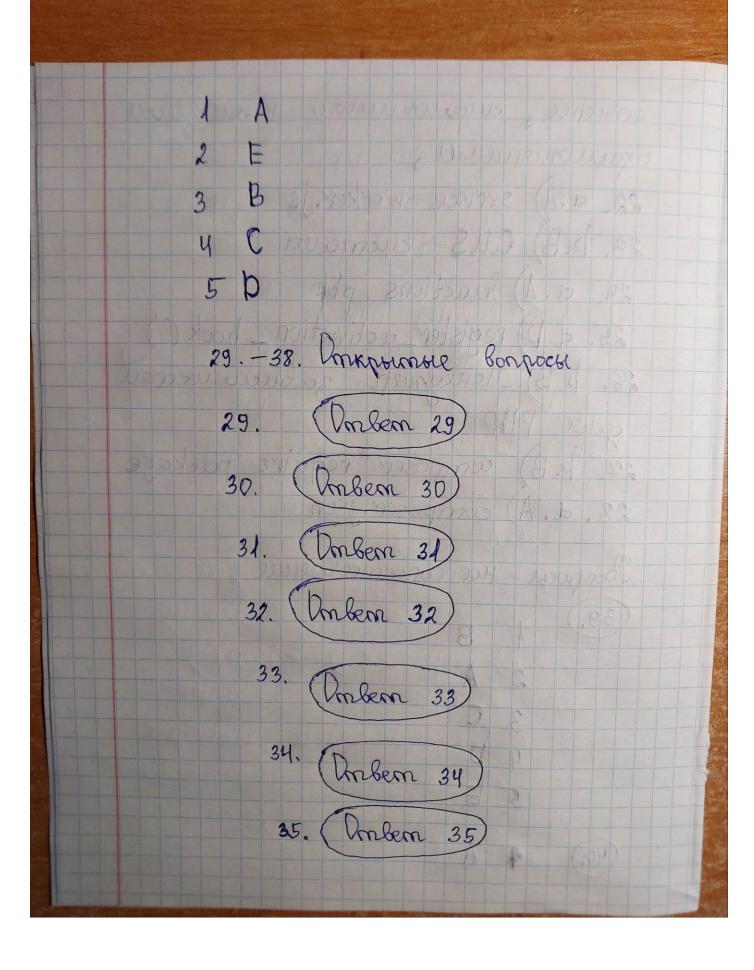
Web-norpadialupobature
Mecon w5
Konocob C.B. 09.03.04. UBM -3, 3 kypc

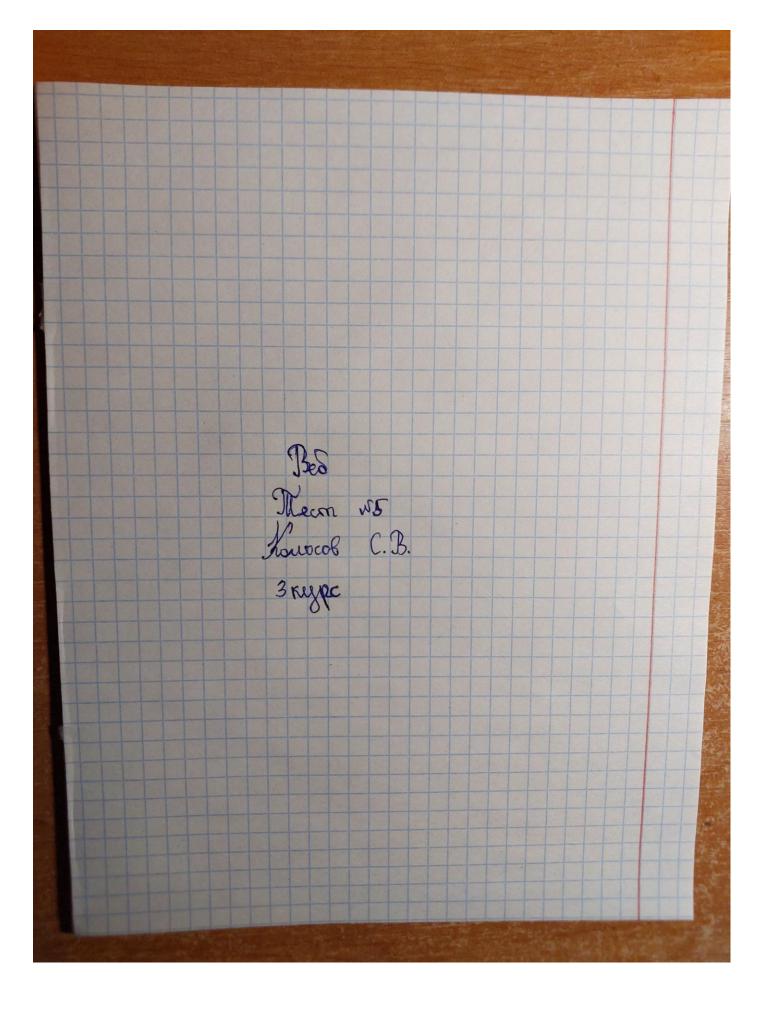
1-28. Bonpose C Berdopour 1. b. B) Preienbork gua paspado man Ha PHP 2. c. C) laravel new 3. a. A) config/database. pht 4. a. A) php artisan serve 5. c. C) Unconfiguretion gila abmanamuzanjun zargan 6. 3 a. A) 2533 7. b.B) Uranpymetionel que paromer c BD u cozganua morany 8. b. B) response () -> json() 9. a. A) php artisan migrate 10. c.C) The nyme , Komppel onledemarour, kork appagamenga-11. a. A) Blade

12. c. C) php artisan make: resource 13. b. B) composer create-projectpreser-dist-laravel/laravel 14. b. B) Imo Sudwiemera gua passoner c 50; 15. a. A) Auth: user () 16. b.B) Budemornera gua cozgourne nous observent chuse unappercol; 17. a. A) Vue composent () 18. a.A) props 19. b. B) Apxuenekrnypuser comme gua cozganua bed-cepbucob; 20. b.B) POST 21. b. B) Bed-repulsamence, Korro ! poe ucnoutzyern cobpenierrent led-mexicanner qua negoc maburens noutzobamentickoro

ontima, anautominero comultical squaltmerusul; 22. a.A) service-worker, js 23. b.B) Cells-cumerica 24. a. A) functions pho 25. d. D) register_activation_hook() 26. b. B) Meregnier zabucuencoren gue PHP 24. p. B) composer require package 28. d. A) composer. json Bonpoer na conocinaberence (39.) 1 B 2 A 3 C



	6	
36.	Ombern 3	56)
34.	Ombern	34
38.	Anlem	38)



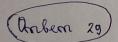
Middleware в Laravel — это механизм, который позволяет фильтровать HTTP-запросы, выполняемые вашим приложением. Middleware может использоваться для выполнения различных задач, таких как аутентификация, логирование, проверка прав доступа и т.д.

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');
```

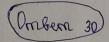


- GET-запросы используются для получения данных с сервера. Они кэшируются, могут быть закладками и сохраняются в истории браузера. Данные передаются через URL.
- POST-запросы используются для отправки данных на сервер. Они не кэшируются, не могут быть закладками и не сохраняются в истории браузера. Данные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

Пример маршрутов:

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

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



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'));
}
```

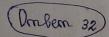
Ornbern 31

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);
    }
}
```



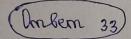
Dependency Injection — это паттерн, который позволяет внедрять зависимости в класс через конструктор или методы. В Laravel это используется для упрощения тестирования и управления зависимостями.

Пример:

```
// Внедрение зависимости в контроллер

public function __construct(UserRepository $users)

{
    $this->users = $users;
}
```



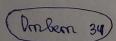
```
Миграции используются для управления структурой базы данных.
```

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();
        });
}

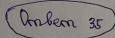
// Применение миграции
php artisan migrate
```



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
```



События и слушатели позволяют реализовать асинхронную обработку задач.

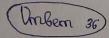
php artisan make:listener SendWelcomeEmail --event=UserRegistered

Пример:

```
// Создание события php artisan make:event UserRegistered
```

// Создание слушателя

```
// Регистрация слушателя в EventServiceProvider
protected $listen = [
    'App\Events\UserRegistered' => [
        'App\Listeners\SendWelcomeEmail',
        ],
    ];
```



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.

Anbern 32

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() }}
```

Ombern 38)