Student Registration Portal - Phase 6: User Interface Development

Lightning Web Components (LWC)

1. Student Registration Component

studentRegistration.html

```
<template>
    <lightning-card title="Student Registration" icon-name="standard:education">
        <div class="slds-p-horizontal_medium">

            <!-- Progress Indicator -->
            <lightning-progress-indicator
                current-step={currentStep}
                type="path"
                variant="base">
                <lightning-progress-step label="Personal Info" value="step1"></lightning-
                <lightning-progress-step label="Academic Info" value="step2"></lightning-
                <lightning-progress-step label="Application" value="step3"></lightning-pr
                <lightning-progress-step label="Confirmation" value="step4"></lightning-p
            </lightning-progress-indicator>

            <!-- Step 1: Personal Information -->
            <div if:true={isStep1} class="slds-m-top_medium">
                <div class="slds-grid slds-wrap slds-gutters">
                    <div class="slds-col slds-size_1-of-2">
                        <lightning-input
                            label="First Name"
                            value={studentData.firstName}
                            data-field="firstName"
                            onchange={handleInputChange}
                            required>
                        </lightning-input>
                    </div>
                    <div class="slds-col slds-size_1-of-2">
                        <lightning-input
                            label="Last Name"
                            value={studentData.lastName}
                            data-field="lastName"
                            onchange={handleInputChange}
                            required>
                        </lightning-input>
                    </div>
                    <div class="slds-col slds-size_1-of-2">
                        <lightning-input
                            label="Email"
                            type="email"
```

```
                    value={studentData.email}
                    data-field="email"
                    onchange={handleInputChange}
                    required>
                </lightning-input>
            </div>
            <div class="slds-col slds-size_1-of-2">
                <lightning-input
                    label="Phone"
                    type="tel"
                    value={studentData.phone}
                    data-field="phone"
                    onchange={handleInputChange}>
                </lightning-input>
            </div>
            <div class="slds-col slds-size_1-of-2">
                <lightning-input
                    label="Date of Birth"
                    type="date"
                    value={studentData.dateOfBirth}
                    data-field="dateOfBirth"
                    onchange={handleInputChange}
                    required>
                </lightning-input>
            </div>
            <div class="slds-col slds-size_1-of-1">
                <lightning-textarea
                    label="Address"
                    value={studentData.address}
                    data-field="address"
                    onchange={handleInputChange}>
                </lightning-textarea>
            </div>
        </div>
    </div>

    <!-- Step 2: Academic Information -->
    <div if:true={isStep2} class="slds-m-top_medium">
        <div class="slds-grid slds-wrap slds-gutters">
            <div class="slds-col slds-size_1-of-2">
                <lightning-combobox
                    label="Previous Education Level"
                    value={studentData.previousEducation}
                    options={educationOptions}
                    data-field="previousEducation"
                    onchange={handleInputChange}
                    required>
                </lightning-combobox>
            </div>
            <div class="slds-col slds-size_1-of-2">
                <lightning-input
                    label="Previous GPA"
                    type="number"
                    step="0.01"
                    min="0"
                    max="4"
```

```
                        value={applicationData.gpa}
                        data-field="gpa"
                        onchange={handleAppInputChange}>
                    </lightning-input>
                </div>
                <div class="slds-col slds-size_1-of-2">
                    <lightning-input
                        label="Emergency Contact Name"
                        value={studentData.emergencyContactName}
                        data-field="emergencyContactName"
                        onchange={handleInputChange}>
                    </lightning-input>
                </div>
                <div class="slds-col slds-size_1-of-2">
                    <lightning-input
                        label="Emergency Contact Phone"
                        type="tel"
                        value={studentData.emergencyContactPhone}
                        data-field="emergencyContactPhone"
                        onchange={handleInputChange}>
                    </lightning-input>
                </div>
            </div>
        </div>

        <!-- Step 3: Application Information -->
        <div if:true={isStep3} class="slds-m-top_medium">
            <div class="slds-grid slds-wrap slds-gutters">
                <div class="slds-col slds-size_1-of-2">
                    <lightning-combobox
                        label="Intended Program"
                        value={applicationData.program}
                        options={programOptions}
                        data-field="program"
                        onchange={handleAppInputChange}
                        required>
                    </lightning-combobox>
                </div>
                <div class="slds-col slds-size_1-of-2">
                    <lightning-combobox
                        label="Intended Major"
                        value={applicationData.major}
                        options={majorOptions}
                        data-field="major"
                        onchange={handleAppInputChange}
                        required>
                    </lightning-combobox>
                </div>
                <div class="slds-col slds-size_1-of-1">
                    <lightning-textarea
                        label="Personal Statement"
                        value={applicationData.personalStatement}
                        data-field="personalStatement"
                        onchange={handleAppInputChange}
                        max-length="2000"
                        required>
```

```
                    </lightning-textarea>
                </div>
                <div class="slds-col slds-size_1-of-2">
                    <lightning-input
                        label="Test Scores (e.g., SAT: 1400)"
                        value={applicationData.testScores}
                        data-field="testScores"
                        onchange={handleAppInputChange}>
                    </lightning-input>
                </div>
            </div>
        </div>

        <!-- Step 4: Confirmation -->
        <div if:true={isStep4} class="slds-m-top_medium">
            <div class="slds-box slds-theme_success">
                <h3 class="slds-text-heading_medium">Registration Successful!</h3>
                <p class="slds-m-top_small">
                    Your application has been submitted successfully.
                </p>
                <p class="slds-m-top_small">
                    <strong>Application Number:</strong> {applicationNumber}
                </p>
                <p class="slds-m-top_small">
                    You will receive a confirmation email shortly with next steps.
                </p>
            </div>
        </div>

        <!-- Error Message -->
        <div if:true={errorMessage} class="slds-m-top_medium">
            <lightning-formatted-text
                value={errorMessage}
                class="slds-text-color_error">
            </lightning-formatted-text>
        </div>

        <!-- Navigation Buttons -->
        <div class="slds-m-top_large slds-text-align_center">
            <lightning-button
                if:false={isStep1}
                label="Previous"
                variant="neutral"
                onclick={handlePrevious}
                class="slds-m-right_small">
            </lightning-button>

            <lightning-button
                if:false={isStep4}
                label={nextButtonLabel}
                variant="brand"
                onclick={handleNext}
                disabled={isLoading}>
            </lightning-button>

            <lightning-button
```

```
                    if:true={isStep4}
                    label="Go to Course Registration"
                    variant="brand"
                    onclick={navigateToCourseRegistration}>
                </lightning-button>
            </div>

            <!-- Loading Spinner -->
            <div if:true={isLoading} class="slds-spinner_container">
                <lightning-spinner alternative-text="Processing..." size="medium"></light
            </div>

        </div>
    </lightning-card>
</template>
```

studentRegistration.js

```
import { LightningElement, track } from 'lwc';
import { ShowToastEvent } from 'lightning/platformShowToastEvent';
import { NavigationMixin } from 'lightning/navigation';
import registerStudent from '@salesforce/apex/StudentRegistrationController.registerStude

export default class StudentRegistration extends NavigationMixin(LightningElement) {
    @track currentStep = 'step1';
    @track studentData = {
        firstName: '',
        lastName: '',
        email: '',
        phone: '',
        dateOfBirth: '',
        address: '',
        previousEducation: '',
        emergencyContactName: '',
        emergencyContactPhone: ''
    };

    @track applicationData = {
        program: '',
        major: '',
        gpa: '',
        personalStatement: '',
        testScores: ''
    };

    @track applicationNumber = '';
    @track errorMessage = '';
    @track isLoading = false;

    // Step management
    get isStep1() { return this.currentStep === 'step1'; }
    get isStep2() { return this.currentStep === 'step2'; }
    get isStep3() { return this.currentStep === 'step3'; }
    get isStep4() { return this.currentStep === 'step4'; }
```

```
    get nextButtonLabel() {
        if (this.currentStep === 'step3') return 'Submit Application';
        return 'Next';
    }

    // Picklist options
    educationOptions = [
        { label: 'High School', value: 'High School' },
        { label: 'Undergraduate', value: 'Undergraduate' },
        { label: 'Graduate', value: 'Graduate' }
    ];

    programOptions = [
        { label: 'Undergraduate', value: 'Undergraduate' },
        { label: 'Graduate', value: 'Graduate' },
        { label: 'Certificate', value: 'Certificate' }
    ];

    majorOptions = [
        { label: 'Computer Science', value: 'Computer Science' },
        { label: 'Mathematics', value: 'Mathematics' },
        { label: 'Physics', value: 'Physics' },
        { label: 'Chemistry', value: 'Chemistry' },
        { label: 'Biology', value: 'Biology' },
        { label: 'English', value: 'English' }
    ];

    // Event handlers
    handleInputChange(event) {
        const field = event.target.dataset.field;
        this.studentData[field] = event.target.value;
    }

    handleAppInputChange(event) {
        const field = event.target.dataset.field;
        this.applicationData[field] = event.target.value;
    }

    handlePrevious() {
        if (this.currentStep === 'step2') this.currentStep = 'step1';
        else if (this.currentStep === 'step3') this.currentStep = 'step2';
        this.errorMessage = '';
    }

    async handleNext() {
        if (this.validateCurrentStep()) {
            if (this.currentStep === 'step1') {
                this.currentStep = 'step2';
            } else if (this.currentStep === 'step2') {
                this.currentStep = 'step3';
            } else if (this.currentStep === 'step3') {
                await this.submitApplication();
            }
        }
    }
```

```
    validateCurrentStep() {
        let isValid = true;
        this.errorMessage = '';

        if (this.currentStep === 'step1') {
            if (!this.studentData.firstName || !this.studentData.lastName ||
                !this.studentData.email || !this.studentData.dateOfBirth) {
                this.errorMessage = 'Please fill in all required fields.';
                isValid = false;
            } else if (!this.studentData.email.endsWith('@university.edu')) {
                this.errorMessage = 'Please use a valid university email address.';
                isValid = false;
            }
        } else if (this.currentStep === 'step2') {
            if (!this.studentData.previousEducation) {
                this.errorMessage = 'Please select your previous education level.';
                isValid = false;
            }
        } else if (this.currentStep === 'step3') {
            if (!this.applicationData.program || !this.applicationData.major ||
                !this.applicationData.personalStatement) {
                this.errorMessage = 'Please fill in all required application fields.';
                isValid = false;
            }
        }

        return isValid;
    }

    async submitApplication() {
        this.isLoading = true;
        this.errorMessage = '';

        try {
            const result = await registerStudent({
                studentData: this.studentData,
                applicationData: this.applicationData
            });

            this.applicationNumber = result;
            this.currentStep = 'step4';

            this.showToast('Success', 'Application submitted successfully!', 'success');

        } catch (error) {
            this.errorMessage = error.body?.message || 'An error occurred while submittin
            console.error('Registration error:', error);

        } finally {
            this.isLoading = false;
        }
    }

    navigateToCourseRegistration() {
        this[NavigationMixin.Navigate]({
            type: 'standard__component',
```

```
            attributes: {
                componentName: 'c__courseEnrollment'
            }
        });
    }

    showToast(title, message, variant) {
        const event = new ShowToastEvent({
            title: title,
            message: message,
            variant: variant
        });
        this.dispatchEvent(event);
    }
}
```

2. Course Enrollment Component

courseEnrollment.html

```
<template>
    <lightning-card title="Course Enrollment" icon-name="standard:calibration">
        <div class="slds-p-horizontal_medium">

            <!-- Filters -->
            <div class="slds-grid slds-wrap slds-gutters slds-m-bottom_medium">
                <div class="slds-col slds-size_1-of-3">
                    <lightning-combobox
                        label="Program Level"
                        value={selectedProgram}
                        options={programOptions}
                        onchange={handleProgramChange}
                        placeholder="Select Program">
                    </lightning-combobox>
                </div>
                <div class="slds-col slds-size_1-of-3">
                    <lightning-combobox
                        label="Semester"
                        value={selectedSemester}
                        options={semesterOptions}
                        onchange={handleSemesterChange}
                        placeholder="Select Semester">
                    </lightning-combobox>
                </div>
                <div class="slds-col slds-size_1-of-3">
                    <lightning-combobox
                        label="Department"
                        value={selectedDepartment}
                        options={departmentOptions}
                        onchange={handleDepartmentChange}
                        placeholder="All Departments">
                    </lightning-combobox>
                </div>
            </div>
```

```
            <!-- Course Selection Table -->
            <div class="slds-m-bottom_medium">
                <h3 class="slds-text-heading_small slds-m-bottom_small">Available Courses

                <lightning-datatable
                    data={filteredCourses}
                    columns={courseColumns}
                    key-field="Id"
                    selected-rows={selectedCourseIds}
                    onrowselection={handleCourseSelection}
                    max-row-selection="6"
                    hide-checkbox-column={false}>
                </lightning-datatable>
            </div>

            <!-- Enrollment Summary -->
            <div if:true={hasSelectedCourses} class="slds-box slds-theme_shade slds-m-bot
                <h3 class="slds-text-heading_small slds-m-bottom_small">Enrollment Summar

                <div class="slds-grid slds-wrap">
                    <div class="slds-col slds-size_1-of-3">
                        <dl class="slds-list_horizontal">
                            <dt class="slds-item_label">Selected Courses:</dt>
                            <dd class="slds-item_detail">{selectedCourseIds.length}</dd>
                        </dl>
                    </div>
                    <div class="slds-col slds-size_1-of-3">
                        <dl class="slds-list_horizontal">
                            <dt class="slds-item_label">Total Credits:</dt>
                            <dd class="slds-item_detail">{totalCredits}</dd>
                        </dl>
                    </div>
                    <div class="slds-col slds-size_1-of-3">
                        <dl class="slds-list_horizontal">
                            <dt class="slds-item_label">Total Fee:</dt>
                            <dd class="slds-item_detail">{formattedTotalFee}</dd>
                        </dl>
                    </div>
                </div>

                <!-- Selected Courses List -->
                <div class="slds-m-top_small">
                    <h4 class="slds-text-heading_x-small">Selected Courses:</h4>
                    <ul class="slds-list_dotted slds-m-top_x-small">
                        <template for:each={selectedCourseDetails} for:item="course">
                            <li key={course.Id} class="slds-item">
                                {course.Course_Code__c} - {course.Course_Name__c}
                                ({course.Credits__c} credits)
                            </li>
                        </template>
                    </ul>
                </div>
            </div>

            <!-- Error Messages -->
```

```
            <div if:true={errorMessage} class="slds-m-bottom_medium">
                <lightning-formatted-text
                    value={errorMessage}
                    class="slds-text-color_error">
                </lightning-formatted-text>
            </div>

            <!-- Action Buttons -->
            <div class="slds-text-align_center">
                <lightning-button
                    label="Reset Selection"
                    variant="neutral"
                    onclick={handleReset}
                    disabled={isLoading}
                    class="slds-m-right_small">
                </lightning-button>

                <lightning-button
                    label="Enroll in Selected Courses"
                    variant="brand"
                    onclick={handleEnrollment}
                    disabled={enrollButtonDisabled}
                    class="slds-m-right_small">
                </lightning-button>
            </div>

            <!-- Loading Spinner -->
            <div if:true={isLoading} class="slds-spinner_container">
                <lightning-spinner alternative-text="Loading..." size="medium"></lightnir
            </div>

        </div>
    </lightning-card>

    <!-- Enrollment Success Modal -->
    <template if:true={showSuccessModal}>
        <section role="dialog" tabindex="-1" aria-modal="true" class="slds-modal slds-fad
            <div class="slds-modal__container">
                <header class="slds-modal__header">
                    <h2 class="slds-text-id_focus slds-modal__title">Enrollment Successfu
                </header>
                <div class="slds-modal__content slds-p-around_medium">
                    <p class="slds-m-bottom_small">
                        You have been successfully enrolled in {enrollmentResult.enrollme
                    </p>
                    <p class="slds-m-bottom_small">
                        <strong>Total Fee:</strong> {enrollmentResult.formattedTotalFee}
                    </p>
                    <p class="slds-text-body_small">
                        You will receive confirmation emails for each course enrollment.
                    </p>
                </div>
                <footer class="slds-modal__footer">
                    <lightning-button
                        label="Close"
                        variant="brand"
```

```
                        onclick={handleCloseModal}>
                    </lightning-button>
                </footer>
            </div>
        </section>
        <div class="slds-backdrop slds-backdrop_open"></div>
    </template>
</template>
```

courseEnrollment.js

```
import { LightningElement, track, wire } from 'lwc';
import { ShowToastEvent } from 'lightning/platformShowToastEvent';
import getAvailableCourses from '@salesforce/apex/StudentRegistrationController.getAvaila
import enrollInCourses from '@salesforce/apex/StudentRegistrationController.enrollInCours
import CURRENT_USER_ID from '@salesforce/user/Id';

const COURSE_COLUMNS = [
    { label: 'Course Code', fieldName: 'Course_Code__c', type: 'text' },
    { label: 'Course Name', fieldName: 'Course_Name__c', type: 'text' },
    { label: 'Department', fieldName: 'Department__c', type: 'text' },
    { label: 'Credits', fieldName: 'Credits__c', type: 'number' },
    { label: 'Fee', fieldName: 'Course_Fee__c', type: 'currency' },
    {
        label: 'Capacity',
        fieldName: 'capacityInfo',
        type: 'text',
        cellAttributes: { alignment: 'center' }
    },
    {
        label: 'Status',
        fieldName: 'Status__c',
        type: 'text',
        cellAttributes: {
            iconName: { fieldName: 'statusIcon' },
            iconPosition: 'left'
        }
    }
];

export default class CourseEnrollment extends LightningElement {
    @track selectedProgram = 'Undergraduate';
    @track selectedSemester = 'Fall';
    @track selectedDepartment = '';
    @track selectedCourseIds = [];
    @track courses = [];
    @track filteredCourses = [];
    @track isLoading = false;
    @track errorMessage = '';
    @track showSuccessModal = false;
    @track enrollmentResult = {};

    courseColumns = COURSE_COLUMNS;
    currentStudentId = ''; // This would be set based on current logged in student
```

```
    // Picklist options
    programOptions = [
        { label: 'Undergraduate', value: 'Undergraduate' },
        { label: 'Graduate', value: 'Graduate' }
    ];

    semesterOptions = [
        { label: 'Fall', value: 'Fall' },
        { label: 'Spring', value: 'Spring' },
        { label: 'Summer', value: 'Summer' }
    ];

    departmentOptions = [
        { label: 'All Departments', value: '' },
        { label: 'Computer Science', value: 'Computer Science' },
        { label: 'Mathematics', value: 'Mathematics' },
        { label: 'Physics', value: 'Physics' },
        { label: 'Chemistry', value: 'Chemistry' },
        { label: 'Biology', value: 'Biology' },
        { label: 'English', value: 'English' }
    ];

    connectedCallback() {
        // Initialize with default values
        this.loadCourses();
    }

    // Computed properties
    get hasSelectedCourses() {
        return this.selectedCourseIds.length > 0;
    }

    get selectedCourseDetails() {
        return this.courses.filter(course =>
            this.selectedCourseIds.includes(course.Id)
        );
    }

    get totalCredits() {
        return this.selectedCourseDetails.reduce((total, course) =>
            total + (course.Credits__c || 0), 0
        );
    }

    get totalFee() {
        return this.selectedCourseDetails.reduce((total, course) =>
            total + (course.Course_Fee__c || 0), 0
        );
    }

    get formattedTotalFee() {
        return new Intl.NumberFormat('en-US', {
            style: 'currency',
            currency: 'USD'
        }).format(this.totalFee);
    }
```

```
    get enrollButtonDisabled() {
        return this.isLoading || this.selectedCourseIds.length === 0;
    }

    // Event handlers
    handleProgramChange(event) {
        this.selectedProgram = event.detail.value;
        this.loadCourses();
    }

    handleSemesterChange(event) {
        this.selectedSemester = event.detail.value;
        this.loadCourses();
    }

    handleDepartmentChange(event) {
        this.selectedDepartment = event.detail.value;
        this.filterCourses();
    }

    handleCourseSelection(event) {
        this.selectedCourseIds = event.detail.selectedRows.map(row => row.Id);
    }

    handleReset() {
        this.selectedCourseIds = [];
        this.errorMessage = '';

        // Clear selection in datatable
        this.template.querySelector('lightning-datatable').selectedRows = [];
    }

    async handleEnrollment() {
        if (this.selectedCourseIds.length === 0) {
            this.errorMessage = 'Please select at least one course to enroll.';
            return;
        }

        if (this.selectedCourseIds.length > 6) {
            this.errorMessage = 'You can enroll in maximum 6 courses per semester.';
            return;
        }

        this.isLoading = true;
        this.errorMessage = '';

        try {
            const result = await enrollInCourses({
                studentId: this.currentStudentId,
                courseIds: this.selectedCourseIds
            });

            if (result.success) {
                this.enrollmentResult = {
                    enrollmentCount: result.enrollmentCount,
```

```
                formattedTotalFee: new Intl.NumberFormat('en-US', {
                    style: 'currency',
                    currency: 'USD'
                }).format(result.totalFee),
                enrolledCourses: result.enrolledCourses
            };

            this.showSuccessModal = true;
            this.handleReset();
            this.loadCourses(); // Refresh course data

        } else {
            this.errorMessage = result.error || 'Enrollment failed. Please try again.
        }

    } catch (error) {
        this.errorMessage = error.body?.message || 'An error occurred during enrollme
        console.error('Enrollment error:', error);

    } finally {
        this.isLoading = false;
    }
}

handleCloseModal() {
    this.showSuccessModal = false;
}

// Data loading methods
async loadCourses() {
    this.isLoading = true;
    this.errorMessage = '';

    try {
        const result = await getAvailableCourses({
            program: this.selectedProgram,
            semester: this.selectedSemester
        });

        this.courses = result.map(course => ({
            ...course,
            capacityInfo: `${course.Current_Enrollment__c}/${course.Max_Capacity__c}`
            statusIcon: this.getStatusIcon(course.Status__c)
        }));

        this.filterCourses();

    } catch (error) {
        this.errorMessage = error.body?.message || 'Error loading courses.';
        console.error('Course loading error:', error);

    } finally {
        this.isLoading = false;
    }
}
```

```
    filterCourses() {
        if (this.selectedDepartment) {
            this.filteredCourses = this.courses.filter(course =>
                course.Department__c === this.selectedDepartment
            );
        } else {
            this.filteredCourses = [...this.courses];
        }
    }

    getStatusIcon(status) {
        switch (status) {
            case 'Active':
                return 'utility:success';
            case 'Full':
                return 'utility:warning';
            case 'Inactive':
                return 'utility:error';
            default:
                return 'utility:info';
        }
    }

    showToast(title, message, variant) {
        const event = new ShowToastEvent({
            title: title,
            message: message,
            variant: variant
        });
        this.dispatchEvent(event);
    }
}
```

Lightning App Builder Configuration

1. Student Registration App Page

```
<!-- studentRegistrationApp.app-meta.xml -->
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>58.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__AppPage</target>
        <target>lightning__HomePage</target>
    </targets>
    <targetConfigs>
        <targetConfig targets="lightning__AppPage">
            <supportedFormFactors>
                <supportedFormFactor type="Large"/>
                <supportedFormFactor type="Small"/>
            </supportedFormFactors>
        </targetConfig>
```

```
    </targetConfigs>
</LightningComponentBundle>
```

2. Home Page Configuration

- **Header**: University logo and navigation menu
- **Main Area**: Welcome message and quick action tiles
- **Sidebar**: Recent applications and announcements
- **Footer**: Contact information and links

3. Record Page Layouts

Student Record Page

```
{
    "layout": "two-column",
    "regions": [
        {
            "name": "main",
            "components": [
                {
                    "componentName": "force:recordDetailPanel",
                    "properties": {
                        "recordId": "{recordId}",
                        "objectApiName": "Student__c"
                    }
                },
                {
                    "componentName": "forceContent:recordDetailPanel",
                    "properties": {
                        "body": "Related Applications"
                    }
                }
            ]
        },
        {
            "name": "sidebar",
            "components": [
                {
                    "componentName": "runtime_sales_activities:activityPanel"
                },
                {
                    "componentName": "forceChatter:recordFeedContainer"
                }
            ]
        }
    ]
}
```

Navigation Configuration

1. Custom Tab Creation

- **Student Management Tab**: Access to student records and applications
- **Course Catalog Tab**: Browse available courses
- **Enrollment Tab**: Course enrollment functionality
- **Reports Tab**: Academic reports and analytics

2. Navigation Menu Items

```
// Custom navigation menu configuration
const navigationItems = [
    {
        "label": "Home",
        "name": "Home",
        "type": "standard__namedPage",
        "attributes": {
            "pageName": "home"
        }
    },
    {
        "label": "Student Registration",
        "name": "StudentRegistration",
        "type": "standard__component",
        "attributes": {
            "componentName": "c__studentRegistration"
        }
    },
    {
        "label": "Course Enrollment",
        "name": "CourseEnrollment",
        "type": "standard__component",
        "attributes": {
            "componentName": "c__courseEnrollment"
        }
    },
    {
        "label": "My Applications",
        "name": "MyApplications",
        "type": "standard__objectPage",
        "attributes": {
            "objectApiName": "Application__c",
            "actionName": "list"
        }
    }
];
```

Utility Bar Configuration

1. Quick Actions

```
<!-- quickActions.xml -->
<quickActions>
    <quickAction>
        <label>New Student Registration</label>
        <name>New_Student_Registration</name>
        <optionsCreateFeedItem>true</optionsCreateFeedItem>
        <quickActionLayout>
            <layoutSectionStyle>TwoColumnsLeftToRight</layoutSectionStyle>
            <quickActionLayoutColumns>
                <quickActionLayoutItems>
                    <field>First_Name__c</field>
                </quickActionLayoutItems>
                <quickActionLayoutItems>
                    <field>Last_Name__c</field>
                </quickActionLayoutItems>
                <quickActionLayoutItems>
                    <field>Email__c</field>
                </quickActionLayoutItems>
            </quickActionLayoutColumns>
        </quickActionLayout>
        <successMessage>Student registration initiated successfully!</successMessage>
        <targetObject>Student__c</targetObject>
        <type>Create</type>
    </quickAction>
</quickActions>
```

2. Utility Bar Items

- **Help & Support**: Access to documentation and support resources
- **Notifications**: Real-time application status updates
- **Calendar**: Academic calendar and important dates
- **Calculator**: GPA and credit hour calculator

Custom CSS Styling

1. studentPortal.css

```
/* Custom CSS for Student Portal */
.student-portal-container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 1rem;
}

.registration-progress {
    margin-bottom: 2rem;
```

```
}

.step-content {
    min-height: 400px;
    padding: 1rem 0;
}

.course-selection-table {
    background: white;
    border-radius: 0.25rem;
    box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}

.enrollment-summary {
    background: #f3f3f3;
    padding: 1rem;
    border-radius: 0.25rem;
    margin: 1rem 0;
}

.summary-stats {
    display: flex;
    justify-content: space-around;
    margin-bottom: 1rem;
}

.stat-item {
    text-align: center;
}

.stat-value {
    font-size: 2rem;
    font-weight: bold;
    color: #0176d3;
}

.stat-label {
    font-size: 0.875rem;
    color: #706e6b;
}

.selected-courses-list {
    background: white;
    padding: 1rem;
    border-radius: 0.25rem;
    border-left: 4px solid #0176d3;
}

.success-modal-content {
    text-align: center;
    padding: 2rem;
}

.success-icon {
    color: #04844b;
    font-size: 3rem;
```

```
    margin-bottom: 1rem;
}

/* Responsive design */
@media (max-width: 768px) {
    .student-portal-container {
        padding: 0.5rem;
    }

    .summary-stats {
        flex-direction: column;
        gap: 1rem;
    }

    .slds-grid .slds-col {
        width: 100% !important;
    }
}

/* Custom button styles */
.enrollment-button {
    background: linear-gradient(45deg, #0176d3, #0d5db8);
    border: none;
    color: white;
    padding: 0.75rem 1.5rem;
    border-radius: 0.25rem;
    font-weight: bold;
    cursor: pointer;
    transition: all 0.3s ease;
}

.enrollment-button:hover {
    background: linear-gradient(45deg, #0d5db8, #094a94);
    transform: translateY(-2px);
    box-shadow: 0 4px 8px rgba(0,0,0,0.2);
}

.enrollment-button:disabled {
    background: #dddbda;
    color: #706e6b;
    cursor: not-allowed;
    transform: none;
    box-shadow: none;
}

/* Loading spinner overlay */
.loading-overlay {
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background: rgba(255,255,255,0.8);
    display: flex;
    align-items: center;
    justify-content: center;
```

```
    z-index: 9999;
}

/* Error message styling */
.error-message {
    background: #fef7f7;
    border: 1px solid #c23934;
    color: #c23934;
    padding: 1rem;
    border-radius: 0.25rem;
    margin: 1rem 0;
}

.error-message::before {
    content: "⚠ ";
    margin-right: 0.5rem;
}

/* Success message styling */
.success-message {
    background: #f3f9f1;
    border: 1px solid #04844b;
    color: #04844b;
    padding: 1rem;
    border-radius: 0.25rem;
    margin: 1rem 0;
}

.success-message::before {
    content: "✓ ";
    margin-right: 0.5rem;
}
```

Lightning Web Component Metadata

1. Component Bundle Structure

```
studentRegistration/
├── studentRegistration.html
├── studentRegistration.js
├── studentRegistration.js-meta.xml
├── studentRegistration.css
└── __tests__/
    └── studentRegistration.test.js

courseEnrollment/
├── courseEnrollment.html
├── courseEnrollment.js
├── courseEnrollment.js-meta.xml
├── courseEnrollment.css
└── __tests__/
    └── courseEnrollment.test.js
```

2. Meta.xml Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>58.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__AppPage</target>
        <target>lightning__RecordPage</target>
        <target>lightning__HomePage</target>
        <target>lightningCommunity__Page</target>
    </targets>
    <targetConfigs>
        <targetConfig targets="lightning__RecordPage">
            <objects>
                <object>Student__c</object>
                <object>Application__c</object>
            </objects>
        </targetConfig>
        <targetConfig targets="lightningCommunity__Page">
            <property name="title" type="String" label="Component Title" default="Student
        </targetConfig>
    </targetConfigs>
</LightningComponentBundle>
```