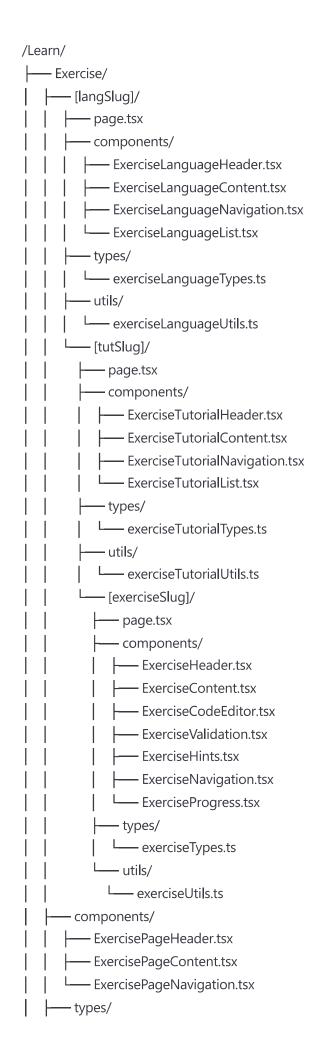
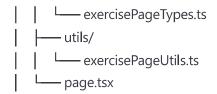
Exercise Pages Structure & Implementation Guide

Directory Structure





Page Parameters Structure

Exercise Main Page (/Learn/Exercise/page.tsx)

• **Route**: /Learn/Exercise

• Parameters: None

• **Purpose**: Lists all programming languages available for exercises

Exercise Language Page (/Learn/Exercise/[langSlug]/page.tsx)

• **Route**: (/Learn/Exercise/[langSlug])

Parameters:

• (langSlug): string (e.g., "javascript", "python", "java")

• Purpose: Lists all tutorials available for the selected language

Exercise Tutorial Page (/Learn/Exercise/[langSlug]/[tutSlug]/page.tsx)

Route: (/Learn/Exercise/[langSlug]/[tutSlug])

Parameters:

- (langSlug): string (programming language)
- (tutSlug): string (tutorial identifier)
- Purpose: Lists all exercises available for the selected tutorial

Exercise Detail Page (/Learn/Exercise/[langSlug]/[tutSlug]/[exerciseSlug]/page.tsx)

Route: (/Learn/Exercise/[langSlug]/[tutSlug]/[exerciseSlug])

• Parameters:

- (langSlug): string (programming language)
- (tutSlug): string (tutorial identifier)
- (exerciseSlug): string (exercise identifier)
- Purpose: Individual exercise with code editor, validation, hints, and progress tracking

Data Fetching Functions (SSR)

Base Fetch Function Location

File: (/Learn/Exercise/utils/exercisePageUtils.ts) Uses: (fetchCollection) from (@/lib/FetchDataPayload.ts) getData pattern: Create specific functions in (@/lib/getData.ts)

Function Signatures and Locations

1. getExerciseLanguages

```
typescript

// Location: @/lib/getData.ts

// Function: getExerciseLanguages(): Promise<ExerciseLanguage[]>

// Collection: "exercise-languages"

// Parameters: none

// Returns: Array of language objects with slug, name, description, icon

// Usage: Exercise main page listing
```

2. getExerciseLanguageBySlug

```
typescript
```

```
// Location: @/lib/getData.ts
// Function: getExerciseLanguageBySlug(langSlug: string): Promise<ExerciseLanguage | null>
// Collection: "exercise-languages"
// Where clause: { slug: { equals: langSlug } }
// Returns: Single language object or null
// Usage: Exercise language page data
```

${\bf 3.\ getExercise Tutorials By Language Id}$

```
typescript
```

```
// Location: @/lib/getData.ts

// Function: getExerciseTutorialsByLanguageId(languageId: string): Promise<ExerciseTutorial[]>

// Collection: "exercise-tutorials"

// Where clause: { language: { equals: languageId } }

// Sort: "index"

// Returns: Array of tutorial objects for the language

// Usage: Exercise language page tutorial listing
```

4. getExerciseTutorialBySlug

```
// Location: @/lib/getData.ts

// Function: getExerciseTutorialBySlug(tutSlug: string, languageId: string): Promise<ExerciseTutorial | null>

// Collection: "exercise-tutorials"

// Where clause: { and: [{ slug: { equals: tutSlug } }, { language: { equals: languageId } }] }

// Returns: Single tutorial object or null

// Usage: Exercise tutorial page data
```

5. getExercisesByTutorialId

```
typescript

// Location: @/lib/getData.ts

// Function: getExercisesByTutorialId(tutorialId: string): Promise<Exercise[]>

// Collection: "exercises"

// Where clause: { tutorial: { equals: tutorialId } }

// Sort: "index"

// Returns: Array of exercise objects for the tutorial
```

6. getExerciseBySlug

// Usage: Exercise tutorial page exercise listing

```
typescript

// Location: @/lib/getData.ts

// Function: getExerciseBySlug(exerciseSlug: string, tutorialId: string): Promise<Exercise | null>

// Collection: "exercises"

// Where clause: { and: [{ slug: { equals: exerciseSlug } }, { tutorial: { equals: tutorialId } }] }

// Depth: 2 (for full relationships)

// Returns: Single exercise object with full details
```

Type Definitions

// Usage: Exercise detail page data

Base Types Location

All exercise types should be defined in: (/Learn/Exercise/types/ExerciseTypes.ts) Import pattern: (import { ExerciseLanguage, ExerciseTutorial, Exercise } from "@/app/Learn/Exercise/types/ExerciseTypes")

Key Type Interfaces

ExerciseLanguage

```
interface ExerciseLanguage {
  id: string;
  slug: string;
  title: string;
  description: string;
  logo?: {
    url: string;
    alt: string;
  };
  tutorialCount: number;
  exerciseCount: number;
  index: number;
  createdAt: string;
  updatedAt: string;
}
```

ExerciseTutorial

```
interface ExerciseTutorial {
  id: string;
  slug: string;
  title: string;
  description: string;
  content?: string;
  language: ExerciseLanguage | string;
  difficulty: 'beginner' | 'intermediate' | 'advanced';
  exerciseCount: number;
  estimatedTime: string;
  index: number;
  createdAt: string;
  updatedAt: string;
}
```

Exercise

typescript

```
interface Exercise {
 id: string;
 slug: string;
 title: string;
 description: string;
 content?: string;
 language: ExerciseLanguage | string;
 tutorial: ExerciseTutorial | string;
 difficulty: 'easy' | 'medium' | 'hard';
 starterCode: string;
 solution: string;
 testCases: TestCase[];
 hints: string[];
 concepts: string[];
 timeEstimate: string;
 index: number;
 createdAt: string;
 updatedAt: string;
}
```

TestCase

```
typescript
interface TestCase {
  id: string;
  input: string;
  expectedOutput: string;
  description: string;
  hidden: boolean;
}
```

Component Props Types

typescript

```
interface ExercisePageContentProps {
 exercises: Exercise[];
 language: ExerciseLanguage;
 tutorial: ExerciseTutorial;
 currentExercise: Exercise:
 previousExercise?: Exercise;
 nextExercise?: Exercise;
}
interface ExerciseLanguagePageProps {
 language: ExerciseLanguage;
 tutorials: ExerciseTutorial[];
interface ExerciseTutorialPageProps {
 language: ExerciseLanguage;
 tutorial: ExerciseTutorial;
 exercises: Exercise[];
}
```

Component Architecture

Exercise Main Page Components (/Learn/Exercise/)

- (ExercisePageHeader.tsx): Main header with title "Programming Exercises", description, search/filter
- (ExercisePageContent.tsx): Language grid layout (similar to (LanguageGrid.tsx))
- (ExerciseLanguageCard.tsx): Individual language cards (similar to (LanguageCard.tsx))

Exercise Language Page Components (/Learn/Exercise/[langSlug]/)

- (ExerciseLanguageHeader.tsx): Language header with logo, stats, description (similar to LanguageHeader.tsx)
- (ExerciseLanguageContent.tsx): Main content wrapper
- (ExerciseTutorialsList.tsx): Tutorial grid/list (similar to (TutorialsList.tsx))
- ExerciseTutorialCard.tsx): Individual tutorial cards
- ExerciseLanguageBreadcrumb.tsx): Navigation breadcrumb

Exercise Tutorial Page Components (/Learn/Exercise/[langSlug]/[tutSlug]/)

- (ExerciseTutorialHeader.tsx): Tutorial header with info, progress, navigation
- (ExerciseTutorialContent.tsx): Main content wrapper

- (ExercisesList.tsx): Exercise grid/list layout
- (ExerciseCard.tsx): Individual exercise cards with difficulty, concepts
- (ExerciseTutorialBreadcrumb.tsx): Navigation breadcrumb
- (ExerciseTutorialSidebar.tsx): Sidebar for exercise navigation (similar to (TutorialSidebar.tsx))

Exercise Detail Page Components (/Learn/Exercise/[langSlug]/[tutSlug]/[exerciseSlug]/)

- (ExerciseDetailHeader.tsx): Exercise title, difficulty badge, concepts tags
- (ExerciseDetailContent.tsx): Main content wrapper
- (ExerciseDescription.tsx): Problem description and requirements
- (ExerciseCodeEditor.tsx): Code editor with syntax highlighting
- (ExerciseTestRunner.tsx): Test execution and validation feedback
- (ExerciseHintsPanel.tsx): Progressive hint system with toggle
- (ExerciseDetailNavigation.tsx): Previous/next exercise navigation
- (ExerciseProgressTracker.tsx): Progress tracking and completion status
- (ExerciseDetailSidebar.tsx): Exercise list sidebar for quick navigation
- ExerciseDetailBreadcrumb.tsx): Navigation breadcrumb

Shared Components (Global)

- (ExerciseCard.tsx): Reusable exercise card component
- (DifficultyBadge.tsx): Difficulty level indicator
- (ConceptTag.tsx): Concept/skill tags
- (CodeBlock.tsx): Syntax highlighted code display
- (ProgressBar.tsx): Progress indicator component

Design Theme Integration

Color Scheme

Follow the existing Learn section color palette:

- **Primary**: Blue gradient (from-blue-600 to-blue-800)
- **Background**: (bg-slate-900 dark:bg-slate-950) for main sections
- Cards: (bg-white dark:bg-slate-900) with borders
- **Text**: (text-white), (text-sky-100), (text-slate-700 dark:text-slate-300
- Accent: Sky blue ((sky-50), (sky-100), (sky-200)) for highlights

• Interactive: Blue hover states (hover:bg-blue-700)

Typography Classes

- Main Headers: (text-4xl font-bold text-white)
- **Section Headers**: (text-3xl font-extrabold text-slate-900 dark:text-white)
- **Subheaders**: (text-xl font-semibold text-slate-800 dark:text-gray-200)
- Body Text: (text-lg leading-relaxed text-slate-700 dark:text-slate-300)
- Code: (font-mono text-sm) with appropriate background colors
- Strong Text: (font-semibold text-blue-800 dark:text-blue-300)

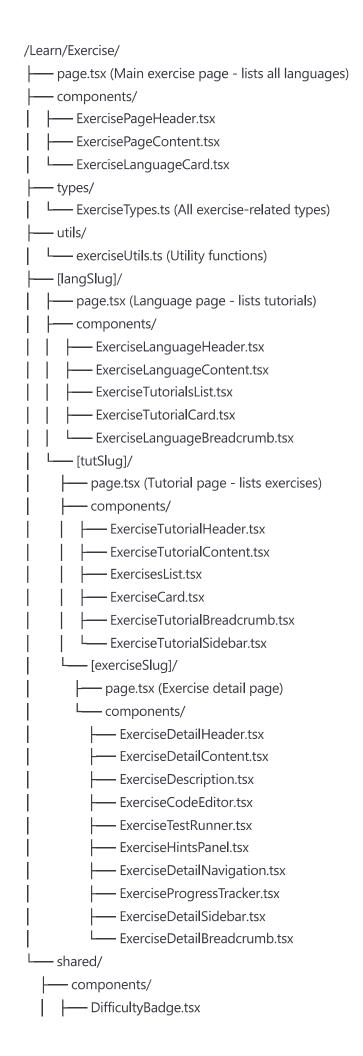
Layout Patterns

- Container: (mx-auto max-w-6xl px-6 py-12) (main content)
- **Grid**: (grid grid-cols-1 gap-6 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 xl:grid-cols-5)
- Cards: (rounded-lg bg-white dark:bg-slate-900 shadow-lg) with hover effects
- Sidebar: (w-80 bg-white dark:bg-slate-900) with toggle functionality
- **Breadcrumb**: Match existing (TutorialBreadcrumb) pattern

Interactive Elements

- **Buttons**: (bg-blue-600 hover:bg-blue-700 text-white rounded-lg px-4 py-2)
- Code Editor: (bg-slate-900 to-slate-800 rounded-xl border border-slate-700)
- Progress Bars: (bg-gradient-to-r from-blue-600 to-blue-800 h-2 rounded-full)
- Hover Effects: (hover:scale-[1.02] transform transition-all duration-300)
- Modals: Follow existing modal patterns with backdrop blur

File Structure Pattern





Global Utilities Location

- Base utilities: (/Learn/utils/) (existing)
- Exercise-specific utilities: (/Learn/Exercise/utils/exerciseUtils.ts)
- Shared helpers: (/Learn/Exercise/shared/utils/exerciseHelpers.ts

getData Functions Location

All data fetching functions in: @/lib/getData.ts (following existing pattern) Uses: (fetchCollection) from @/lib/FetchDataPayload.ts

Page Implementation Patterns

Exercise Main Page (/Learn/Exercise/page.tsx)

typescript

```
// Pattern: Similar to main Learn page
// Layout: Language grid (similar to LanguageGrid component)
// Data: getExerciseLanguages()
// Components: ExercisePageHeader, ExercisePageContent, ExerciseLanguageCard
// Styling: bg-slate-900 dark:bg-slate-950, grid layout
```

Exercise Language Page (/Learn/Exercise/[langSlug]/page.tsx)

typescript

```
// Pattern: Similar to Learn/[langSlug]/page.tsx
// Layout: Header + tutorial grid
// Data: getExerciseLanguageBySlug(), getExerciseTutorialsByLanguageId()
// Components: ExerciseLanguageHeader, ExerciseTutorialsList
// Features: Breadcrumb navigation, tutorial cards
// Error handling: notFound() if language not found
```

Exercise Tutorial Page (/Learn/Exercise/[langSlug]/[tutSlug]/page.tsx)

```
// Pattern: Similar to tutorial listing page

// Layout: Header + sidebar + exercise grid

// Data: getExerciseTutorialBySlug(), getExercisesByTutorialId()

// Components: ExerciseTutorialHeader, ExerciseTutorialSidebar, ExercisesList

// Features: Progress tracking, exercise cards with difficulty

// Sidebar: Exercise navigation (similar to TutorialSidebar)
```

Exercise Detail Page (/Learn/Exercise/[langSlug]/[tutSlug]/[exerciseSlug]/page.tsx)

typescript

```
// Pattern: Similar to TutorialPageContent with interactive elements
// Layout: Sidebar + main content (split view)
// Data: getExerciseBySlug() with full details
// Components: ExerciseDetailSidebar, ExerciseDetailContent, ExerciseCodeEditor
// Features: Code editor, test runner, hints system, progress tracking
// Interactive: Code execution, validation, hint progression
```

Navigation Flow

- 1. (/Learn/Exercise) \rightarrow Shows all programming languages
- 2. (/Learn/Exercise/[langSlug] $) \rightarrow$ Shows tutorials for selected language
- 3. $(/Learn/Exercise/[langSlug]/[tutSlug]) \rightarrow Shows exercises for selected tutorial$
- 4. $(\label{eq:learn/Exercise/[langSlug]/[tutSlug]/[exerciseSlug]}) \rightarrow Individual exercise interface$

Integration Points

- Uses existing (fetchCollection) function from the provided utility
- Follows existing Learn section component patterns
- Integrates with existing navigation and breadcrumb systems
- Maintains consistency with existing type definitions and styling
- Uses the same revalidation strategy (60 seconds) as existing pages