Detailed Notes for BCA Students – MVC & Laravel

# 1. Introduction to MVC

MVC is a software design pattern used to make applications easier to develop and maintain.  
It divides an application into three parts:  
  
1. Model – Handles data of the application. Works with the database. Example: Student model stores student details.  
2. View – Deals with presentation (UI). Example: HTML page showing student data.  
3. Controller – Acts as a middleman. Takes user request, interacts with model, and returns a view.  
  
This separation makes apps organized and easy to manage.

# 2. Uses of MVC Framework

- Keeps application structure clean  
- Reusability of code  
- Scalable for large apps  
- Allows teamwork (designers, developers, testers)  
- Easier debugging and testing

# 3. Advantages & Disadvantages of MVC

Advantages:  
- Separation of concerns  
- Faster development  
- Scalable and reusable  
- Better testing and debugging  
  
Disadvantages:  
- Requires more files  
- Complex for small projects  
- New learners may find it difficult  
- Synchronization between components needed

# 4. Laravel – Introduction

Laravel is an open-source PHP MVC framework created by Taylor Otwell in 2011.  
  
Features:  
- MVC architecture  
- Blade template engine  
- Eloquent ORM for database  
- Routing system  
- Artisan CLI for automation  
- Security (CSRF protection, authentication)  
- Migration system for databases  
- Strong community support

# 5. How Laravel is Better than Other Frameworks

- Simple and clean syntax  
- Blade Template (separates HTML and PHP)  
- Eloquent ORM for easy database handling  
- Built-in authentication & security  
- Artisan CLI saves time  
- Huge ecosystem of packages  
- Better documentation and tutorials

# 6. Steps to Install and Configure Laravel

1. Install Composer from getcomposer.org  
2. Create Laravel project: composer create-project laravel/laravel myproject  
3. Navigate: cd myproject  
4. Run server: php artisan serve  
5. Open: http://127.0.0.1:8000

# 7. Laravel Directory Structure

- app/ → Models, Controllers, Middleware  
- bootstrap/ → Initializes app  
- config/ → Settings and configuration  
- database/ → Migrations, seeders  
- public/ → Entry point, CSS, JS, images  
- resources/ → Views (Blade), assets  
- routes/ → web.php, api.php  
- storage/ → Cache, logs, uploads  
- tests/ → Testing files  
- vendor/ → Composer packages

# 8. Routing in Laravel

Routing decides which controller or view handles a request.  
  
Example (routes/web.php):  
Route::get('/', function () {  
 return view('welcome');  
});

# 9. Route File Functions

- Route::get('/url', ...) → GET request  
- Route::post('/url', ...) → POST request  
- Route::put('/url', ...) → Update  
- Route::delete('/url', ...) → Delete  
- Route::any('/url', ...) → Any request type  
- Route::redirect('/from', '/to') → Redirect  
- Route::view('/url', 'viewName') → Load view directly

# 10. Controllers in Laravel

Creating Controller:  
php artisan make:controller StudentController  
  
Example Controller:  
class StudentController extends Controller {  
 public function index() {  
 return "Welcome to Student Controller";  
 }  
  
 public function show($id) {  
 return "Student ID: " . $id;  
 }  
  
 public function getData(Request $request) {  
 $name = $request->input('name');  
 return "Hello, " . $name;  
 }  
}  
  
Connecting Controller with Routes (web.php):  
use App\Http\Controllers\StudentController;  
  
Route::get('/student', [StudentController::class, 'index']);  
Route::get('/student/{id}', [StudentController::class, 'show']);  
Route::post('/submit', [StudentController::class, 'getData']);

# Accessing Form Data in Controller

Form (Blade view):  
<form method="POST" action="/submit">  
 @csrf  
 <input type="text" name="name">  
 <button type="submit">Submit</button>  
</form>  
  
Controller Function:  
public function getData(Request $request) {  
 $name = $request->input('name');  
 return "Welcome, " . $name;  
}

# Quick Exam-Ready Summary

- MVC = Model (data), View (UI), Controller (logic)  
- Uses of MVC → clean structure, reusable, teamwork  
- Laravel = PHP MVC framework with Blade, Eloquent, Artisan  
- Install → Composer → Create project → php artisan serve  
- Directory → app/, routes/, resources/, config/, database/, public/  
- Routes → GET, POST, PUT, DELETE, view(), redirect()  
- Controllers → make:controller, connected in routes  
- Form data → $request->input('name')