



1. Sprint Details











- **Sprint Number:** 3
 - **Sprint Pod Name:** Core Frontend Logic & Testing
 - **Pod Members:**
 - Member 1 – Aniket Das
 - Member 2 – Anish Dey
 - Member 3 – Arya Mane
 - Member 4 – Aymaan Shabbir
 - Member 5 – Bidisha Talukdar
 - Member 6 – Disha Makal
 - **Submission Date:** 04-07-2025
-

2. Sprint Goal

To build dynamic, reusable components for the EduSphere frontend using React/Angular, implement robust form handling and validation, enforce consistent code quality with ESLint, and ensure optimized bundling and dependency management using Webpack and Yarn. The sprint focuses on modular architecture, maintainable code, and test coverage to prepare the codebase for production-readiness.

3. User Stories Completion

User Story ID	Task Description	Assigned To	Status	Acceptance Criteria Met
US3-1.1	Create reusable CourseCard and	Member 1	 Done	 Yes

User Story ID	Task Description	Assigned To	Status	Acceptance Criteria Met
	InstructorCard components			
US3-2.1	Build feedback form using React Hooks or Angular Reactive Forms	Member 2	 Done	 Yes
US3-3.1	Configure ESLint, define rules, and fix code quality issues	Arya Satish Mane (M3)	 Done	 Yes
US3-4.1	Write unit tests for forms, CourseCard, InstructorCard, and filters	Member 4	 Done	 Yes
US3-5.1	Configure Webpack for optimized bundling and asset handling	Member 5	 Done	 Yes
US3-6.1	Install and configure Yarn; lock dependencies using yarn.lock	Member 6	 Done	 Yes



4. Code Submission

- **GitHub Repository Link:** [Link](#)
- **Relevant Branches or Tags:** sprint-3-final, main, eslint-config, webpack-setup, etc.
- **Key Folder Paths:**
 - /src/components/ – Reusable components


- /src/forms/ – Feedback form logic
 - /src/___tests___/ – Unit test cases
 - .eslintrc.json, webpack.config.js, yarn.lock – Tooling configs
-

✓ Included Features

- **Component-Based Architecture:**
 - Reusable <CourseCard /> and <InstructorCard /> (React) with props, styles, and hover effects
 - Easily maintainable and scalable UI structure
- **Form Handling & Validation:**
 - Feedback form using React Hooks with controlled inputs
 - Validation for required fields, character limits, email format, etc.
- **Code Quality (ESLint):**
 - ESLint integrated with TypeScript/React
 - Custom rule set applied across the codebase
 - Errors/warnings resolved and auto-fix scripts added
- **Testing:**
 - Unit tests for CourseCard, feedback form, and filters using Jest/React Testing Library
 - Code coverage > 80% for tested components
- **Webpack & Yarn Setup:**
 - Webpack configured for module bundling and asset optimization
 - Yarn used for consistent dependency resolution across environments

5. Sprint Review Document

What Was Planned vs. Delivered:

-  All planned tasks including component development, validation logic, code linting, testing, and bundling were delivered.

Challenges Faced:

- Initial ESLint compatibility issues with TypeScript JSX
- Managing test mocks for form submissions
- Configuring Webpack loaders for different file types (CSS/Images)
- Yarn dependency conflicts resolved using clean lockfile management

Learnings:

- Importance of using **component-based design** for scalability
- Gained hands-on experience with **React Hooks, ESLint, Webpack, and Jest**
- Understood how to enforce and maintain **consistent code quality** in a collaborative team setup
- Realized the value of **automated testing** and continuous integration for production-quality code

6. Retrospective Notes

What Went Well:

- Clear task distribution and ownership among all 6 members
- Code was well-formatted and easy to maintain after ESLint integration

- Testing improved confidence in key user flows
- Yarn and Webpack greatly improved project structure and speed

What Didn't Go Well:

- Webpack took extra time to configure for different file types
- Some test files required mocking complex dependencies
- A few merge conflicts occurred during integration of ESLint and components

Areas to Improve:

- Configure Husky and Prettier for auto-formatting
- Start testing earlier in the sprint
- Use a shared component library for uniformity

Actionable Learnings for Next Sprint:

- Integrate CI/CD pipeline for automated testing and linting
- Increase test coverage for all user-interactive components
- Continue improving bundle size and code performance