

Web-программирование

Место №5

Кочетов С.В.

09.03.04.

УБМ-3, 3 курс

Место, №5
1-28. Модульный контроль
Вопросы с выбором

1. б. В) Фреймворк для разрабо-
тки на PHP

2. с. С) laravel new

3. а. А) config/database.php

4. а. А) php artisan serve

5. с. С) Инструмент для ав-
томатизации задач

6. ~~а. А)~~ { { { }

7. б. В) Инструменты для работы
с БД и создания модели

8. б. В) response() → json()

9. а. А) php artisan migrate

10. с. С) Это пути, которые
определяют, как обрабаты-
ваются HTTP-запросы;

11. а. А) Blade

12. c. C) php artisan make:resource

13. b. B) composer create-project --prefer-dist-laravel/laravel

14. b. B) Это библиотека для работы с БД;

15. a. A) Auth::user()

16. b. B) Библиотека для создания пользовательских интерфейсов;

17. a. A) Vue.component()

18. a. A) props

19. b. B) Архитектурный стиль для создания веб-сервисов;

20. b. B) POST

21. b. B) Веб-приложение, которое использует современные веб-технологии для предоставления пользовательского

опыта, автоматизированное взаимодействие;

22. а. А) service-worker.js

23. б. В) CSS-стили

24. а. А) functions.php

25. д. Д) register_activation_hook()

26. б. В) Менеджер зависимостей для PHP

27. б. В) composer require package

28. а. А) composer.json

Вопросы на составление

39.

1 В

2 А

3 С

4 Д

5 Е

40.

1 А

1 A

2 E

3 B

4 C

5 D

29.-38. Открытые вопросы

29.

Dmbern 29

30.

Dmbern 30

31.

Dmbern 31

32.

Dmbern 32

33.

Dmbern 33

34.

Dmbern 34

35.

Dmbern 35

36. Amber 36

37. Amber 37

38. Amber 38

Бед

Мечта 15

Колосов С.В.

3 куст

Middleware в Laravel — это механизм, который позволяет фильтровать HTTP-запросы, выполняемые вашим приложением. Middleware может использоваться для выполнения различных задач, таких как аутентификация, логирование, проверка прав доступа и т.д.

Пример использования middleware:

// Создание middleware

```
php artisan make:middleware CheckAge
```

// Регистрация middleware в ядре приложения (app/Http/Kernel.php)

```
protected $routeMiddleware = [  
    'check.age' => \App\Http\Middleware\CheckAge::class,  
];
```

// Применение middleware к маршруту

```
Route::get('user/profile', function () {  
    //  
})->middleware('check.age');
```

Anbern 29

- GET-запросы используются для получения данных с сервера. Они кэшируются, могут быть закладками и сохраняются в истории браузера. Данные передаются через URL.
- POST-запросы используются для отправки данных на сервер. Они не кэшируются, не могут быть закладками и не сохраняются в истории браузера. Данные передаются в теле запроса.

Пример маршрутов:

```
Route::get('/user', 'UserController@index');
```

```
Route::post('/user', 'UserController@store');
```

Amber 30

Контроллеры в Laravel используются для группировки логики обработки запросов. Они помогают организовать код и сделать его более читаемым и поддерживаемым.

Пример контроллера:

// Создание контроллера

```
php artisan make:controller UserController
```

// Пример метода в контроллере

```
public function index()
{
    $users = User::all();
    return view('users.index', compact('users'));
}
```

Проблем 31

Связь "один ко многим" устанавливается, когда одна модель имеет множество связанных моделей.

Пример:

// Модель Post

```
class Post extends Model
{
    public function comments()
    {
        return $this->hasMany(Comment::class);
    }
}
```

// Модель Comment

```
class Comment extends Model
{
    public function post()
    {
        return $this->belongsTo(Post::class);
    }
}
```

Пример 32

Dependency Injection — это паттерн, который позволяет внедрять зависимости в класс через конструктор или методы. В Laravel это используется для упрощения тестирования и управления зависимостями.

Пример:

// Внедрение зависимости в контроллер

```
public function __construct(UserRepository $users)
{
    $this->users = $users;
}
```

Имверн 33

Миграции используются для управления структурой базы данных.

Пример создания и применения миграции:

// Создание миграции

```
php artisan make:migration create_users_table
```

// Пример миграции

```
public function up()
```

```
{
```

```
    Schema::create('users', function (Blueprint $table) {
```

```
        $table->id();
```

```
        $table->string('name');
```

```
        $table->string('email')->unique();
```

```
        $table->timestamps();
```

```
    });
```

```
}
```

// Применение миграции

```
php artisan migrate
```

Домашнее 34

Валидация данных в Laravel может быть выполнена с помощью встроенных правил валидации.

Пример:

```
public function store(Request $request)
{
    $validated = $request->validate([
        'name' => 'required|max:255',
        'email' => 'required|email|unique:users',
    ]);
    // Сохранение данных
}
```

Итого 35

События и слушатели позволяют реализовать асинхронную обработку задач.

Пример:

// Создание события

```
php artisan make:event UserRegistered
```

// Создание слушателя

```
php artisan make:listener SendWelcomeEmail --event=UserRegistered
```

// Регистрация слушателя в EventServiceProvider

```
protected $listen = [  
    'App\Events\UserRegistered' => [  
        'App\Listeners\SendWelcomeEmail',  
    ],  
];
```

Unben 36

помощью встроенных механизмов.

Основные шаги:

1. Установка пакета аутентификации: `composer require laravel/ui`
2. Генерация аутентификационных маршрутов и представлений: `php artisan ui vue --auth`
3. Настройка базы данных и миграций.
4. Регистрация маршрутов аутентификации в `web.php`.

Инверн 37

Пагинация позволяет разбивать большие наборы данных на страницы.

Пример использования пагинации в Eloquent:

// Получение данных с пагинацией

```
$users = User::paginate(15);
```

// Отображение данных в представлении

```
return view('users.index', compact('users'));
```

// В представлении

```
{{ $users->links() }}
```

Don't know 38