INTERVIEW QUESTION OF CODEIGNITER

1. What is CodeIgniter?

Answer:
CodeIgniter is an open-source PHP framework used to develop dynamic web applications. It follows the **Model-View-Controller (MVC)** architectural pattern, which helps in separating the logic from the presentation. CodeIgniter is lightweight, fast, and easy to set up, with a rich set of built-in libraries and helpers that assist in common tasks like form validation, database interactions, and session management.

---

2. Explain the MVC architecture in CodeIgniter.

Answer:
The **Model-View-Controller (MVC)** architecture in CodeIgniter is used to separate the application's logic, data, and user interface.

- **Model**: Manages the data and the business logic. It interacts with the database and provides the necessary information to the controller.
- **View**: The user interface of the application, which displays the data from the model. It is the part that the user interacts with.
- **Controller**: Acts as a middleman that processes requests, interacts with the model to fetch data, and loads views for display.

---

3. How does CodeIgniter handle URL routing?

Answer:
CodeIgniter uses a **routing system** that maps a URL request to a specific controller and method. The routing configuration is handled in the application/config/routes.php file. By default, the URL structure is like this:
http://your-domain/controller/method/parameters

You can define custom routes to make the URLs more user-friendly or to map URLs to specific controllers.

For example, a default route in routes.php might look like:

Php

```
$route['default_controller'] = 'welcome';
```

---

4. What is the purpose of helpers in CodeIgniter?

Answer:
Helpers in CodeIgniter are a collection of functions designed to assist with common tasks. They provide a way to simplify and streamline tasks such as URL creation, form handling, and working with arrays or strings. You can load a helper using $this->load->helper('helper_name');.

Some common helpers include:

- url_helper: Functions for generating URLs.
- form_helper: Functions for creating HTML forms.
- file_helper: Functions for file operations.

---

5. What is the difference between load->view() and load->library() in CodeIgniter?

Answer:

- **$this->load->view()**: It is used to load a **view** (the user interface) in the application. Views are the HTML files that display the data. Example:

  php
  ```
  $this->load->view('welcome_message');
  ```

- **$this->load->library()**: It is used to load **libraries** that provide functionality for specific tasks like session handling, form validation, or database interaction. Example:

  php
  ```
  $this->load->library('session');
  ```

---

6. What are CodeIgniter hooks?

Answer:
Hooks in CodeIgniter are special points in the framework where developers can plug in custom code to modify its behavior. Hooks allow you to run code before or after certain

system events, such as controller execution, page output, or database query execution, without modifying the core files.

To use hooks, you must enable them in the application/config/hooks.php file. Example of a hook:

```
php
$hook['pre_controller'][] = array(
    'class'    => 'MyHookClass',
    'function' => 'some_function',
    'filename' => 'myhook.php',
    'filepath' => 'hooks'
);
```

---

7. How do you perform database operations in CodeIgniter?

Answer:
CodeIgniter provides a **Database Class** that simplifies database operations. You can use it to interact with the database using Active Record, which helps with SQL queries.

Example of basic database operations:

- **Loading the database**:

```
php

$this->load->database();
```

- **Selecting data**:

```
php

$query = $this->db->get('users');

foreach ($query->result() as $row) {

    echo $row->username;

}
```

- **Inserting data**:

```
php

$data = array(
    'username' => 'JohnDoe',
    'email' => 'john@example.com'
);
$this->db->insert('users', $data);
```

---

8. What are libraries in CodeIgniter, and how do you load them?

Answer:
Libraries in CodeIgniter are pre-built classes that provide functionality for specific tasks, such as form validation, session management, email sending, and more.

- To load a library, use the following syntax:

php

```
$this->load->library('library_name');
```

- For example, to load the session library:

php

```
$this->load->library('session');
```

- You can also load a library with parameters:

php

```
$this->load->library('form_validation', $config);
```

---

9. Explain the CodeIgniter session management.

Answer:
CodeIgniter provides a session class to manage user sessions. The session data is stored in the server, either in the database or as cookies. The session class allows you to store, retrieve, and delete session data easily.

To use sessions in CodeIgniter:

1. **Load the session library**:

php

```
$this->load->library('session');
```

2. **Set session data**:

php

```
$this->session->set_userdata('username', 'JohnDoe');
```

3. **Get session data**:

php

```
$username = $this->session->userdata('username');
```

4. **Unset session data**:

php

```
$this->session->unset_userdata('username');
```

---

10. What is the base_url() function in CodeIgniter?

Answer:
The base_url() function in CodeIgniter is used to generate the full URL to your website's base directory. It is helpful when linking assets like images, CSS, or JavaScript, ensuring the paths are correct regardless of the environment.

For example, to load an image:

php

```
<img src="<?php echo base_url('assets/images/logo.png'); ?>" alt="Logo">
```

The base_url() function can be configured in the application/config/config.php file by setting the $config['base_url'].