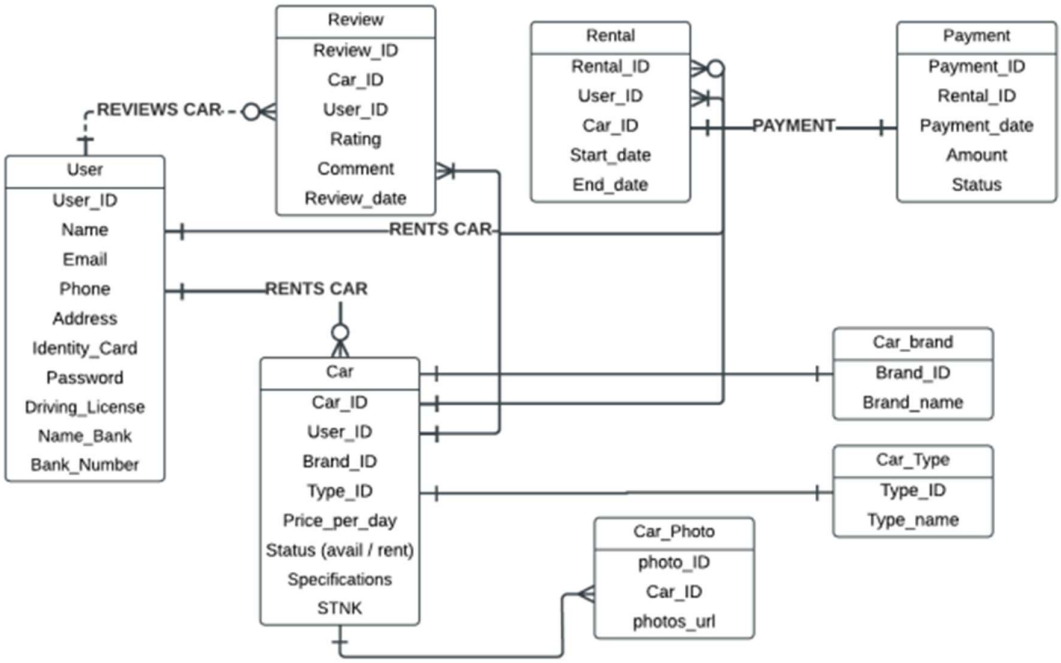
SSIP PROJECT REPORT

Gifari Anugrah Putra Madia	-	(001202400182)	-	(Leader)
Gamma Ahmad Zaki Kurnia Budihardjo	-	(001202400011)	-	(Member)
Gideon A. Siagian	-	(001202400074)	-	(Member)

1. **Information of The Table**
   a. **The ERD (To provide a clear representation of the relationships between tables)**



b. Information of All Table (Using The Laravel Model)

- **Car.php**

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Relations\BelongsTo;
use Illuminate\Database\Eloquent\Relations\HasMany;

class Car extends Model
{
    public $timestamps = false;

    use HasFactory;

    protected $primaryKey = 'car_id';

    protected $fillable = ['user_id', 'brand_id', 'type_id', 'price_per_day', 'status', 'specifications', 'stnk'];

    protected $appends = ['status'];

    public function getStatusAttribute(): string
    {
        $now = now();

        $hasActiveRental = $this->rentals()
            ->where('start_date', '<=', $now)
            ->where('end_date', '>=', $now)
            ->exists();

        return $hasActiveRental ? 'rent' : 'avail';
    }

    public function owner(): BelongsTo
    {
        return $this->belongsTo(User::class, 'user_id');
    }

    public function brand(): BelongsTo
    {
        return $this->belongsTo(CarBrand::class, 'brand_id');
    }

    public function type(): BelongsTo
    {
        return $this->belongsTo(CarType::class, 'type_id');
    }

    public function photos(): HasMany
    {
        return $this->hasMany(CarPhoto::class, 'car_id');
    }

    public function reviews(): HasMany
    {
        return $this->hasMany(Review::class, 'car_id');
    }
}
```

- **CarBrand.php**

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Relations\HasMany;

class CarBrand extends Model
{
    public $timestamps = false;

    use HasFactory;

    protected $primaryKey = 'brand_id';

    protected $fillable = ['brand_name'];

    public function cars(): HasMany
    {
        return $this->hasMany(Car::class, 'brand_id');
    }
}
```

- **CarPhoto.php**

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Relations\BelongsTo;

class CarPhoto extends Model
{
    public $timestamps = false;

    use HasFactory;

    protected $primaryKey = 'photo_id';

    protected $fillable = ['car_id', 'photos_url'];

    public function car(): BelongsTo
    {
        return $this->belongsTo(Car::class, 'car_id');
    }
}
```

- **CarType.php**

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Relations\HasMany;

class CarType extends Model
{
    public $timestamps = false;

    use HasFactory;

    protected $primaryKey = 'type_id';

    protected $fillable = ['type_name'];

    public function cars(): HasMany
    {
        return $this->hasMany(Car::class, 'type_id');
    }
}
```

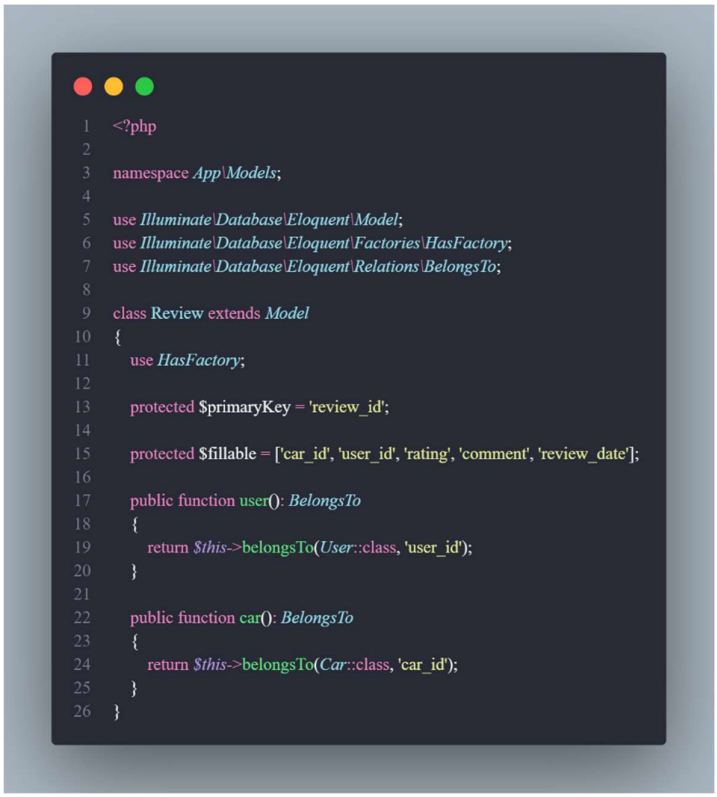
- **Payment.php**

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Relations\BelongsTo;

class Payment extends Model
{
    use HasFactory;

    protected $primaryKey = 'payment_id';

    protected $fillable = ['rental_id', 'payment_date', 'amount', 'status'];

    public function rental(): BelongsTo
    {
        return $this->belongsTo(Rental::class, 'rental_id');
    }
}
```

- **Rental.php**

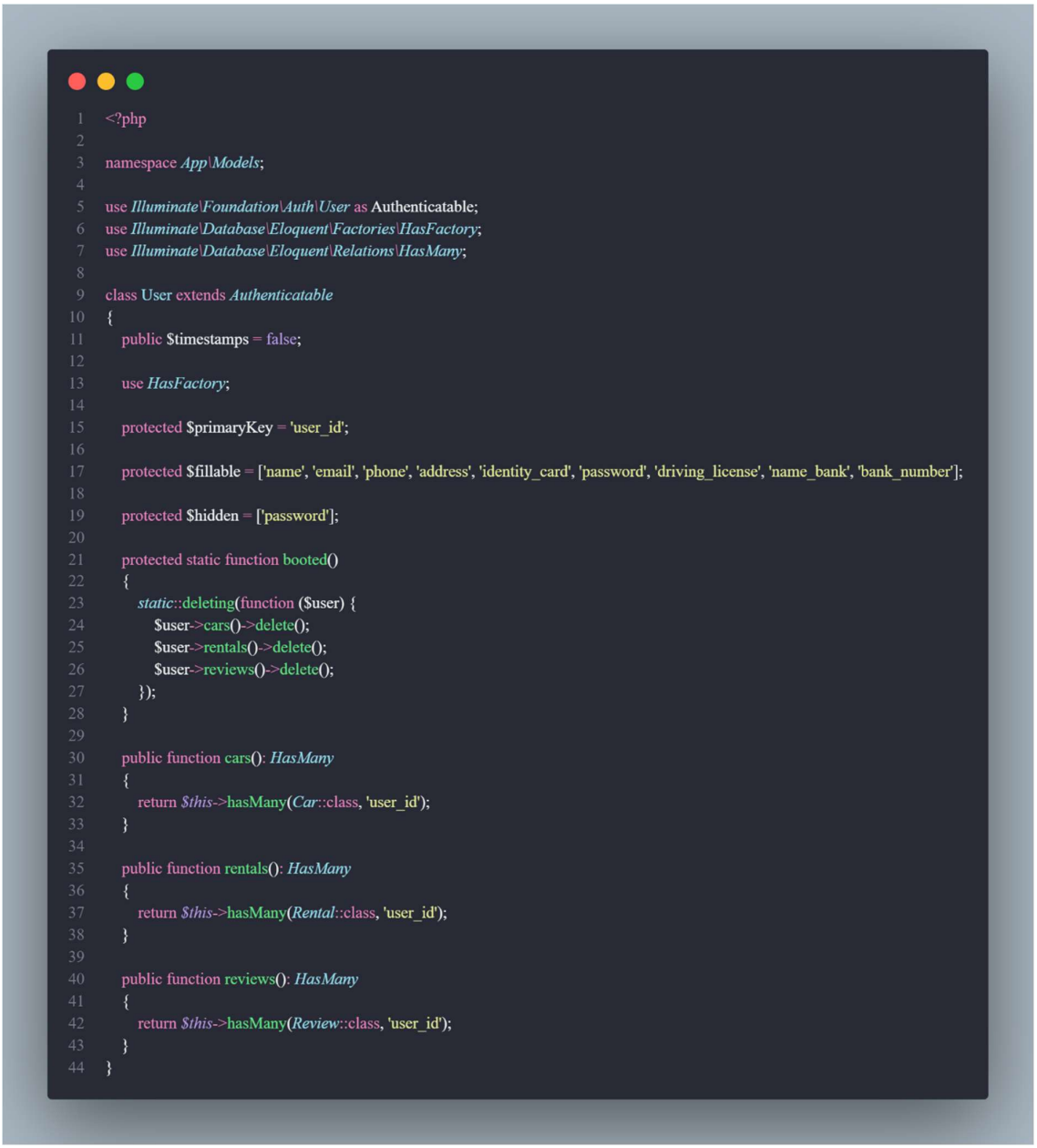
```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Relations\BelongsTo;
use Illuminate\Database\Eloquent\Relations\HasOne;

class Rental extends Model
{
    use HasFactory;

    protected $primaryKey = 'rental_id';

    protected $fillable = ['user_id', 'car_id', 'start_date', 'end_date'];

    public function user(): BelongsTo
    {
        return $this->belongsTo(User::class, 'user_id');
    }

    public function car(): BelongsTo
    {
        return $this->belongsTo(Car::class, 'car_id');
    }

    public function payment(): HasOne
    {
        return $this->hasOne(Payment::class, 'rental_id');
    }
}
```

- **Review.php**



```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Relations\BelongsTo;

class Review extends Model
{
    use HasFactory;

    protected $primaryKey = 'review_id';

    protected $fillable = ['car_id', 'user_id', 'rating', 'comment', 'review_date'];

    public function user(): BelongsTo
    {
        return $this->belongsTo(User::class, 'user_id');
    }

    public function car(): BelongsTo
    {
        return $this->belongsTo(Car::class, 'car_id');
    }
}
```

- **User.php**



```php
<?php

namespace App\Models;

use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Relations\HasMany;

class User extends Authenticatable
{
    public $timestamps = false;

    use HasFactory;

    protected $primaryKey = 'user_id';

    protected $fillable = ['name', 'email', 'phone', 'address', 'identity_card', 'password', 'driving_license', 'name_bank', 'bank_number'];

    protected $hidden = ['password'];

    protected static function booted()
    {
        static::deleting(function ($user) {
            $user->cars()->delete();
            $user->rentals()->delete();
            $user->reviews()->delete();
        });
    }

    public function cars(): HasMany
    {
        return $this->hasMany(Car::class, 'user_id');
    }

    public function rentals(): HasMany
    {
        return $this->hasMany(Rental::class, 'user_id');
    }

    public function reviews(): HasMany
    {
        return $this->hasMany(Review::class, 'user_id');
    }
}
```

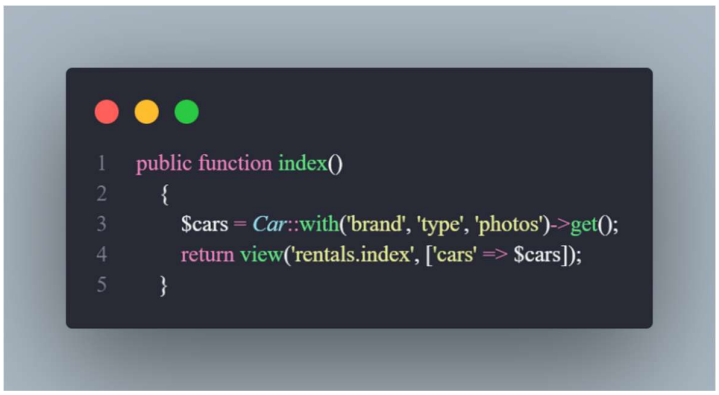
2. **The Flow of Application (Step by Step)**
    a. Start, Render the landing page.
    b. The user click **Book Now** button to Navigate to the SignIn/SignUp Form.
    c. After Navigate to the SignIn/SignUp Form, The User must sign up first before signing in.
    d. After completing the SignUp process, the user can proceed to SignIn.
    e. After signup, the user was required to complete the remaining data to finalize the verification process.
    f. Next, the user can add their car for rental through the 'Manage Rentals' section on the sidebar. In this stage, user will go through several steps.
        - First, they will add a car, which redirects the to the Create Form(rentals.create).
        - Once the form successfully submitted, the user will be redirected back to the Manage Rentals Page (rentals.index)
        - There, the newly added car will be displayed.
        - At this point, the user can either update or delete their car information. To do so, they can click on their car's card, which opens the Update and Delete Forms located in the same View file(rentals.edit).
    g. The can rent a car by selecting 'View All Cars' from the sidebar. In this process, the user will go through several steps.
        - First, they click on a car card listed by another user. This action will redirect them to the car details page, where they can view information such as images, rental price, and other specifications.
        - Next, the user can click the 'Book' button to proceed to the car rental process.
        - At this stage, the user will fill out a form, including the rental duration. The system will automatically calculate the total rental cost.
        - Once completed, the user can click the 'Pay' button, which will redirect them to the Review page.
    h. Finally, the user can leave a review for the cars they have rented through the 'Review' section on the sidebar.
        - In this section, all the cars previously rented by the user will be displayed.
        - To submit a review, the user can click on any available car card. This will redirect them to the review form.
        - After successfully submitting the review, the user will be redirected back to the Review Page (review.index).
    i. End

Notes: We have included a full simulation demonstrating how the application works.

3. **The CRUD Processes (Using The Laravel Controller) – Using The RentalController for Example**
   a. **Create**
      - **Store()**

```php
public function store(Request $request)
    {
        $validated = $request->validate([
            'brand_name' => 'required|string',
            'car_type' => 'required|string',
            'price_per_day' => 'required|numeric|min:0',
            'specifications' => 'nullable|string',
            'stnk' => 'required|file|mimes:pdf|max:2048',
            'photos_url.*' => 'image|mimes:jpeg,png,jpg|max:2048'
        ]);

        $user_id = Auth::id();

        $stnkPath = $request->file('stnk')->store('stnk', 'public');

        $brand = CarBrand::firstOrCreate(['brand_name' => $validated['brand_name']]);

        $carType = CarType::firstOrCreate(['type_name' => $validated['car_type']]);


        $car = Car::create([
            'user_id' => $user_id,
            'brand_id' => $brand->brand_id,
            'type_id' => $carType->type_id,
            'price_per_day' => $validated['price_per_day'],
            'status' => 'avail',
            'specifications' => $validated['specifications'],
            'stnk' => $stnkPath,
        ]);

        if ($request->hasFile('photos_url')) {
            $files = $request->file('photos_url');
            foreach (array_slice($files, 0, 3) as $file) {
                $path = $file->store('photos_url', 'public');
                CarPhoto::create([
                    'car_id' => $car->car_id,
                    'photos_url' => $path,
                ]);
            }
        }

        return redirect()->route('rentals.index')->with('success', 'Mobil berhasil ditambahkan!');
    }
```

   b. **Read / Select**
      - **Index()**



```php
public function index()
    {
        $cars = Car::with('brand', 'type', 'photos')->get();
        return view('rentals.index', ['cars' => $cars]);
    }
```

- **Edit($car_id)**

```php
public function edit($car_id)
    {
        $car = Car::findOrFail($car_id);
        return view('rentals.edit', compact('car'));
    }
```

c. Update

```php
public function update(Request $request, $car_id)
{
    $car = Car::findOrFail($car_id);

    $validated = $request->validate([
        'brand_name' => 'required|string',
        'car_type' => 'required|string',
        'price_per_day' => 'required|numeric|min:0',
        'specifications' => 'nullable|string',
        'stnk' => 'nullable|file|mimes:pdf|max:2048',
        'photos_url.*' => 'nullable|image|mimes:jpeg,png,jpg|max:2048',
    ]);


    $brand = CarBrand::firstOrCreate(['brand_name' => $validated['brand_name']]);

    $carType = CarType::firstOrCreate(['type_name' => $validated['car_type']]);

    if ($request->hasFile('stnk')) {
        if ($car->stnk && \Storage::disk('public')->exists($car->stnk)) {
            \Storage::disk('public')->delete($car->stnk);
        }

        $stnkPath = $request->file('stnk')->store('stnk', 'public');
        $car->stnk = $stnkPath;
    }

    // Update data mobil
    $car->brand_id = $brand->brand_id;
    $car->type_id = $carType->type_id;
    $car->price_per_day = $validated['price_per_day'];
    $car->specifications = $validated['specifications'];
    $car->save();


    if ($request->hasFile('photos_url')) {

        $files = $request->file('photos_url');
        foreach (array_slice($files, 0, 3) as $file) {
            $path = $file->store('photos_url', 'public');
            CarPhoto::create([
                'car_id' => $car->car_id,
                'photos_url' => $path,
            ]);
        }
    }

    return redirect()->route('rentals.index')->with('success', 'Mobil berhasil diperbarui!');
}
```

d. Delete

```php
public function destroy($car_id)
    {
        $car = Car::findOrFail($car_id);

        if ($car->stnk && \Storage::disk('public')->exists($car->stnk)) {
            \Storage::disk('public')->delete($car->stnk);
        }

        foreach ($car->photos as $photo) {
            if ($photo->photos_url && \Storage::disk('public')->exists($photo->photos_url)) {
                \Storage::disk('public')->delete($photo->photos_url);
            }
            $photo->delete();
        }

        $car->delete();

        return redirect()->route('rentals.index')->with('success', 'Mobil berhasil dihapus!');
    }

```