**Effort Estimation**

**1. Template-Driven Form with Validation (Completed In 3 Hours)**

**Project**: *User Registration Form*

*Features:*

* Full Name (required)
* Email (required, must be valid)
* Password (required, min 6 characters)
* Confirm Password (must match password)
* Age (custom validator: must be 18+)

*Requirements:*

* Use [(ngModel)] for two-way binding.
* Built-in validators: required, email, minlength.
* Custom validator directive:
* For confirming password.
* For age validation.
* Show user-friendly error messages.

Shape

**2. Reactive Form with Built-in & Custom Validators (3 Hours)**

**Project**: *Product Entry Form*

*Features:*

* Product Name (required)
* SKU (custom validator: must start with "SKU-")
* Price (required, must be > 0)
* Available (checkbox)
* Category (dropdown)
* Create a drop down to select (shoes,clothes)
* And on the basis of that you have to add fields
* Console the values

*Requirements:*

* Use FormBuilder, FormGroup, FormControl.
* Built-in validators: Validators.required, Validators.min.
* Custom validator for SKU format.
* Disable submit if the form is invalid.

Shape

**3. Two-Way Binding Practice ( 2 Hours 30 minutes)**

**Project**: *Live Blog Post Preview*

*Features:*

* Input fields: Blog Title and Content
* Preview shows real-time updates as user types

*Requirements:*

* Use [(ngModel)] for two-way binding.
* Single component setup.

Shape

**4. Input & Output Decorators ( 3 Hours 30 minutes)**

**Project**: *Task List with Removable Items*

*Components:*

* **ParentComponent**: Manages a list of tasks
* **ChildComponent**: Displays one task with a "Delete" button

*Requirements:*

* Pass task data using @Input().
* Emit delete event using @Output() and EventEmitter.
* Parent should remove the task from the list.

Shape

**5. Angular Service for Data Sharing ( 2 Hours)**

**Project**: *Task Manager with Service Layer*

*Features:*

* Task list managed using a service.
* Components:
* TaskListComponent: Displays list
* TaskFormComponent: Adds a new task

*Requirements:*

* Create a TaskService:
* Store tasks in an in-memory array.
* Methods: getTasks(), addTask(), deleteTask().
* Use BehaviorSubject or Subject to update components in real-time.
* Inject service in both components using constructor(private taskService: TaskService).

Shape

**6. Bonus: Combine Everything ( 6 hours )**

**Project**: *User Profile Editor*

*Features:*

* Reactive Form for editing user data.
* Preview user info in real-time (via two-way binding).
* Profile picture handled via child component:
* @Input() for current image URL
* @Output() to notify parent of image change
* Use a UserService to fetch/save profile data.

Shape