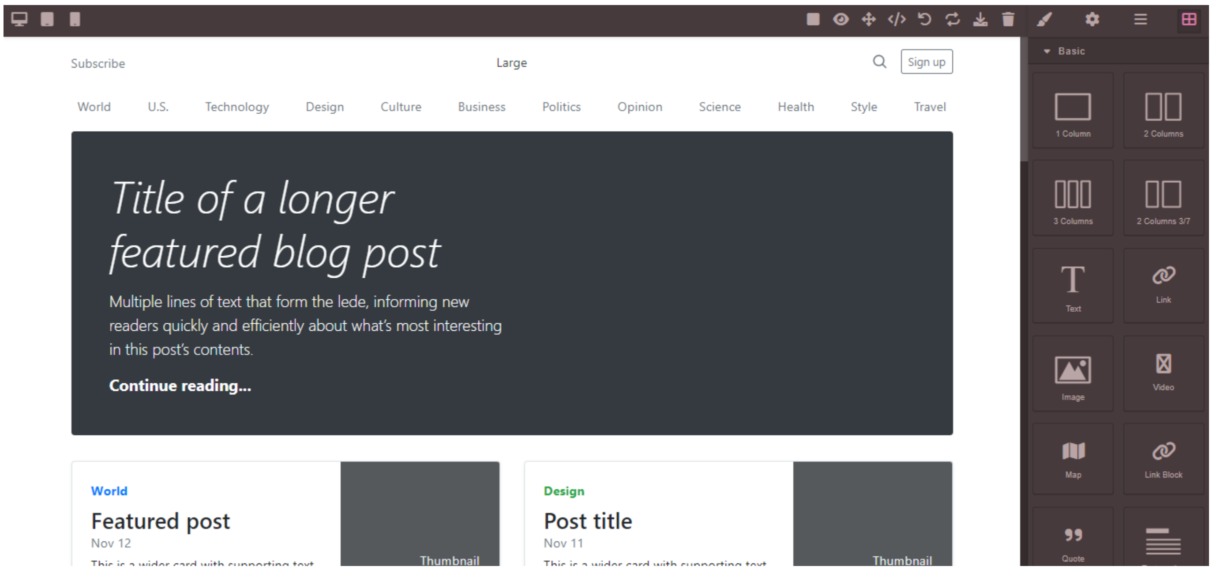


Title of a longer featured blog post

Multiple lines of text that form the lede, informing new readers quickly and efficiently about what's most interesting in this post's contents.

Continue reading...

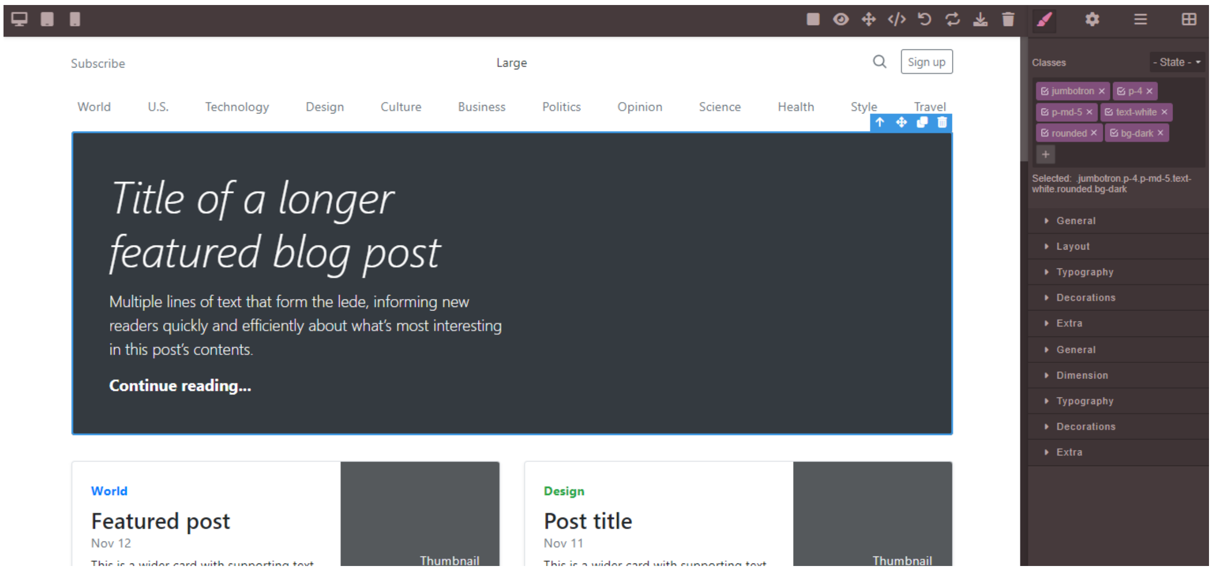
Step 5: After choosing a template system will redirect you to builder module.



Title of a longer featured blog post

Multiple lines of text that form the lede, informing new readers quickly and efficiently about what's most interesting in this post's contents.

Continue reading...



Title of a longer featured blog post

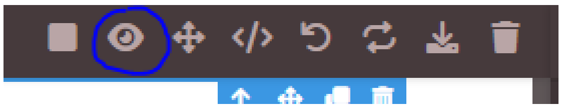
Multiple lines of text that form the lede, informing new readers quickly and efficiently about what's most interesting in this post's contents.

Continue reading...

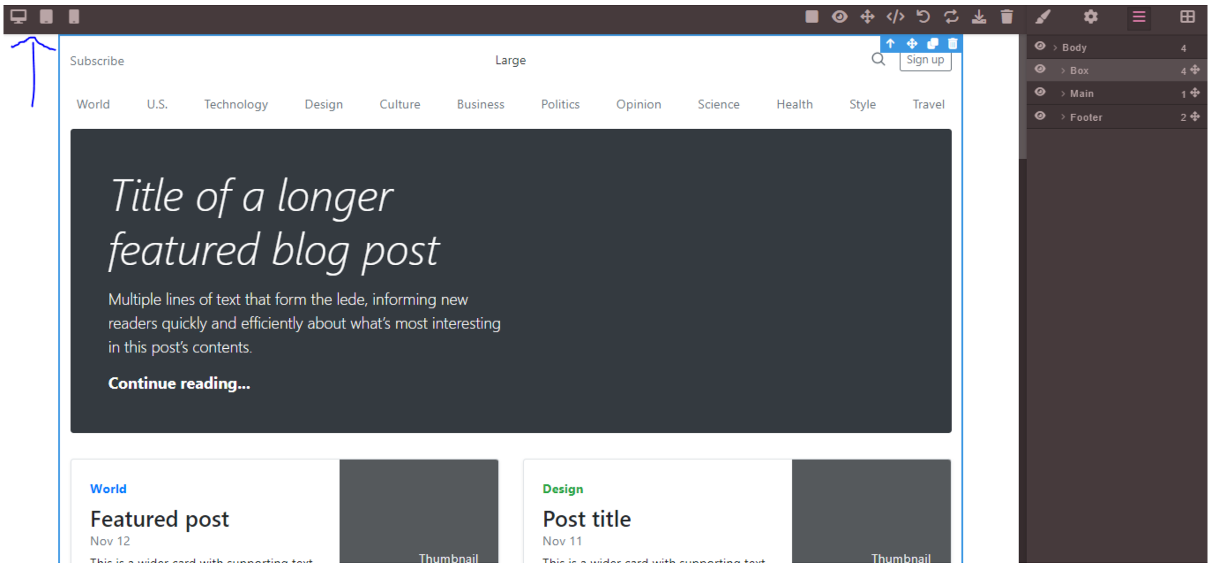
jumbotron ✕ p-4 ✕ p-md-5 ✕ text-white ✕ rounded ✕ bg-dark ✕

Selected: .jumbotron.p-4.p-md-5.text-white.rounded.bg-dark

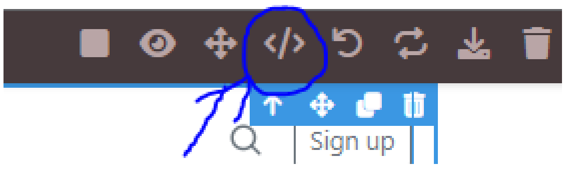
Step 6: After designing the page click on preview button.



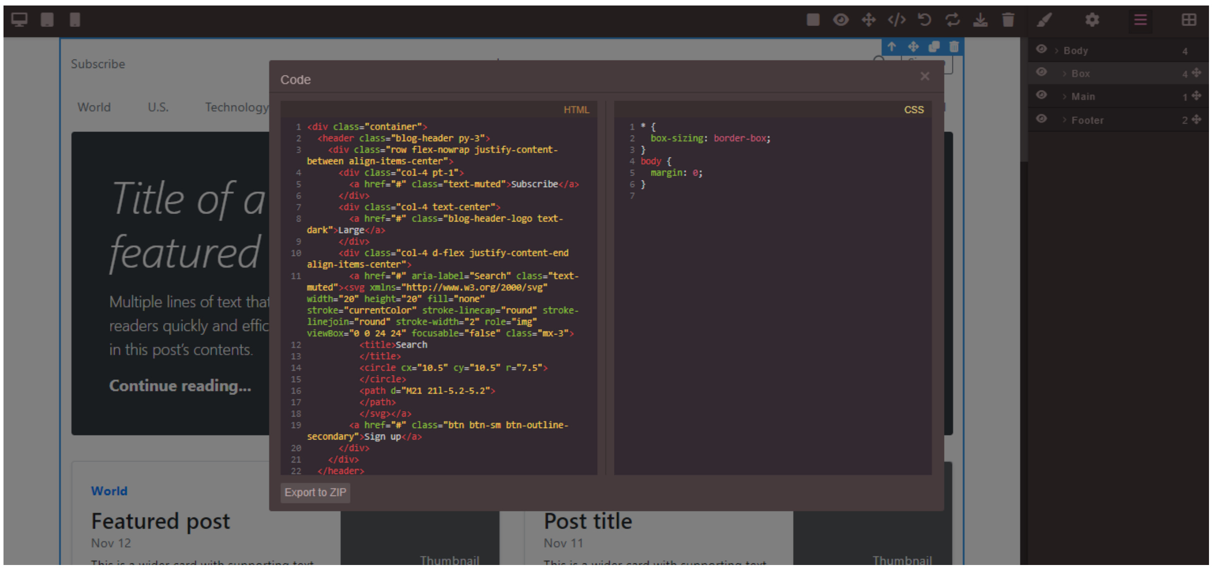
To change the views to mobile, tablet, desktop views. Click on devices in left side.



Step 7: After viewing everything click on codes "</>" icon to export the code.



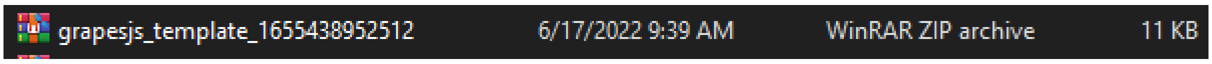
Now click "Export ZIP" to download the code.



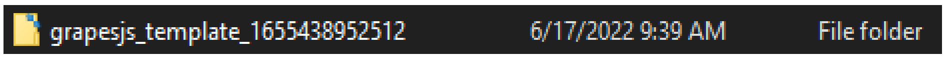
```html
<div class="container">
  <header class="blog-header py-3">
    <div class="row flex-nowrap justify-content-
between align-items-center">
      <div class="col-4 pt-1">
        <a href="#" class="text-muted">Subscribe</a>
      </div>
      <div class="col-4 text-center">
        <a href="#" class="blog-header-logo text-
dark">Large</a>
      </div>
      <div class="col-4 d-flex justify-content-end
align-items-center">
        <a href="#" aria-label="Search" class="text-
muted"><svg xmlns="http://www.w3.org/2000/svg"
width="20" height="20" fill="none"
stroke="currentColor" stroke-linecap="round" stroke-
linejoin="round" stroke-width="2" role="img"
viewBox="0 0 24 24" focusable="false" class="mx-3">
          <title>Search
          </title>
          <circle cx="10.5" cy="10.5" r="7.5">
          </circle>
          <path d="M21 21l-5.2-5.2">
          </path>
          </svg></a>
        <a href="#" class="btn btn-sm btn-outline-
secondary">Sign up</a>
      </div>
    </div>
  </header>
```

```css
* {
  box-sizing: border-box;
}
body {
  margin: 0;
}
```

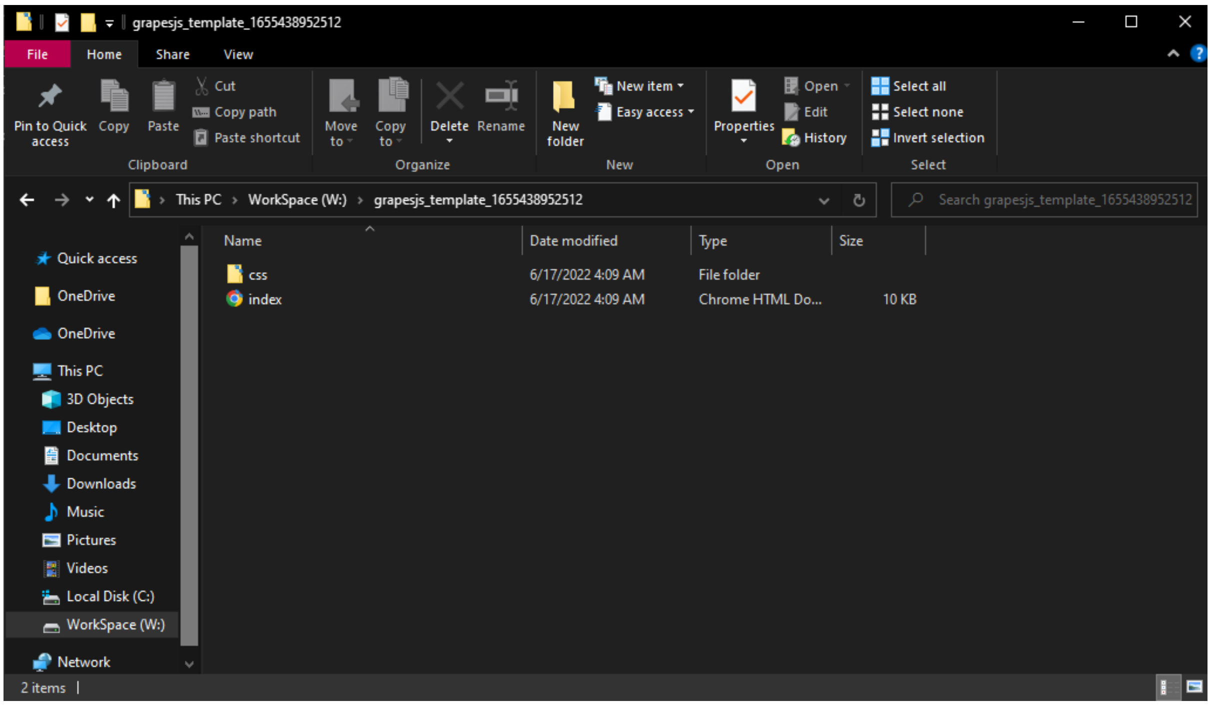
Step 8: Now open the download location and export the file. All the sources will be available in it.



grapesjs_template_1655438952512 6/17/2022 9:39 AM WinRAR ZIP archive 11 KB



grapesjs_template_1655438952512 6/17/2022 9:39 AM File folder



This PC > WorkSpace (W:) > grapesjs_template_1655438952512

Name	Date modified	Type	Size
css	6/17/2022 4:09 AM	File folder	
index	6/17/2022 4:09 AM	Chrome HTML Do...	10 KB

2 items

10. COST ESTIMATION

MATERIAL	**COST**	**TOTAL**
BROWSING	500	2000
TRAINING	500	2000
REPORT PREPARATION	250	1000
	TOTAL	5000

11. CONCLUSION

This is now easy to implement as most of the mobile phones, tablets, laptops and desktops today have the required resolution in order to render the webpage. The project offers responsive and easy to navigate webpages. User doesn’t have to know any coding languages to build web pages. With the help of this system user can work on main functionality required in their web page.

12. BIBLOGRAPHY

11.1 References

- From Wikipedia, “HTML”, “CSS”, “JAVA SCRIPT”, “PHP”, “MYSQL”
    - **https://en.wikipedia.org/wiki/HTML**
    - **https://en.wikipedia.org/wiki/CSS**
    - **https://en.wikipedia.org/wiki/JAVASCRIPT**
    - **https://en.wikipedia.org/wiki/PHP**
    - **https://en.wikipedia.org/wiki/MYSQL**
- M.T. Hoogvliet, "SaaS Interface Design", presented at Rotterdam University, 2008.
- From Wikipedia, "On-demand Pricing", http://en.wikipedia.org/wiki/On-demand.
- Christian Wenz, Essential Silverlight 2 Up-to-Date, O'Reilly, 2008.
    - http://www.Adobe.com/Dreamweaver
- From Wikipedia, "Microsoft Expression Web",
    - http://en.wikipedia.org/wiki/Microsoft Expression_Web.
- Jeff Scanlon, Accelerated Silverlight 2, Apress, 2008.
- Brennon Williams, Microsoft Expression Blend UNLEASHED, SAMS, 2008.
- Matthew MacDonald. Pro Silverlight 2 in C# 2008, Apress, 2008.

13. USER MANUAL

Welcome to WEB PAGE BUILDER, here you can know full specification about the system.

These user’s manual guide you to use the application.

- Open the application.
- Register an account and login to your account.
- Choose a name for your webpage.
- Choose a predesigned templates or choose custom design.
- Edit the texts, images, etc.
- Preview the webpage.
- Export the code.
- Extract the code and run in your browser.