دانشکده علوم ریاضی

پروژه کارشناسی رشته آمار

طراحی و پیاده‌سازی سامانه آموزش تعاملی پاکسازی داده با زبان R مبتنی بر R Markdown و WebR با قابلیت اجرای آفلاین و ارزیابی خودکار

استاد راهنما : دکتر کلکین نما

نگارش و پیاده سازی : کوثر نگراوی

زمستان ۱۴۰۴

فهرست مطالب

چکیده

این پژوهش با هدف طراحی و پیاده‌سازی یک سامانه آموزشی تعاملی برای آموزش «پاکسازی داده‌ها در «R انجام شده است. در بسیاری از دوره‌های آموزشی، محتوای نظری از تمرین عملی جدا بوده و اجرای کد مستلزم نصب و پیکربندی نرم‌افزار در رایانه کاربر است؛ موضوعی که در محیط‌های آموزشی و سازمانی می‌تواند موجب کاهش مشارکت و افزایش خطاهای عملیاتی شود. در سامانه پیشنهادی، محتوای آموزشی به‌صورت ساختاریافته با استفاده از RMarkdown و Bookdown تولید شده و بخش‌های تمرینی به کمک WebR اجرای R در مرورگر مبتنی بر (WebAssembly) به‌صورت تعاملی در صفحات درس ادغام شده‌اند. بدین ترتیب، کاربر می‌تواند بدون نیاز به سرور محاسباتی و مستقیماً در محیط وب، کدهای R مرتبط با پاکسازی داده‌ها را اجرا و خروجی را مشاهده کند. برای تمرین‌ها، سازوکاری مبتنی بر اجرای کنترل‌شده کد و مدیریت خطاهای رایج پیاده‌سازی شده است تا فرآیند یادگیری به‌صورت گام‌به‌گام و قابل ارزیابی انجام شود. خروجی پروژه شامل یک چارچوب قابل توسعه برای تولید فصل‌های آموزشی پاکسازی داده، همراه با نمونه‌سازی عملی از فصل‌ها و تمرین‌های تعاملی است که قابلیت استقرار در قالب وب‌سایت و نیز اجرای آفلاین را دارد.

نسخه کامل سامانه شامل ۱۱۰ ماژول آموزشی است که در این گزارش، به‌دلیل محدودیت حجم، صرفاً معماری کلی و نمونه‌هایی منتخب ارائه شده‌اند.

فصل اول

کلیات پژوهش

۱-۱ مقدمه

در سال‌های اخیر، گسترش تصمیم‌گیری‌های مبتنی بر داده در سازمان‌ها و نهادهای پژوهشی، اهمیت مهارت‌های تحلیل و مدیریت داده را به‌طور چشمگیری افزایش داده است. با این حال، داده‌های اولیه در بسیاری از موارد دارای خطا، ناسازگاری یا نواقص ساختاری هستند و پیش از هرگونه تحلیل آماری، مرحله‌ای بنیادین تحت عنوان «پاکسازی داده (Data Cleaning) «باید انجام شود. این مرحله شامل شناسایی و اصلاح رکوردهای تکراری، مدیریت مقادیر گمشده، رفع ناسازگاری‌های ساختاری، تشخیص داده‌های پرت و اعمال قواعد اعتبارسنجی است. کیفیت اجرای این فرآیند مستقیماً بر صحت نتایج تحلیلی و تصمیم‌گیری‌های مبتنی بر داده اثرگذار است.

زبان برنامه‌نویسی R یکی از ابزارهای اصلی تحلیل داده در محیط‌های دانشگاهی و سازمانی محسوب می‌شود و بسته‌های متعددی برای پاکسازی و کنترل کیفیت داده ارائه می‌دهد. با این حال، در ایران منابع آموزشی Rعمدتاً به‌صورت دوره‌های مستقل یا مطالب پراکنده ارائه می‌شوند و تمرکز آن‌ها بیشتر بر آموزش دستورات پایه یا تحلیل آماری است. آموزش ساختاریافته و منسجمِ مفاهیم پاکسازی داده و مدیریت کیفیت داده کمتر به‌صورت مرحله‌به‌مرحله و نظام‌مند عرضه شده است. این وضعیت می‌تواند مانع شکل‌گیری درک یکپارچه از فرآیند پاکسازی داده و ابزارهای تخصصی مرتبط با آن شود.

علاوه بر این، بخش قابل‌توجهی از آموزش‌های موجود فاقد محیط تعاملی یکپارچه هستند و اجرای تمرین‌ها مستلزم نصب نرم‌افزار، پیکربندی محیط توسعه و مدیریت وابستگی‌های فنی است. این وابستگی‌ها می‌تواند در محیط‌های آموزشی گسترده یا سازمانی، مانعی برای یادگیری عملی ایجاد کند.

بر این اساس، در این پروژه چارچوبی طراحی و پیاده‌سازی شده است که آموزش مفاهیم و تکنیک‌های پاکسازی داده در R را به‌صورت ساختاریافته و همراه با امکان اجرای مستقیم کد در مرورگر ارائه می‌کند. این سامانه با بهره‌گیری از Bookdown برای سازمان‌دهی محتوا و WebR برای اجرای R مبتنی بر WebAssembly، بستری یکپارچه برای تلفیق آموزش نظری و تمرین عملی فراهم می‌سازد؛ به‌گونه‌ای که کاربر بدون نیاز به سرور محاسباتی یا نصب محلی نرم‌افزار، بتواند تمرین‌های مرتبط با پاکسازی داده را اجرا و نتیجه را مشاهده کند.

۲-۱ بیان مسئله

با وجود گسترش کاربرد زبان R در تحلیل داده، آموزش عملی پاکسازی داده‌ها همچنان با چالش‌هایی مواجه است. در بسیاری از منابع آموزشی، محتوای نظری و تمرین عملی از یکدیگر جدا بوده و اجرای کد نیازمند نصب نرم‌افزار و پیکربندی محیط اجرایی است. این موضوع می‌تواند مانع یادگیری پیوسته و یکپارچه شود، به‌ویژه در محیط‌هایی که دسترسی به زیرساخت فنی استاندارد وجود ندارد.

از سوی دیگر، نبود یک چارچوب آموزشی تعاملی که امکان اجرای مستقیم کد را در همان محیط مطالعه فراهم کند، موجب گسست میان یادگیری مفهومی و تمرین عملی می‌شود. همچنین در اغلب محیط‌های آموزشی موجود، مدیریت خطاهای کاربر به‌صورت کنترل‌شده انجام نمی‌شود و سیستم در برابر ورودی‌های نادرست پایدار نیست.

بر این اساس، مسئله اصلی این پژوهش را می‌توان چنین صورت‌بندی کرد:

چگونه می‌توان آموزش پاکسازی داده‌ها در R را به‌گونه‌ای طراحی کرد که:

- محتوای نظری و تمرین عملی در یک محیط یکپارچه ارائه شوند؛
- اجرای کد بدون نیاز به نصب محلی یا سرور محاسباتی انجام گیرد؛
- کاربر بتواند در همان صفحه آموزشی کد خود را اجرا و خروجی را مشاهده کند؛
- سامانه در برابر خطاهای کاربر پایدار و قابل‌کنترل باشد؛
- ساختار آموزشی قابلیت توسعه به تعداد زیادی فصل را داشته باشد.

فقدان یک چارچوب سبک، مقیاس‌پذیر و مستقل از زیرساخت محاسباتی مرکزی برای آموزش عملی پاکسازی داده‌ها، خلأیی است که این پروژه درصدد پاسخ به آن است.

۱-۳ اهداف پژوهش

هدف کلی

طراحی و پیاده‌سازی یک سامانه آموزشی تعاملی برای آموزش پاکسازی داده‌ها در R که امکان اجرای مستقیم کد در مرورگر و ارزیابی تمرین‌ها را بدون نیاز به زیرساخت سرور محاسباتی فراهم سازد.

اهداف اختصاصی

- طراحی یک ساختار استاندارد و قابل تعمیم برای توسعه فصل‌های آموزشی پاکسازی داده؛
- به‌کارگیری RMarkdown و Bookdown به‌منظور تولید محتوای آموزشی ساختاریافته و یکپارچه؛
- پیاده‌سازی محیط اجرای R در مرورگر با استفاده ازWebR ؛
- توسعه سازوکار اجرای کنترل‌شده کد کاربر و مدیریت خطاهای احتمالی؛
- فراهم‌سازی امکان استقرار وب‌محور سامانه و قابلیت استفاده در حالت آفلاین.

۱-۴ ضرورت و اهمیت پژوهش

پاکسازی داده‌ها در حوزه‌های آماری، مالی، پزشکی، زیست‌محیطی و سازمانی نقش تعیین‌کننده‌ای در تضمین صحت تحلیل‌ها دارد. خطا یا نقص در این مرحله می‌تواند منجر به سوگیری مدل‌های آماری، تولید نتایج نادرست و در نهایت تصمیم‌گیری‌های غیرقابل اتکا شود. از این‌رو، آموزش دقیق، ساختاریافته و عملی این مهارت برای دانشجویان و تحلیل‌گران داده ضروری است.

با این حال، شیوه‌های آموزشی سنتی که در آن‌ها تمرین عملی از بستر نظری جداست یا اجرای کد وابسته به نصب نرم‌افزار و پیکربندی محیط است، می‌تواند مانعی برای یادگیری مستمر و گسترده ایجاد کند. طراحی یک سامانه آموزشی تعاملی که امکان اجرای مستقیم کد را در همان محیط مطالعه فراهم کند، می‌تواند این گسست را برطرف سازد.

از منظر فناوری نیز، رویکرد اجرای سمت کاربر موجب کاهش وابستگی به سرور محاسباتی مرکزی، کاهش هزینه‌های نگهداری، افزایش مقیاس‌پذیری و بهبود دسترس‌پذیری سامانه در محیط‌های آموزشی مختلف می‌شود. این ویژگی‌ها اهمیت طراحی چنین بستری را دوچندان می‌کند.

۱-۵ نوآوری پژوهش

نوآوری این پروژه در ترکیب هدفمند سه مؤلفه فنی و آموزشی نهفته است:

- به‌کارگیری فناوری WebAssembly از طریق WebR برای اجرای مستقیم کد R در مرورگر؛
- طراحی یک چارچوب ساختاریافته برای آموزش تعاملی پاکسازی داده، بدون وابستگی به سرور محاسباتی؛
- پیاده‌سازی سازوکار اجرای کنترل‌شده کد و مدیریت خطا در سمت کاربر به‌منظور ایجاد بازخورد فوری و پایدار.

نوآوری این پروژه نه در توسعه یک فناوری جدید، بلکه در تلفیق این مؤلفه‌ها در قالب یک بستر آموزشی مقیاس‌پذیر برای پاکسازی داده در R است. این ترکیب امکان تولید تعداد زیادی فصل آموزشی با رفتار اجرایی یکنواخت و قابلیت توسعه‌پذیری را فراهم می‌کند.

۱-۶ ساختار استاندارد هر ماژول آموزشی

در این پروژه، هر ماژول آموزشی بر اساس یک الگوی طراحی ثابت و از پیش‌تعریف‌شده توسعه یافته است. هدف از این طراحی، ایجاد انسجام مفهومی و تسهیل شکل‌گیری ساختار ذهنی پایدار در یادگیرنده برای تحلیل ابزارهای مختلف پاکسازی داده است.

ساختار هر ماژول شامل اجزای زیر است:

۱. معرفی مفهومی تابع
۲. تبیین مبانی نظری یا الگوریتمی مرتبط
۳. تحلیل ساختار، آرگومان‌ها و پارامترهای ورودی
۴. بررسی رفتار تابع در سناریوهای متفاوت
۵. ارائه مثال‌های مرحله‌به‌مرحله همراه با تحلیل خروجی
۶. کاربرد در مسائل واقعی داده‌محور
۷. بیان محدودیت‌ها، خطاهای رایج و نکات بهینه‌سازی
۸. طراحی تمرین‌های تعاملی هدفمند برای تعمیق یادگیری

ثبات این ساختار موجب کاهش بار شناختی (Cognitive Load) در فرآیند یادگیری می‌شود؛ زیرا فراگیر در هر فصل با قالبی یکنواخت مواجه است و تمرکز خود را بر درک مفاهیم و تحلیل رفتار ابزار موردنظر

معطوف می‌کند، نه بر تطبیق با ساختار جدید. افزون بر این، چنین طراحی‌ای امکان توسعه مقیاس‌پذیر تعداد زیادی فصل آموزشی را بدون تغییر در چارچوب کلی فراهم می‌سازد.

فصل دوم

مبانی نظری و فناوری‌های مورد استفاده

۲-۱ پاکسازی داده و مدیریت کیفیت داده

۲-۱-۱ جایگاه پاکسازی داده در چرخه تحلیل

در چرخه عمر داده(Data Lifecycle) ، پس از مرحله گردآوری و پیش از تحلیل اکتشافی یا مدل‌سازی، مرحله‌ای بنیادین تحت عنوان «پاکسازی داده» قرار دارد. این مرحله با هدف شناسایی،صلاح یا حذف خطاها و ناسازگاری‌های موجود در داده‌های خام انجام می‌شود. گزارش‌های تجربی در حوزه علم داده نشان می‌دهد که بخش قابل توجهی از زمان پروژه‌های تحلیلی در برخی برآوردها تا حدود ۷۰ تا ۸۰ درصد صرف آماده‌سازی و پاکسازی داده می‌شود. این واقعیت بیانگر آن است که کیفیت تحلیل بیش از پیچیدگی مدل، به صحت و انسجام داده ورودی وابسته است.

در این پروژه نیز تمرکز اصلی محتوا و طراحی سامانه آموزشی بر همین مرحله قرار گرفته است.

۲-۱-۲ تعریف کیفیت داده

کیفیت داده (Data Quality) مفهومی چندبعدی است و معمولاً بر اساس شاخص‌های زیر ارزیابی می‌شود:

- **دقت (Accuracy) :** میزان تطابق داده با واقعیت یا منبع اصلی
- **کامل بودن (Completeness) :** نبود مقادیر گمشده یا ناقص
- **سازگاری (Consistency) :** عدم وجود تناقض در میان رکوردها
- **یکتایی (Uniqueness) :** نبود رکوردهای تکراری ناخواسته
- **اعتبار (Validity) :** انطباق داده با قواعد و دامنه‌های تعریف‌شده

پاکسازی داده فرآیندی نظام‌مند برای بهبود این ابعاد کیفیت است و صرفاً به حذف داده‌های مشکل‌دار محدود نمی‌شود.

۲-۱-۳ انواع مسائل رایج در داده‌های خام

داده‌های واقعی معمولاً با مجموعه‌ای از مشکلات ساختاری و محتوایی مواجه‌اند، از جمله:

- رکوردهای تکراری (Duplicate Records)
- مقادیر گمشده (Missing Values)
- ناسازگاری قالبی (Format Inconsistencies)
- داده‌های پرت یا غیرعادی (Outliers / Anomalies)
- نقض قواعد منطقی (Rule Violations)

مدیریت هر یک از این موارد نیازمند ابزارهای تخصصی و رویکردهای تحلیلی مشخص است.

۲-۱-۴ رویکردهای نظام‌مند در پاکسازی داده

پاکسازی داده را می‌توان در قالب یک فرآیند مرحله‌ای مدل‌سازی کرد:

۱. **پروفایل‌سازی داده (Data Profiling) :**
   استخراج شاخص‌های توصیفی برای درک ساختار و الگوهای کلی.
۲. **شناسایی خطا(Error Detection)** :
   کشف تکرارها، مقادیر نامعتبر یا مغایرت‌ها.
۳. **تصحیح یا جانهی(Correction / Imputation)** :
   اصلاح مقادیر، جایگزینی داده‌های گمشده یا حذف رکوردهای نامعتبر.
۴. **اعتبارسنجی(Validation)** :
   بررسی انطباق داده با مجموعه‌ای از قواعد از پیش تعریف‌شده.
۵. **مستندسازی تغییرات:**
   ثبت اصلاحات انجام‌شده برای حفظ شفافیت و قابلیت بازتولید.

در محیط R، بسته‌های متعددی برای پشتیبانی از این مراحل توسعه یافته‌اند که هر یک بخشی از این فرآیند را پوشش می‌دهند.

۲-۱-۵ جایگاه R در مدیریت کیفیت داده

R به‌دلیل ساختار برداری، انعطاف‌پذیری data.frame وtibble ، و اکوسیستم گسترده بسته‌ها، بستری مناسب برای پیاده‌سازی فرآیندهای پاکسازی داده فراهم می‌کند. ابزارهای موجود برای شناسایی تکرار، مدیریت مقادیر گمشده، اعتبارسنجی قواعد و تطبیق رشته‌ای، امکان اجرای تحلیل‌های دقیق کیفیت داده را فراهم می‌سازند.

در چارچوب این پروژه، تمرکز آموزشی بر استفاده نظام‌مند از این ابزارها در قالب سناریوهای عملی قرار گرفته است.

۲-۱-۶ تمرکز پروژه بر پاکسازی داده

اگرچه سامانه طراحی‌شده قابلیت توسعه به سایر حوزه‌های آموزشی R را دارد، در نسخه فعلی حدود ۸۰ درصد محتوای تولیدشده به موضوعات مرتبط با پاکسازی داده و مدیریت کیفیت داده اختصاص یافته است. این تمرکز به‌دلیل نقش راهبردی این مرحله در تحلیل‌های داده‌محور و نیاز سازمانی به کنترل کیفیت داده‌های عملیاتی انتخاب شده است.

۲-۲ WebAssembly و اجرای کد در مرورگر

۲-۲-۱ معرفیWebAssembly

WebAssembly به اختصار (WASM) یک استاندارد باز و سطح پایین برای اجرای کد باینری در محیط مرورگر است که توسط کنسرسیوم W3C توسعه یافته است. هدف اصلی WebAssembly فراهم‌سازی بستری برای اجرای برنامه‌هایی با کارایی نزدیک به کد بومی (Native) در محیط وب است

در معماری سنتی وب، اجرای منطق برنامه در سمت کاربر عمدتاً محدود به JavaScript بود. اگرچه JavaScript زبانی قدرتمند و منعطف است، اما برای محاسبات علمی سنگین یا اجرای زبان‌های برنامه‌نویسی سطح بالا طراحی نشده است WebAssembly .این محدودیت را با معرفی یک فرمت باینری قابل‌حمل، امن و بهینه برای اجرا در موتورهای مرورگر برطرف می‌کند. این ویژگی، WebAssemblyرا به گزینه‌ای مناسب برای اجرای زبان‌های محاسباتی در محیط وب تبدیل کرده است؛ قابلیتی که در این پروژه برای اجرای زبان R در مرورگر مورد استفاده قرار گرفته است.

ویژگی‌های کلیدی WebAssembly عبارت‌اند از:

- **قابلیت حمل (Portability) :** اجرا در تمامی مرورگرهای مدرن بدون وابستگی به سیستم‌عامل؛
- **کارایی بالا:** کامپایل و اجرای نزدیک به سرعت بومی؛
- **امنیت مبتنی بر sandbox :** اجرای کد در محیط ایزوله و کنترل‌شده؛
- **تعامل‌پذیری با JavaScript** : امکان تبادل داده و فراخوانی متقابل بین WASM و JS

۲-۲-۲ معماری اجرایی WebAssembly در مرورگر

در سطح معماری، WebAssemblyبه‌صورت یک ماژول باینری (.wasm) بارگذاری می‌شود. مرورگر این ماژول را:

۱. دانلود می‌کند؛

۲. اعتبارسنجی ساختاری (Validation) انجام می‌دهد؛

۳. به کد داخلی موتور اجرا (Just-In-Time Compilation یا AOT) تبدیل می‌کند؛

۴. در یک محیط ایزوله اجرا می‌کند.

محیط اجرایی WebAssembly شامل مؤلفه‌های زیر است:

- حافظه خطی (Linear Memory) مدیریت‌شده؛
- جدول توابع(Function Table) ؛
- واسط واردات و صادرات (Imports/Exports) برای تعامل با محیط میزبان.

برخلاف برنامه‌های بومی، کد WASM دسترسی مستقیم به فایل‌سیستم، شبکه یا پردازش‌های سیستم‌عامل ندارد. تمامی تعاملات تنها از طریق واسط‌های کنترل‌شده JavaScript انجام می‌شود. این طراحی سطح حملات امنیتی را به‌طور قابل توجهی کاهش می‌دهد.

۲-۲-۳ کامپایل زبان‌های سطح بالا به WebAssembly

یکی از قابلیت‌های مهم WebAssembly، امکان کامپایل زبان‌های سطح بالا مانند C، C++ و Rust به فرمت WASM است. این فرآیند معمولاً با استفاده از زنجیره ابزارهایی مانند LLVM انجام می‌شود.

در این مسیر، کد منبع زبان سطح بالا ابتدا به کد میانی (Intermediate Representation) و سپس به باینری WebAssembly تبدیل می‌شود. این سازوکار باعث شده است زبان‌هایی که ذاتاً برای اجرا در مرورگر طراحی نشده‌اند، بتوانند در محیط وب اجرا شوند.

۲-۲-۴ اجرای زبان R در بستر WebAssembly

زبان R در هسته خود مبتنی بر C نوشته شده است. با استفاده از ابزارهای کامپایل، این هسته می‌تواند به WebAssembly تبدیل شود و در مرورگر اجرا گردد. این رویکرد اساس فناوری WebR را تشکیل می‌دهد.

در این معماری:

- مفسر R به‌صورت یک ماژول WASM در مرورگر بارگذاری می‌شود؛
- حافظه و محیط اجرایی R در داخل sandbox مرورگر ایجاد می‌شود؛
- دستورات R از طریق JavaScript به این مفسر ارسال می‌شود؛
- خروجی‌ها مجدداً به JavaScript بازگردانده می‌شوند و

در(DOM : Document Object Model) نمایش داده می‌شوند.

به این ترتیب، اجرای R کاملاً در سمت کاربر انجام می‌شود و نیازی به ارسال کد به سرور وجود ندارد.

۲-۲-۵ مدل امنیتی و ایزولاسیون

اجرای کد کاربر همواره با ملاحظات امنیتی همراه است. WebAssembly به‌طور پیش‌فرض در محیطی ایزوله اجرا می‌شود که

- دسترسی مستقیم به فایل‌سیستم سیستم‌عامل ندارد؛
- امکان اجرای دستورات سطح پایین سیستم را فراهم نمی‌کند؛
- تنها از طریق API های تعریف‌شده می‌تواند با محیط میزبان تعامل داشته باشد.

مدل sandbox باعث می‌شود اجرای کد R کاربر در مرورگر، خطر دسترسی غیرمجاز به منابع سیستم را نداشته باشد. حتی در صورت ورود کد مخرب، دامنه تأثیر آن محدود به محیط مجازی WebAssembly خواهد بود.

۲-۲-۶ مدیریت حافظه و محدودیت‌های اجرایی

WebAssembly از یک مدل حافظه خطی استفاده می‌کند که اندازه آن در زمان اجرا قابل افزایش است، اما همچنان تحت محدودیت‌های مرورگر قرار دارد. بنابراین:

- پردازش مجموعه‌داده‌های بسیار بزرگ ممکن است با محدودیت حافظه مواجه شود؛
- کارایی وابسته به توان پردازشی دستگاه کاربر است؛
- برخی کتابخانه‌های وابسته به امکانات سیستم‌عامل ممکن است قابل استفاده نباشند.

این محدودیت‌ها در طراحی سامانه آموزشی مدنظر قرار گرفته‌اند و دامنه تمرین‌ها بر مجموعه‌داده‌های با اندازه کنترل‌شده متمرکز شده است.

۲-۲-۷ مقایسه اجرای مبتنی بر WebAssembly با معماری سرورمحور

در معماری سنتی آموزش تعاملی برنامه‌نویسی:

کاربر ← ارسال کد به سرور ← اجرا در محیط مرکزی ← بازگرداندن خروجی

در این مدل، هر درخواست کاربر مستلزم تخصیص منابع پردازشی مرکزی، مدیریت بار سیستم و کنترل هم‌زمانی کاربران است که پیچیدگی عملیاتی و هزینه نگهداری را افزایش می‌دهد.

در مقابل، در معماری مبتنی برWebAssembly :

کاربر ← اجرای مستقیم کد در مرورگر ← نمایش خروجی محلی

در این رویکرد، پردازش به‌صورت توزیع‌شده و در سمت کاربر انجام می‌شود. هر کاربر از منابع دستگاه خود استفاده می‌کند و بار سامانه مرکزی به حداقل می‌رسد. این ویژگی موجب مقیاس‌پذیری ذاتی، کاهش تأخیر شبکه و سادگی استقرار سامانه در محیط‌های سازمانی می‌شود.

۲-۳ معرفی WebR و سازوکار اجرایی آن در این پروژه

۲-۳-۱ معرفی WebR

WebRپیاده‌سازی زبان برنامه‌نویسی R در بستر WebAssembly است که امکان اجرای مفسر R را به‌صورت کامل در مرورگر فراهم می‌کند. در این معماری، هسته زبان R به کد WebAssembly کامپایل شده و به‌عنوان یک ماژول باینری در محیط مرورگر بارگذاری می‌شود.

WebR یک اجرای واقعی (Real R Runtime) از زبان R ارائه می‌دهد، نه یک شبیه‌ساز ساده. بدین معنا که ساختارهای داده، سیستم بسته‌ها، مفسر دستورات و مدیریت حافظه R در همان چارچوب اصلی خود اجرا می‌شوند، با این تفاوت که محیط اجرا در sandbox مرورگر قرار دارد.

۲-۳-۲ معماری داخلیWebR

WebR از سه لایه اصلی تشکیل شده است:

۱. **لایهWebAssembly** :

شامل هسته کامپایل‌شده R که در محیط WASM اجرا می‌شود.

۲. **لایهJavaScript Wrapper** :

یک واسط جاوااسکریپتی که امکان تعامل با مفسر R را فراهم می‌کند (مانندevalR ، installPackages و مدیریت خروجی).

۳. **لایه رابط کاربر(UI Integration)** :

که در پروژه حاضر با استفاده از Bookdown و عناصر HTML برای دریافت کد کاربر و نمایش خروجی پیاده‌سازی شده است.

تعامل میان JavaScript و R از طریق پیام‌های غیرهمزمان (asynchronous) انجام می‌شود. بدین ترتیب، اجرای دستورات R در مرورگر باعث قفل شدن رابط کاربری نمی‌شود.

۳-۳-۲ فرآیند مقداردهی اولیه WebR در پروژه

در این پروژه، WebRاز طریق بارگذاری ماژول ES6 از منبع رسمی آن مقداردهی اولیه می‌شود:

```
import { WebR } from "https://webr.r-wasm.org/latest/webr.mjs";
```

پس از ایجاد نمونه سراسری(global instance) ، فرآیند راه‌اندازی شامل مراحل زیر است:

۱. **Init()**
بارگذاری و فعال‌سازی مفسر R در محیطWASM

۲. **flush()**
پاکسازی بافرهای اولیه و آماده‌سازی محیط.

۳. **installPackages()**
نصب بسته‌های مورد نیاز آموزشی در فایل‌سیستم مجازیWebR

۴. **library()**
بارگذاری بسته‌ها برای استفاده در تمرین‌های تعاملی.

۵. پس از تکمیل این مراحل، وضعیت آماده بودن محیط با یک متغیر کنترلی (webRReady) مشخص می‌شود تا از اجرای زودهنگام کد کاربر جلوگیری شود.

۲-۳-۴ مدیریت بسته‌ها در محیط WebR

در WebR ، نصب بسته‌ها در یک فایل‌سیستم مجازی (Virtual File System) در حافظه مرورگر انجام می‌شود. در صورت پشتیبانی مرورگر، این بسته‌ها می‌توانند در IndexedDB ذخیره شوند تا در بارگذاری‌های بعدی مجدداً دانلود نشوند.

در این پروژه، مجموعه‌ای از بسته‌های تخصصی مرتبط با پاکسازی داده در مرحله راه‌اندازی نصب و بارگذاری می‌شوند تا محیط تمرینی یکپارچه فراهم گردد.

برای افزایش پایداری، بارگذاری کتابخانه‌ها در بلوک‌های محافظت‌شده
suppressPackageStartupMessages انجام می‌شود تا در صورت بازنشانی محیط، تمرین‌ها

همچنان قابل اجرا باشند .

۲-۳-۵ سازوکار اجرای کد کاربر

اجرای کد کاربر در این پروژه بر اساس یک الگوی کنترلی چندمرحله‌ای انجام می‌شود:

۱. دریافت کد از textarea ؛
۲. بررسی آماده بودن محیط WebR ؛
۳. ارسال کد به مفسر R از طریق evalR ؛
۴. اجرای دستورات در بلوک tryCatch برای مدیریت خطا؛
۵. هدایت خروجی به یک فایل موقت در محیط مجازی(tmp) ؛
۶. خواندن خروجی و بازگرداندن آن به JavaScript ؛
۷. نمایش نتیجه در صفحه.

اجرای خط‌به‌خط با استفاده از parse و eval امکان شناسایی جداگانه خطاها و چاپ مستقل خروجی هر دستور را فراهم می‌کند. این سازوکار یک محیط اجرایی ایزوله و کنترل‌شده در سمت کاربر ایجاد می‌کند که بدون نیاز به ارتباط با سرور عمل می‌کند.

۲-۳-۶ مدیریت خطا و پایداری سیستم

برای جلوگیری از اختلال در فرآیند یادگیری، اجرای کد درون ساختار tryCatch قرار گرفته است. در صورت بروز خطا:

- پیام خطا استخراج می‌شود؛
- اجرای کلی سیستم متوقف نمی‌شود؛
- کاربر بازخورد فوری دریافت می‌کند.

همچنین بررسی خالی نبودن خروجی و ارائه پیام‌های راهنما از طریق JavaScript انجام می‌شود تا تجربه کاربری ساختاریافته باقی بماند.

۲-۳-۷ تعامل WebR باBookdown

در این پروژه، فایل پیکربندی WebR از طریق بخش includes: in_header در خروجی Bookdown بارگذاری می‌شود. بدین ترتیب:

- WebRپیش از تکمیل رندر صفحه فعال می‌شود؛
- تمامی فصل‌ها به نمونه سراسری WebR دسترسی دارند؛
- تمرین‌های تعاملی بدون راه‌اندازی مجدد محیط اجرا می‌شوند.

این یکپارچه‌سازی موجب انسجام میان ساختار کتاب و موتور اجرای R شده است.

۲-۳-۸ مزایا و محدودیت‌های استفاده از WebR در این پروژه

مزایا:

- حذف نیاز به سرور محاسباتی مرکزی؛
- مقیاس‌پذیری مبتنی بر منابع کاربر؛
- کاهش تأخیر شبکه؛
- امنیت مبتنی برsandbox ؛
- امکان استقرار در محیط‌های سازمانی محدود.

محدودیت‌ها:

- وابستگی به توان پردازشی دستگاه کاربر؛
- محدودیت حافظه مرورگر؛
- زمان اولیه برای دانلود و آماده‌سازی بسته‌ها.
- در طراحی سامانه، مثال‌ها و تمرین‌ها متناسب با این محدودیت‌ها انتخاب شده‌اند.

۲-۴ معماری کلی سامانه آموزشی و جریان اجرای تمرین‌ها

۲-۴-۱ هدف طراحی معماری

معماری سامانه به‌گونه‌ای طراحی شده است که محتوای آموزشی و تمرین عملی در یک محیط واحد ارائه شوند، در حالی‌که اجرای کد به زیرساخت سرور وابسته نباشد. در این رویکرد، پردازش محاسباتی به سمت کاربر منتقل شده و مرورگر نقش محیط اجرای R را ایفا می‌کند. این تصمیم معماری موجب حذف لایه سرور محاسباتی از طراحی سامانه، کاهش پیچیدگی عملیاتی و تسهیل استقرار در محیط‌های سازمانی می‌شود.

۲-۴-۲ مؤلفه‌های اصلی سامانه

سامانه از چهار مؤلفه اصلی تشکیل شده است:

۱. **لایه محتوا(Content Layer)** :

شامل صفحات آموزشی تولیدشده که متن، مثال‌ها، کدهای نمونه و تمرین‌ها را در قالب HTML ارائه می‌کنند.

۲. **لایه رابط تمرین(Exercise UI Layer)** :

شامل عناصر HTML مانندtextarea ، دکمه اجرا، بخش نمایش خروجی و بخش راهنما (Hint) که تعامل کاربر را مدیریت می‌کنند.

۳. **لایه کنترل و منطق اجرا(Execution Controller Layer)** :

شامل اسکریپت‌های JavaScript که وضعیت آماده بودن محیط، دریافت کد کاربر، مدیریت خطا و نمایش خروجی را بر عهده دارند.

۴. **لایه موتور محاسباتی(R Runtime Layer)** :

شامل WebR به‌عنوان مفسر R در مرورگر که بر پایه WebAssembly اجرا می‌شود و دستورات کاربر را ارزیابی می‌کند.

این تفکیک لایه‌ای باعث می‌شود توسعه سامانه (افزودن فصل، افزودن تمرین یا تغییر ظاهر) بدون دستکاری هسته اجرایی امکان‌پذیر باشد.

۲-۴-۳ جریان کلی داده و کنترل(Control & Data Flow)

فرآیند اجرای یک تمرین تعاملی به‌صورت زیر مدل‌سازی می‌شود:

- بارگذاری صفحه آموزشی
- راه‌اندازی و مقداردهی اولیهWebR
- ورود کد توسط کاربر
- اعتبارسنجی اولیه درJavaScript
- ارسال کد به موتور R و اجرای کنترل‌شده
- استخراج خروجی و پیام‌های خطا
- نمایش نتیجه در پنل خروجی

این جریان امکان طی چرخه تعاملی «نوشتن–اجرا–مشاهده–اصلاح» را بدون ترک محیط آموزشی فراهم می‌کند.

۲-۴-۴ مدل اجرای سمت کاربر و پیامدهای مقیاس‌پذیری

در معماری‌های سرورمحور، هر کاربر نیازمند یک محیط اجرایی مرکزی، مدیریت نشست (Session Management) و کنترل همزمانی است. افزایش تعداد کاربران مستقیماً به افزایش بار زیرساخت منجر می‌شود.

در سامانه حاضر، اجرای کد به‌صورت کامل در سمت کاربر انجام می‌شود. در نتیجه:

- مدیریت نشست در سطح سرور حذف می‌شود؛
- نیاز به تخصیص منابع پردازشی مرکزی وجود ندارد؛
- وضعیت اجرایی هر کاربر مستقل از سایر کاربران است؛
- گلوگاه سرور در مقیاس‌های بزرگ شکل نمی‌گیرد.

این ویژگی سامانه را از نظر مقیاس‌پذیری برای کاربردهای آموزشی گسترده مناسب می‌سازد.

۲-۴-۵ ملاحظات امنیتی در معماری

امنیت در این سامانه در سطح معماری تضمین شده است. از آنجا که اجرای کد در محیط ایزوله مرورگر انجام می‌شود:

- دسترسی مستقیم به منابع سیستم‌عامل وجود ندارد؛
- تعامل با محیط خارج از sandbox تنها از طریق API های کنترل‌شده انجام می‌شود؛
- اجرای کد کاربر اثری بر سایر کاربران یا زیرساخت مرکزی ندارد.

بدین ترتیب، ریسک‌های مرتبط با اجرای کد کاربر در مقایسه با معماری‌های سرورمحور به‌طور معناداری کاهش می‌یابد.

۲-۵ معرفی Bookdown و ساختار تولید کتاب تعاملی

۲-۵-۱ معرفی Bookdown و مسئله‌ای که حل می‌کند

Bookdown یک چارچوب مبتنی بر RMarkdown برای تولید اسناد چندفصلی و کتاب‌های الکترونیکی است. این چارچوب امکان سازمان‌دهی محتوای طولانی در قالب فصل و بخش، شماره‌گذاری خودکار، ایجاد ارجاع‌های درون‌متنی و تولید خروجی در قالب‌های مختلف مانند HTML و PDF را فراهم می‌کند.

در پروژه‌هایی که محتوای آموزشی از یک فایل منفرد فراتر می‌رود، مدیریت ساختار، پیونددهی و یکپارچگی خروجی اهمیت پیدا می‌کند. در پروژه حاضر، مسئله اصلی تولید و انتشار یک محتوای آموزشی چندفصلی همراه با تمرین‌های تعاملی است Bookdown .بستری فراهم می‌کند که در آن:

فصل‌ها به‌صورت ماژولار و مستقل توسعه یابند؛

- کل سامانه با یک فرآیند ساخت منسجم به خروجی واحد تبدیل شود؛
- ناوبری، فهرست مطالب و پیوندهای داخلی به‌صورت خودکار تولید شوند؛
- اجزای مشترک (CSS و JavaScript) به‌صورت سراسری در تمام صفحات اعمال شوند.

۲-۵-۲ دلایل انتخاب Bookdown در معماری سامانه

انتخاب Bookdown در این پروژه مبتنی بر ملاحظات معماری و ساختاری بوده است:

۱. **مدیریت ساختار چندفصلی استاندارد**

آموزش پاکسازی داده ماهیتی چندبخشی دارد و شامل مفاهیم، ابزارها، سناریوها و تمرین‌هاست . Bookdownاین ساختار را در قالبی استاندارد و قابل توسعه مدیریت می‌کند.

۲. **پشتیبانی از خروجی وب با ناوبری آموزشی مناسب**

قالب‌های کتاب‌محور امکان مطالعه پیوسته، دسترسی سریع به فصل‌ها و تجربه کاربری منسجم را فراهم می‌کنند.

۳. **امکان بارگذاری سراسری منابع مشترک**

برای فعال‌سازی WebR در تمام فصل‌های دارای تمرین تعاملی، لازم است فایل‌های CSS و JavaScript به‌صورت سراسری بارگذاری شوند. Bookdown این قابلیت را از طریق تنظیمات includes فراهم می‌کند.

۲-۵-۳ ساختار ورودی پروژه در Bookdown

در ساختار متعارف Bookdown، پروژه شامل اجزای زیر است:

- یک فایل اصلی برای تنظیمات و متادیتا (مانند index.Rmd)
- مجموعه فایل‌های فصل‌ها (هر فصل یک فایل Rmd مستقل)
- فایل‌های منابع شامل CSS، فونت‌ها و JavaScript
- تنظیمات خروجی و مسیر انتشار

در پروژه حاضر، فایل index.Rmdنقطه شروع ساخت سامانه است و اطلاعاتی مانند عنوان، نویسندگان، نوع خروجی و فایل‌های سراسری در آن تعریف شده‌اند. این ساختار ماژولار امکان توسعه تدریجی فصل‌ها را بدون اختلال در کل پروژه فراهم می‌کند.

۲-۵-۴ قالب خروجی و دلیل انتخابGitBook

Bookdown قالب‌های خروجی متعددی ارائه می‌دهد. در این پروژه، خروجی وب از نوع GitBook انتخاب شده است، زیرا:

- ناوبری فصل‌به‌فصل و فهرست مطالب تعاملی ارائه می‌دهد؛
- ساختار کتاب‌محور آن برای آموزش مناسب است؛
- نمایش کدها، جداول و بلوک‌های چندرسانه‌ای را به‌صورت منظم پشتیبانی می‌کند؛
- امکان یکپارچه‌سازی فایل‌های CSS و JavaScript در سطح کل کتاب را ساده می‌سازد.

این انتخاب به ایجاد تجربه‌ای منسجم میان متن آموزشی و تمرین‌های تعاملی کمک کرده است.

۲-۵-۵ بارگذاری سراسری فایل‌های مشترک (CSS/JS)

یکی از نیازهای کلیدی سامانه، فعال بودن موتور اجرای WebR و تنظیمات ظاهری در تمام فصل‌هاست. این امر از طریق تنظیمات includes: in_header در Bookdown انجام شده است.

در این بخش:

- فایل‌های CSS برای پشتیبانی از راست‌به‌چپ و فونت فارسی بارگذاری می‌شوند؛
- فایل پیکربندی WebR (مانند webr-setup.html) در تمام صفحات درج می‌شود؛
- توابع اجرایی تمرین‌ها در سطح کل سامانه در دسترس قرار می‌گیرند.

مزیت این رویکرد آن است که هر فصل تنها شامل عناصر تمرینی مربوط به خود است، در حالی‌که موتور اجرا و تنظیمات ظاهری به‌صورت سراسری مدیریت می‌شوند. این تفکیک نگهداری و توسعه پروژه را ساده می‌کند.

۲-۵-۶ سازگاری با زبان فارسی و راست به چپ

در خروجی‌های پیش‌فرض برخی قالب‌های وب، پشتیبانی از راست‌به‌چپ و فونت فارسی کامل و بی‌نقص نیست. در پروژه حاضر، برای افزایش خوانایی و استانداردسازی نمایش فارسی:

- فونت اختصاصی تعریف و فراخوانی شده است؛
- جهت‌نمایی (direction) و چینش عناصر ناوبری اصلاح شده است؛
- برخی اجزای قالب GitBook برای سازگاری با RTL بازنویسی شده‌اند.

این بخش از طراحی برای پروژه‌هایی که مخاطب فارسی‌زبان دارند اهمیت عملی دارد و باعث می‌شود سامانه آموزشی در محیط واقعی قابل استفاده باشد، نه صرفاً در حالت آزمایشی.

۲-۵-۷ تنظیمات انتشار و ساخت خروجی

در Bookdown می‌توان مسیر تولید خروجی را تعیین کرد تا نسخه ساخته شده پروژه در یک پوشه مشخص ایجاد شود. در این پروژه، خروجی در مسیر تعریف‌شده‌ای تولید می‌شود که امکان استقرار مستقیم به‌عنوان یک وب‌سایت استاتیک را فراهم می‌کند.

از آنجا که اجرای کد در سمت کاربر انجام می‌شود و سامانه وابسته به بک‌اند نیست، خروجی نهایی می‌تواند بدون نیاز به سرور محاسباتی و تنها با میزبانی فایل‌های استاتیک منتشر شود. این ویژگی از منظر عملیاتی، استقرار سامانه را ساده و کم‌هزینه می‌کند.

فصل سوم

پیاده‌سازی و جزئیات فنی سامانه

۳-۱ محیط اجرا و قیود طراحی

در این سامانه، محیط اجرای برنامه به‌صورت کامل در مرورگر تعریف شده است. بدین معنا که نه سیستم‌عامل کاربر و نه یک سرور مرکزی، بلکه موتور اجرای مرورگر بستر اصلی پردازش را تشکیل می‌دهد. این تصمیم معماری مجموعه‌ای از قیود طراحی فنی را ایجاد می‌کند که در پیاده‌سازی سامانه مدنظر قرار گرفته‌اند.

مهم‌ترین این قیود عبارت‌اند از:

- اجرای کد R در چارچوب WebAssembly و در محیط ایزوله (sandbox) مرورگر؛
- محدود بودن مدیریت فایل‌ها و بسته‌ها به فایل‌سیستم مجازی داخلی؛
- لزوم کنترل زمان راه‌اندازی اولیه و دانلود وابستگی‌ها؛
- مدیریت کامل تعاملات کاربر (ورود کد، اجرای دستور، نمایش خروجی و راهنما) در ساختار DOM صفحه.

این قیود موجب شده‌اند طراحی سامانه بر اصل «تفکیک هسته اجرا از محتوای آموزشی» استوار باشد. به‌گونه‌ای که موتور اجرای تمرین‌ها به‌صورت مستقل توسعه یافته و هر فصل آموزشی صرفاً از طریق فراخوانی توابع عمومی تعریف‌شده، قابلیت اجرای تعاملی پیدا می‌کند. این رویکرد علاوه بر کاهش وابستگی میان بخش‌ها، توسعه و نگهداری سامانه را ساده‌تر می‌سازد.

۳-۲ ساختار فیزیکی پروژه و سازمان‌دهی فایل‌ها

پروژه به‌صورت یک ساختار ماژولار مبتنی بر Bookdown پیاده‌سازی شده است. این ساختار به‌گونه‌ای طراحی شده که تفکیک میان محتوا، تنظیمات ساخت، و زیرساخت تعاملی به‌صورت شفاف رعایت شود.

در ریشه پروژه، فایل index.Rmd به‌عنوان نقطه شروع فرآیند ساخت (Build Entry Point) قرار دارد. این فایل شامل تنظیمات خروجی، متادیتا (مانند عنوان، نویسندگان و نوع خروجی) و همچنین ارجاع به فایل‌های مشترکی است که در تمامی صفحات کتاب بارگذاری می‌شوند.

فایل bookdown.yml_ مسئول مدیریت تنظیمات کلی ساخت پروژه است. در این فایل ترتیب فصل‌ها، مسیر تولید خروجی و تنظیمات مرتبط با انتشار تعیین می‌شود. این تفکیک تنظیمات از محتوا موجب می‌شود تغییر در ساختار کلی سامانه بدون دستکاری فایل‌های آموزشی انجام شود.

هر فصل آموزشی در قالب یک فایل مستقل RMarkdown توسعه یافته است. نام‌گذاری فایل‌ها به‌صورت ترتیبی و عددی (مانند 001-ch01.Rmd , 002-ch02.Rmd و ...) انجام شده است. این رویکرد دو مزیت اصلی دارد:

- حفظ ترتیب منطقی فصل‌ها در فرآیند ساخت؛
- امکان افزودن فصل‌های جدید بدون ایجاد اختلال در ساختار موجود.

علاوه بر فایل‌های محتوایی، مجموعه‌ای از فایل‌های زیرساخت تعاملی در پروژه وجود دارد که به‌صورت سراسری در تمام صفحات کتاب بارگذاری می‌شوند:

- webr-setup.htmlبرای راه‌اندازی و مقداردهی اولیه موتورWebR ؛
- webr-setup-vim.html (در صورت استفاده از ویرایشگر پیشرفته)؛
- custom-rtl-style.css برای اصلاح راست‌به‌چپ، فونت فارسی و سفارشی‌سازی ظاهر قالب.

این فایل‌های مشترک از طریق تنظیمات خروجی Bookdown در سطح کل سامانه اعمال می‌شوند؛ به‌گونه‌ای که هر فصل بدون نیاز به تعریف مجدد زیرساخت اجرایی، تنها با افزودن عناصر تمرینی مربوط به خود، قابلیت تعاملی پیدا می‌کند.

چنین سازمان‌دهی‌ای توسعه‌پذیری سامانه را افزایش داده و نگهداری آن را ساده‌تر می‌کند، زیرا تغییر در موتور اجرا یا ظاهر کلی سامانه، مستقل از محتوای فصل‌ها انجام می‌شود.

```
project-root/
|
├── index.Rmd
├── _bookdown.yml
├── 001-ch01.Rmd
├── 002-ch02.Rmd
├── 003-ch03.Rmd
|
├── webr-setup.html
├── webr-setup-vim.html
├── custom-rtl-style.css
|
├── _book/          (خروجی موقت Bookdown)
├── docs/           (خروجی نهایی انتشار)
|
├── .R
├── .RData
├── .Rhistory
└── _main.html
```

۳-۳ پیکربندی خروجی و تزریق زیرساخت تعاملی

در این پروژه، تنظیمات خروجی در فایل index.Rmd تعریف شده و قالب اصلی خروجی bookdown::gitbook انتخاب شده است. این قالب علاوه بر ساختار کتاب‌محور، امکان درج فایل‌های سفارشی در بخش <head> تمامی صفحات خروجی را فراهم می‌کند. این قابلیت نقش مهمی در یکپارچه‌سازی زیرساخت تعاملی سامانه دارد.

۳-۳-۱ انتخاب قالب GitBook

در تنظیمات خروجی:

- highlight: tango برای یکسان‌سازی رنگ‌بندی بلوک‌های کد انتخاب شده است؛
- toc_depth: ۲ برای کنترل عمق نمایش سرفصل‌ها در فهرست مطالب تنظیم شده است؛
- number_sections: true برای فعال‌سازی شماره‌گذاری ساختاریافته در خروجی HTML اعمال شده است.

انتخاب GitBook به دلیل ناوبری سمت چپ، ساختار فصل‌محور منظم و امکان بارگذاری فایل‌های سفارشی در سطح کل کتاب انجام شده است. این ویژگی‌ها آن را برای یک سامانه آموزشی تعاملی مناسب می‌سازد.

۳-۳-۲ بارگذاری فایل‌های مشترک در بخش header

در بخش تنظیمات includes: in_header سه فایل به‌صورت سراسری در تمامی صفحات خروجی بارگذاری می‌شوند. این تصمیم طراحی در معماری سامانه نقش کلیدی دارد، زیرا زیرساخت تعاملی را مستقل از محتوای فصل‌ها نگه می‌دارد.

```
includes:
  in_header:
  - araku-mirkamali-rtl-4.css
  - webr-setup.html
  - webr-setup-vim.html
```

۱. فایل CSS سفارشی (RTL و فونت)

فایل custom-rtl-style.css مسئول سازگارسازی قالب GitBook با زبان فارسی است. این فایل:

- جهت نوشتار را به RTL تنظیم می‌کند؛
- چیدمان ناوبری را اصلاح می‌کند؛
- فونت فارسی را اعمال می‌کند؛
- ظاهر جداول و منوها را هماهنگ می‌سازد.

بدون این لایه تنظیمی، نمایش خروجی برای مخاطب فارسی‌زبان از نظر بصری ناسازگار می‌بود.

۲. فایل راه‌اندازی WebR

- فایل webr-setup.html هسته اجرایی تعاملی سامانه است. این فایل:
- موتور WebR را بارگذاری می‌کند؛
- نمونه سراسری runtime را ایجاد می‌کند؛
- بسته‌های آموزشی مورد نیاز را نصب و بارگذاری می‌کند؛
- توابع عمومی اجرای تمرین‌ها را تعریف می‌کند.

قرار گرفتن این فایل در بخش header موجب می‌شود زیرساخت اجرایی پیش از هرگونه تعامل کاربر فعال شده باشد و تمامی فصل‌ها به یک موتور مشترک دسترسی داشته باشند.

۳-۳-۳ جداسازی خروجی انتشار

در تنظیمات پروژه، مسیر انتشار به پوشه docsتعیین شده است. این تصمیم دو مزیت عملی دارد:

- امکان استقرار مستقیم پروژه به‌عنوان وب‌سایت استاتیک؛
- جداسازی خروجی نهایی از پوشه‌های ساخت موقت مانند .book_

در نتیجه، پروژه بدون نیاز به سرور دینامیک، قابلیت میزبانی در سرویس‌های استاتیک را دارد.

۳-۳-۴ تحلیل طراحی

پیکربندی انجام‌شده نشان می‌دهد که:

- زیرساخت تعاملی به‌صورت مرکزی مدیریت می‌شود؛
- فصل‌ها صرفاً مسئول تعریف محتوا و عناصر تمرینی هستند؛
- تغییر در موتور اجرا یا تنظیمات ظاهری نیازمند اصلاح فایل‌های فصول نیست.

این جداسازی مسئولیت‌ها (Separation of Concerns) یکی از اصول طراحی سامانه‌های پایدار است و در این پروژه به‌صورت عملی پیاده‌سازی شده است.

۳-۴ تحلیل سازوکار راه‌اندازی WebR و چرخه عمر موتور اجرایی

۳-۴-۱ بارگذاری ماژولWebR

راه‌اندازی موتور R در مرورگر با استفاده از ماژول ES6 انجام می‌شود:

```
import { WebR } from "https://webr.r-wasm.org/latest/webr.mjs";
```

در این مرحله:

- فایل WebAssembly مربوط به هسته R دانلود می‌شود؛
- لایه JavaScript واسط برای تعامل با موتور بارگذاری می‌گردد؛
- محیط اجرای R در حافظه مرورگر ایجاد می‌شود.

استفاده از مسیر latest موجب دسترسی به نسخه به‌روز WebR می‌شود. با این حال، این انتخاب وابستگی نسخه‌ای ایجاد می‌کند؛ به این معنا که در صورت اعمال تغییرات ناسازگار در نسخه‌های آتی، نیاز به بازبینی تنظیمات سامانه وجود خواهد داشت. این موضوع یکی از ملاحظات نگهداری بلندمدت سامانه محسوب می‌شود.

۳-۴-۲ ایجاد نمونه سراسری و کنترل وضعیت اجرا

```
globalThis.webR = new WebR();
globalThis.webRReady = false;
```

در این طراحی:

در طراحی سامانه، نمونه‌ای یکتا از موتور WebR در محدوده سراسری (Global Scope) ایجاد می‌شود. تمامی تمرین‌های تعاملی در فصل‌های مختلف به همین نمونه مشترک متصل هستند.

همزمان، یک متغیر کنترلی مانند (webRReady) برای مدیریت وضعیت آماده بودن موتور تعریف شده است. این متغیر نقش قفل اجرایی (Execution Lock) را ایفا می‌کند و از اجرای کد پیش از تکمیل فرآیند مقداردهی اولیه جلوگیری می‌کند.

این تصمیم طراحی دو مزیت اصلی دارد:

۱. جلوگیری از راه‌اندازی مجدد موتور در هر تمرین، که موجب کاهش سربار پردازشی و تأخیر می‌شود؛
۲. تضمین این‌که اجرای کد کاربر تنها پس از نصب و بارگذاری کامل بسته‌های مورد نیاز انجام گیرد.

بدین ترتیب، چرخه اجرای تمرین‌ها به یک موتور مرکزی وابسته است، نه نمونه‌های پراکنده.

۳-۴-۳ مقداردهی اولیه و آماده‌سازی محیط

فرآیند آماده‌سازی موتور در یک تابع اختصاصی (مانند initWebR) به‌صورت مرحله‌ای اجرا می‌شود.

مرحله ۱: فعال‌سازیruntime

در این مرحله، هسته R در محیط WASM فعال شده و بافرهای اولیه پاکسازی می‌شوند. این گام، موتور را از وضعیت بارگذاری اولیه به وضعیت عملیاتی منتقل می‌کند.

مرحله ۲: نصب بسته‌ها

```
await webR.installPackages([...]);
```

مجموعه‌ای از بسته‌های تخصصی پاکسازی داده در همان مرحله راه‌اندازی نصب می‌شوند. این تصمیم طراحی باعث می‌شود:

- تمامی تمرین‌ها به کتابخانه‌های مورد نیاز دسترسی داشته باشند؛
- کاربر نیازی به نصب دستی بسته‌ها در محیط تعاملی نداشته باشد؛
- وابستگی‌های آموزشی در سطح مرکزی مدیریت شوند.

نصب بسته‌ها در فایل‌سیستم مجازی WebR انجام می‌شود. در صورت پشتیبانی مرورگر، این بسته‌ها در حافظه پایدار مرورگر کش می‌شوند و در بارگذاری‌های بعدی نیاز به دانلود مجدد ندارند. این سازوکار زمان راه‌اندازی در مراجعات بعدی را کاهش می‌دهد.

مرحله ۳: بارگذاری کتابخانه‌ها

```
await globalThis.webR.evalR(`
  suppressPackageStartupMessages({
    library(...)
  })
`);
```

پس از نصب، کتابخانه‌ها در محیط R بارگذاری می‌شوند. استفاده از بلوک‌های محافظت‌شده (مانند suppressPackageStartupMessages) مانع نمایش پیام‌های غیرضروری در خروجی تمرین‌ها می‌شود و تجربه کاربری را یکنواخت نگه می‌دارد.

مرحله ۴: اعلام آماده بودن

```
globalThis.webRReady = true;
```

در پایان فرآیند مقداردهی اولیه، وضعیت آماده بودن موتور ثبت می‌شود. از این لحظه به بعد، سامانه قادر به دریافت و اجرای دستورات کاربر است.

۳-۴-۴ تحلیل چرخه عمر موتور اجرایی

چرخه عمر موتور WebR در این سامانه را می‌توان در سه فاز مدل‌سازی کرد:

۱. **فاز راه‌اندازی:(Initialization Phase)**
   بارگذاری ماژول، فعال‌سازی runtime و نصب وابستگی‌ها.

۲. **فاز عملیاتی:(Operational Phase)**
   دریافت، ارزیابی و اجرای دستورات کاربر در طول استفاده از سامانه.

۳. **فاز پایانی یا بازنشانی:(Reset/Reload Phase)**
   در صورت بارگذاری مجدد صفحه یا پاکسازی حافظه مرورگر، فرآیند راه‌اندازی مجدداً اجرا می‌شود.

این مدل چرخه عمر نشان می‌دهد که موتور اجرایی در سطح صفحه (Page-Level Lifecycle) مدیریت می‌شود و نه در سطح هر تمرین. چنین طراحی‌ای از ایجاد نمونه‌های متعدد و ناسازگاری‌های احتمالی جلوگیری می‌کند و پایداری سامانه را افزایش می‌دهد.

۳-۵ طراحی موتور اجرای تمرین‌ها

این بخش هسته تعاملی سامانه را تشکیل می‌دهد. در اینجا سازوکاری طراحی شده که امکان اجرای کنترل‌شده کد کاربر، مدیریت خطا و نمایش خروجی را بدون وابستگی به سرور فراهم می‌کند.

۳-۵-۱ طراحی تابع عمومی اجرا

تابع زیر به‌صورت سراسری تعریف شده است:

globalThis.runUserCode = async function (...)

این طراحی باعث می‌شود:

- تمرین‌ها فقط شناسه textarea و خروجی را ارسال کنند؛
- منطق اجرا متمرکز و قابل نگهداری باشد؛
- رفتار همه تمرین‌ها یکنواخت باقی بماند.

یک تابع عمومی در محدوده سراسری تعریف شده است (مانند runUserCode). این تابع نقطه ورودی تمام تمرین‌های تعاملی محسوب می‌شود.

طراحی متمرکز این تابع موجب شده است:

تمرین‌ها تنها شناسه عناصر رابط کاربری textarea) و پنل خروجی (را ارسال کنند؛

منطق اجرا در یک نقطه واحد نگهداری شود؛

رفتار تمامی تمرین‌ها یکنواخت و قابل پیش‌بینی باقی بماند؛

اعمال تغییرات یا بهبودهای آینده بدون اصلاح تک‌تک فصل‌ها انجام شود.

این رویکرد نمونه‌ای از اصل تمرکز منطق اجرایی (Centralized Execution Logic) در معماری‌های تعاملی است.

۳-۵-۲ کنترل‌های پیش‌اجرایی

پیش از ارسال کد به موتور WebR، چند بررسی کنترلی انجام می‌شود:

۱. بررسی خالی نبودن ورودی کاربر؛
۲. بررسی آماده بودن موتور اجرایی؛
۳. نمایش پیام وضعیت مناسب در رابط کاربری.

این مرحله از اجرای زودهنگام، ناقص یا ناسازگار جلوگیری می‌کند و یک لایه اعتبارسنجی اولیه در سمت کاربر ایجاد می‌نماید.

۳-۵-۳ اجرای خط‌به‌خط کد کاربر

در بخش زیر:

```
lines <- unlist(strsplit(userCode, "\n"))
for (ln in lines) {
  val <- eval(parse(text=ln))
}
```

برای افزایش پایداری، کد کاربر به خطوط مستقل تقسیم شده و هر خط به‌صورت جداگانه ارزیابی می‌شود. این رویکرد چند مزیت مهم دارد:

- خطای یک دستور باعث توقف کامل اجرای تمرین نمی‌شود؛
- پیام خطا دقیقاً به همان دستور مرتبط می‌گردد؛
- تجربه‌ای مشابه محیط تعاملی R (REPL) ایجاد می‌شود.

این طراحی عمداً از اجرای یک‌باره کل اسکریپت اجتناب می‌کند، زیرا در محیط آموزشی، هدف ارائه بازخورد تدریجی است نه اجرای دسته‌ای.

البته این روش در برابر کدهای چندخطی وابسته (مانند ساختارهای کنترلی چندسطری) نیازمند مدیریت دقیق‌تر است که در طراحی تمرین‌ها مدنظر قرار گرفته است.

۳-۵-۴ مدیریت خروجی با استفاده ازsink

```
sink('/tmp/out.txt')
...
sink()
```

خروجی چاپ‌شده در یک فایل موقت در فایل‌سیستم مجازی ذخیره می‌شود و سپس:

```
paste(readLines('/tmp/out.txt'), collapse="\n")
```

برای یکپارچه‌سازی خروجی‌ها، از مکانیزمی مشابه sink استفاده شده است تا تمامی خروجی‌های چاپ‌شده در یک فایل موقت در فایل‌سیستم مجازی ذخیره شوند. سپس محتوای این فایل به‌صورت یک رشته متنی بازیابی می‌شود.

این طراحی سه مزیت اصلی دارد:

- تجمیع تمامی خروجی‌ها در یک ساختار واحد؛
- امکان ثبت پیام‌های خطا در همان جریان خروجی؛
- جلوگیری از پراکندگی خروجی در محیط داخلی.WebR

به این ترتیب، رابط کاربری تنها یک رشته نهایی دریافت می‌کند و نمایش خروجی ساده‌تر و یکنواخت‌تر می‌شود.

۳-۵-۵ تبدیل خروجی R بهJavaScript

پس از اجرای :evalR

```
const jsResult = await result.toJs();
```

پس از اجرای دستورات در WebR ، نتیجه به ساختار قابل استفاده در JavaScript تبدیل می‌شود. این تبدیل نوع (Type Conversion) مرحله‌ای کلیدی در تعامل میان محیط WebAssembly و JavaScript محسوب می‌شود.

در این مرحله:

- خروجی R از حافظه WASM استخراج می‌شود؛
- به ساختار داده‌ای قابل استفاده در JavaScript تبدیل می‌گردد؛
- در DOM صفحه درج می‌شود.

این پل ارتباطی، امکان تعامل کامل میان موتور محاسباتی و رابط کاربری وب را فراهم می‌کند.

۳-۶ مدیریت خطا و تضمین پایداری سامانه

در سامانه‌های تعاملی که کد کاربر اجرا می‌شود، مدیریت خطا نه یک قابلیت جانبی، بلکه بخشی از معماری اصلی محسوب می‌شود. در این پروژه، خطاها در دو سطح مستقل مدیریت می‌شوند تا از اختلال در کل سامانه جلوگیری شود.

۳-۶-۱ مدیریت خطا در سطح موتورR

در سطح اجرای دستورات R ، ارزیابی کد کاربر در ساختاری محافظت‌شده) مانند (tryCatch انجام می‌شود. این سازوکار امکان می‌دهد:

- بروز خطا در یک دستور، موجب توقف کامل اجرای تمرین نشود؛
- پیام خطا به‌صورت کنترل‌شده استخراج و ثبت گردد؛
- اجرای خطوط بعدی (در صورت امکان) ادامه یابد.

این طراحی به‌ویژه در محیط آموزشی اهمیت دارد، زیرا هدف، ارائه بازخورد تدریجی به کاربر است نه توقف کامل فرآیند یادگیری.

۳-۶-۲ مدیریت خطا در سطحJavaScript

علاوه بر سطح R ، تعامل میان WebAssembly و JavaScript نیز می‌تواند با خطاهای سیستمی مواجه شود (مانند خطای تبدیل نوع، خطای بارگذاری ماژول یا اختلال در ارتباط با موتور).

در این سطح، فراخوانی‌های غیرهمزمان در بلوک‌های try/catch JavaScript مدیریت می‌شوند. این لایه حفاظتی موجب می‌شود:

- خطاهای سیستمی WebR به رابط کاربر منتقل شوند؛
- کل صفحه دچار شکست اجرایی نشود؛
- وضعیت سامانه قابل بازیابی باقی بماند.

۳-۶-۳ تحلیل طراحی پایداری

مدیریت دولایه‌ای خطا نشان می‌دهد که سامانه صرفاً برای اجرای موفق طراحی نشده، بلکه برای «تحمل خطا» نیز آماده است. این رویکرد نوعی تاب‌آوری (Resilience) در سطح رابط کاربر ایجاد می‌کند و از تجربه‌های مخرب مانند قفل شدن صفحه یا توقف کامل اسکریپت جلوگیری می‌نماید.

۳-۷ طراحی رابط تمرین و ملاحظات تجربه کاربری

اگرچه هسته سامانه محاسباتی است، اما تجربه کاربری نقش تعیین‌کننده‌ای در کارآمدی محیط آموزشی دارد. در این پروژه، طراحی رابط تمرین به‌صورت مستقل از منطق اجرایی انجام شده است.

۳-۷-۱ لایه نمایش و طراحی بصری

فایل CSS سفارشی وظیفه مدیریت لایه نمایش را بر عهده دارد. این لایه شامل موارد زیر است:

- تمایز بصری واضح میان متن نظری و بخش تمرین؛
- طراحی دکمه‌های اجرا و راهنما با فرم متمایز؛
- اعمال راست‌به‌چپ کامل برای سازگاری با زبان فارسی؛
- تعریف رنگ‌بندی متمایز برای پیام‌های موفقیت و خطا.

این عناصر به کاربر کمک می‌کنند بدون سردرگمی میان بخش‌های نظری و تعاملی جابه‌جا شود.

۳-۷-۲ جداسازی لایه نمایش از منطق اجرا

فایل‌های CSS به‌صورت مستقل از منطق JavaScript و WebR نگهداری می‌شوند. این جداسازی:

- تغییر ظاهر را بدون دستکاری موتور اجرایی ممکن می‌سازد؛
- توسعه رابط کاربری را مستقل از تغییرات محاسباتی می‌کند؛
- اصل تفکیک لایه نمایش (Presentation Layer) از لایه منطق (Logic Layer) را رعایت می‌کند.

این تصمیم طراحی از اصول پایه معماری سامانه‌های وب تبعیت می‌کند و به پایداری و توسعه‌پذیری بلندمدت سامانه کمک می‌نماید.

فصل چهارم

نمونه پیاده سازی عملی و تحلیل یک فصل به‌عنوان مطالعه موردی

۴-۱ مقدمه و هدف فصل مطالعه موردی

در این فصل، به‌منظور نمایش عملی خروجی سامانه آموزشی طراحی‌شده، یکی از فصل‌های تولیدشده به‌عنوان «مطالعه موردی» انتخاب و تحلیل می‌شود. هدف از این مطالعه موردی، نشان‌دادن هم‌زمان ساختار آموزشی فصل‌ها و سازوکار تعاملی اجرای تمرین‌ها در مرورگر است.

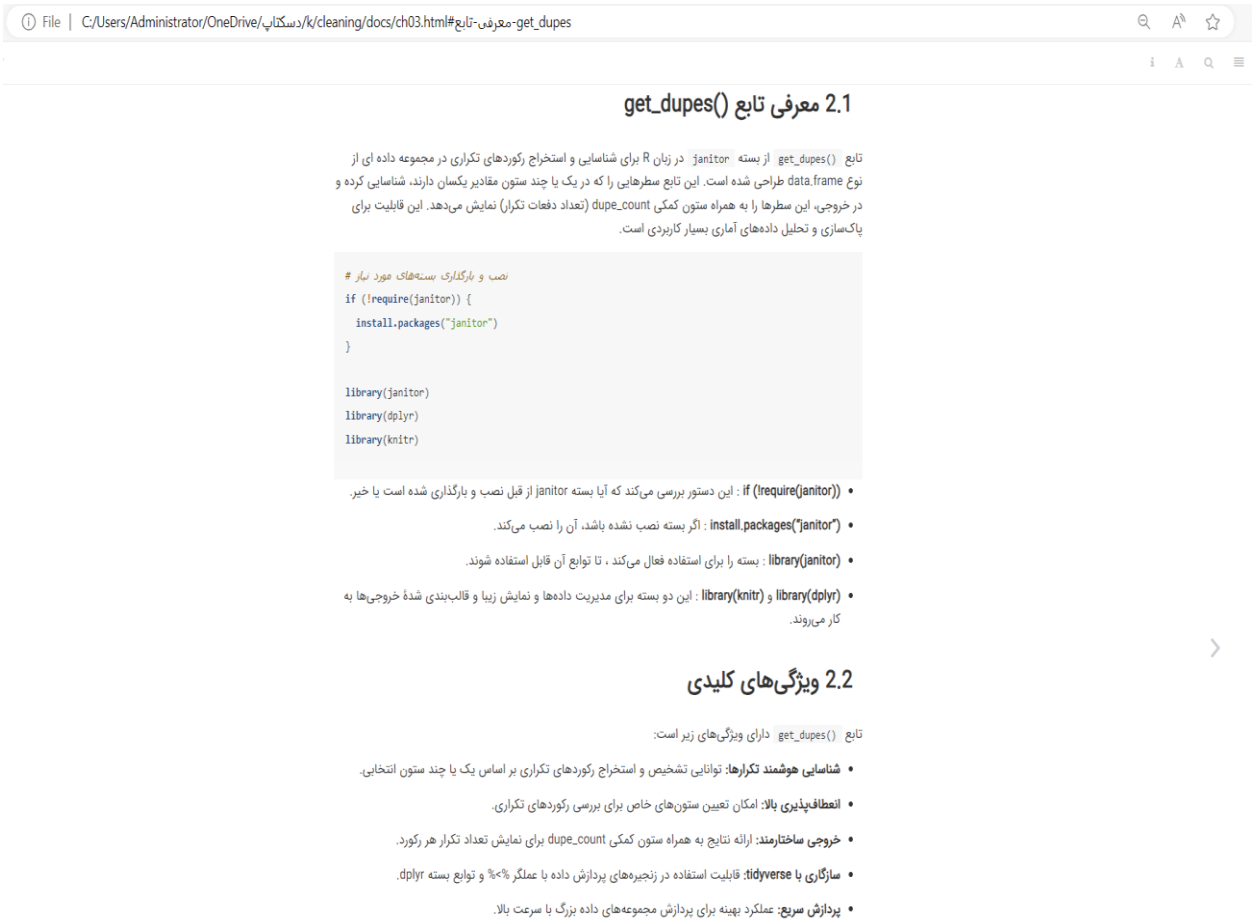
فصل انتخاب‌شده مربوط به معرفی و آموزش تابع get_dupes از بسته janitor در زبان R است. دلیل انتخاب این فصل آن است که تقریباً تمام اجزای الگوی طراحی ماژول‌های آموزشی (مبانی نظری، معرفی پارامترها، مثال‌های واقعی، نکات و خطاهای رایج، تمرین تعاملی) را به‌صورت کامل دربر دارد و برای نمایش قابلیت‌های سامانه مناسب است.

۴-۲ انطباق فصل نمونه با الگوی ثابت طراحی ماژول‌ها

همان‌طور که در فصل‌های قبل بیان شد، ماژول‌های آموزشی سامانه با یک الگوی ثابت طراحی شده‌اند تا انسجام و یادگیری مرحله‌به‌مرحله حفظ شود. در این بخش نشان داده می‌شود که فصل نمونه get_dupes() چگونه این الگو را پوشش می‌دهد.

معرفی مفهومی تابع

در ابتدای فصل، هدف تابع و مسئله‌ای که حل می‌کند به زبان ساده توضیح داده می‌شود.



شکل ۴-۱. معرفی ماژول get_dupes در ساختار فصل آموزشی

در ابتدای فصل، هدف تابع و مسئله‌ای که حل می‌کند به زبان ساده توضیح داده می‌شود.

بیان مبانی نظری و فرمول‌ها

پس از معرفی مفهومی، مبانی نظری مرتبط (تعریف تکرار و نحوه محاسبه تعداد تکرار) ارائه شده است تا یادگیرنده قبل از ورود به اجرا، درک دقیق‌تری از منطق تشخیص تکرار داشته باشد.

2.3 بیان تئوری‌ها و فرمول‌ها

تابع `get_dupes()` بر پایه مفاهیم نظریه مجموعه‌ها و آمار توصیفی کار می‌کند:

2.3.1 تعریف تکرار:

برای مجموعه داده D و ستون‌های مشخص $C = \{c_1, c_2, \ldots, c_k\}$:

یک سطر r_i تکراری محسوب می‌شود اگر حداقل یک سطر دیگر r_j در D وجود داشته باشد که مقادیر آن در تمام ستون‌های C یکسان باشد و تعداد این سطرها بیشتر از یک باشد.

بیان ریاضی :

$$\text{Duplicate}(r_i) = \{r_j \in D : \forall c \in C,\ r_i[c] = r_j[c] \text{ and } |\{r_j\}| > 1\}$$

2.3.2 محاسبه تعداد تکرار:

برای رکورد r_i تعداددفعات تکراری(dupe_count) به این صورت محاسبه می شود:

$$\text{dupe\_count}(r_i) = |\{r_j \in D : \forall c \in C,\ r_i[c] = r_j[c]\}|$$

به عبارت دیگر، dupe_count برابر است با تعداد کل رکوردهایی که در ستون های مشخص شده دقیقا مقادیر یکسانی با رکورد r_i دارند.

شکل ۴-۲. بیان مسئله و تعریف مفهوم رکوردهای تکراری در داده

ساختار تابع و پارامترهای ورودی

در ادامه، قالب کلی تابع و سپس پارامترها همراه با نوع داده ورودی، پیش‌فرض‌ها و توضیحات ارائه شده است. این بخش برای جلوگیری از خطاهای رایج در فراخوانی تابع و افزایش دقت در استفاده از آن ضروری است.

2.4 ساختار و پارامترهای تابع

```
get_dupes(dat, ..., .keep_all = TRUE, .both_ways = TRUE)
```

2.5 پارامترهای ورودی

پارامتر	نوع	پیش‌فرض	الزامی	توضیح
`dat`	data.frame/tibble	-	بله	مجموعه داده‌ای که می‌خواهید رکوردهای تکراری آن را شناسایی کنید. می‌تواند data.frame، tibble یا هر شی قابل تبدیل به data.frame باشد
`...`	column names	تمام ستون‌ها	خیر	نام ستون‌هایی که بر اساس آن‌ها تکرار بررسی می‌شود. اگر مشخص نشود، تمام ستون‌ها در نظر گرفته می‌شوند. می‌توان از bare names یا string استفاده کرد
`keep_all.`	logical	TRUE	خیر	تعیین می‌کند که آیا تمام ستون‌های داده اصلی در خروجی نگه‌داشته شوند (TRUE) یا تنها ستون‌های مشخص شده و ستون dupe_count برگردانده شود (FALSE)
`both_ways.`	logical	TRUE	خیر	تعیین می‌کند که آیا همه نمونه‌های تکراری نمایش داده شوند (TRUE) یا تنها اولین یا آخرین نمونه از هر گروه تکراری برگردانده شود (FALSE)

مثال‌های عملی مرحله‌به‌مرحله

در فصل نمونه، چند مثال با سناریوهای مختلف ارائه شده است (دانشجویی، فروشگاهی، مالی). وجود مثال‌های متنوع باعث می‌شود یادگیرنده کاربرد تابع را در چند حوزه مختلف مشاهده کند و تنها به مثال مصنوعی محدود نشود.

2.6.2 مثال 2: مدیریت محصولات فروشگاه

در این مثال مجموعه داده‌ای از اطلاعات فروش شامل شناسه محصول (product_id)، شناسه مشتری (customer_id)، تاریخ فروش (sale_date)، مبلغ (amount)، و منطقه (region) است.

```
library(pander)

sales_data <- data.frame(
  product_id = c(101, 102, 103, 101, 104, 102),
  customer_id = c("C001", "C002", "C001", "C001", "C003", "C004"),
  sale_date = c("2024-01-01", "2024-01-02", "2024-01-01",
"2024-01-01", "2024-01-03", "2024-01-02"),
  amount = c(1500, 2000, 1200, 1500, 800, 1800),
  region = c("غرب" ,"مرکز" ,"شمال" ,"شمال" ,"جنوب" ,"شمال")
)

# شناسایی تکرارها بر اساس محصول و مشتری
product_customer_dupes <- sales_data %>%
  get_dupes(product_id, customer_id)

kable(product_customer_dupes,caption = "شناسایی تکرارها بر اساس محصول و مشتری")
```

Table 2.2: شناسایی تکرارها بر اساس محصول و مشتری

region	amount	sale_date	dupe_count	customer_id	product_id
شمال	1500	2024-01-01	2	C001	101
شمال	1500	2024-01-01	2	C001	101

```
# شناسایی تکرارها بر اساس تاریخ و منطقه
date_region_dupes <- sales_data %>%
  get_dupes(sale_date, region)

kable(date_region_dupes,caption = "شناسایی تکرارها بر اساس تاریخ و منطقه")
```

Table 2.2: شناسایی تکرارها بر اساس محصول و مشتری

region	amount	sale_date	dupe_count	customer_id	product_id
شمال	1500	2024-01-01	2	C001	101
شمال	1500	2024-01-01	2	C001	101

```
# شناسایی تکرارها بر اساس تاریخ و منطقه
date_region_dupes <- sales_data %>%
  get_dupes(sale_date, region)

kable(date_region_dupes,caption = "شناسایی تکرارها بر اساس تاریخ و منطقه")
```

Table 2.2: شناسایی تکرارها بر اساس تاریخ و منطقه

amount	customer_id	product_id	dupe_count	region	sale_date
1500	C001	101	3	شمال	2024-01-01
1200	C001	103	3	شمال	2024-01-01
1500	C001	101	3	شمال	2024-01-01

شناسایی تکرارها بر اساس محصول و مشتری:

تابع `get_dupes(sales_data, product_id, customer_id)` سطرهای تکراری با شناسه محصول 101 و مشتری C001 را شناسایی کرده و با مقدار dupe_count (تعداد2) نمایش داد است، که نشان‌دهنده دو فروش مشابه برای این ترکیب است.

شناسایی تکرارها بر اساس تاریخ و منطقه:

تابع `get_dupes(sales_data, sale_date, region)` سطرهای تکراری با تاریخ 2024-01-01 و منطقه شمال را شناسایی کرده و با مقدار dupe_count (تعداد 3) نمایش داده است، که نشان‌دهنده سه فروش در این ترکیب است.

2.6.3 مثال 3: داده های مالی

در این مثال مجموعه داده‌ای از اطلاعات مالی شامل :

-شناسه تراکنش(transaction_id)

-شماره حساب(account_number)

-نوع تراکنش (transaction_type)

-مبلغ(amount)

-تاریخ تراکنش(transaction_date)

-کد شعبه(branch_code)

که برخی مقادیر مثل شماره حساب و نوع تراکنش یا تاریخ و شعبه در چند سطر تکرار شده‌اند.

```r
library(dplyr)
library(janitor)

# ایجاد داده مالی پیچیده با نامهای ایرانی
financial_data <- data.frame(
  transaction_id = 1:15,
  account_number = c("محمد احمدی" , "علی رضایی" , "فاطمه کریمی" , "سارا محمدی" , "محمد احمدی","یمی",
 "محمد احمدی" , "سارا محمدی" , "فاطمه کریمی" , "علی رضایی" , "حسن جعفری"),


 transaction_type = c("برداشت" , "انتقال" , "واریز" , "برداشت" , "واریز",
                      "انتقال" , "برداشت" , "واریز" , "انتقال" , "واریز",
                      "برداشت" , "واریز" , "برداشت" , "انتقال" , "واریز"),

  amount = c(5000, 200, 5000, 1500000, 3000, 5000, 2500, 4000, 1000, 2000,
             3500, 1800, 2000, 3200, 1200),

  transaction_date = c("1403-10-25", "1403-10-25", "1403-10-25", "1403-10-26",
                       "1403-10-26", "1403-10-25", "1403-10-27", "1403-10-27",
                       "1403-10-26", "1403-10-28", "1403-10-28", "1403-10-29",
                       "1403-10-25", "1403-10-29", "1403-10-30"),

  branch_code = c("شعبه میدان ولیعصر" , "شعبه نیاوران" , "شعبه میدان ولیعصر" , "شعبه پاسداران" , "
"شعبه میدان ولیعصر" , "شعبه نیاوران" , "شعبه پاسداران" , "شعبه اقدسیه" , "شعبه جردن" , "شعبه پاسداران" , "
)

kable(financial_data,caption = " داده های مالی ")
```

Table 2.3: داده های مالی

branch_code	transaction_date	amount	transaction_type	account_number	transaction_id
شعبه میدان ولیعصر	1403-10-25	5000	واریز	محمد احمدی	1
شعبه پاسداران	1403-10-25	200	برداشت	فاطمه کریمی	2
شعبه میدان ولیعصر	1403-10-25	5000	واریز	محمد احمدی	3
شعبه اقدسیه	1403-10-26	1500000	انتقال	علی رضایی	4
شعبه پاسداران	1403-10-26	3000	برداشت	فاطمه کریمی	5
شعبه میدان ولیعصر	1403-10-25	5000	واریز	محمد احمدی	6
شعبه نیاوران	1403-10-27	2500	انتقال	سارا محمدی	7
شعبه میدان ولیعصر	1403-10-27	4000	واریز	فاطمه کریمی	8
شعبه اقدسیه	1403-10-26	1000	برداشت	علی رضایی	9
شعبه پاسداران	1403-10-28	2000	انتقال	محمد احمدی	10
شعبه جردن	1403-10-28	3500	واریز	حسن جعفری	11
شعبه اقدسیه	1403-10-29	1800	انتقال	علی رضایی	12
شعبه پاسداران	1403-10-25	2000	برداشت	فاطمه کریمی	13
شعبه نیاوران	1403-10-29	3200	واریز	سارا محمدی	14
شعبه میدان ولیعصر	1403-10-30	1200	برداشت	محمد احمدی	15

شناسایی تراکنش‌های مشکوک:

تابع `get_dupes(financial_data, account_number, transaction_type, amount)` تراکنش‌های تکراری با شماره حساب «محمد احمدی»، نوع «واریز»، و مبلغ 5000 را شناسایی کرده و با مقدار dupe_count (تعداد 3) نمایش داده است، که نشان‌دهنده سه تراکنش مشابه است.

```
# تحلیل پیشرفته تکرارها
library(dplyr)

# 1. شناسایی تراکنش‌های مشکوک (حساب، نوع و مبلغ یکسان)
suspicious_transactions <- financial_data %>%
  get_dupes(account_number, transaction_type, amount) %>%
  filter(dupe_count >= 2)

kable(suspicious_transactions,caption = "تراکنش‌های مشکوک")
```

Table 2.4: تراکنش‌های مشکوک

e	transaction_date	transaction_id	dupe_count	amount	transaction_type	account_number
	1403-10-25	1	3	5000	واریز	محمد احمدی
	1403-10-25	3	3	5000	واریز	محمد احمدی
	1403-10-25	6	3	5000	واریز	محمد احمدی

```
# 3. شناسایی حساب‌های پرتراکنش در یک روز
high_activity_accounts <- financial_data %>%
  get_dupes(account_number, transaction_date) %>%
  filter(dupe_count > 2) %>%
  group_by(account_number, transaction_date) %>%
  summarise(
    total_transactions = n(),
    different_branches = n_distinct(branch_code),
    total_volume = sum(amount),
    transaction_types = paste(unique(transaction_type), collapse = ", "),
    .groups = "drop"
  ) %>%
  arrange(desc(total_transactions))

kable(high_activity_accounts,caption = "حساب‌های پرتراکنش در یک روز")
```

Table 2.5: حساب‌های پرتراکنش در یک روز

ypes	total_volume	different_branches	total_transactions	transaction_date	account_number
	15000	1	3	1403-10-25	محمد احمدی

شناسایی حساب‌های پرتراکنش در یک روز:

تابع `get_dupes(financial_data, account_number, transaction_date)` حساب «محمد احمدی» را در تاریخ 25-10-1403 با dupe_count (تعداد 3) شناسایی کرده است، که نشان‌دهنده فعالیت بالا و آمادگی داده برای بررسی تراکنش‌های متمرکز است.

کاربردهای متنوع و سناریوهای واقعی

در فصل نمونه، تابع get_dupes() در سناریوهای نزدیک به دنیای واقعی استفاده شده است (مانند سرشماری، انبارداری و داده‌های پزشکی). این بخش نشان می‌دهد سامانه فقط آموزش دستور نیست، بلکه **اتصال به مسئله واقعی** را هم پوشش می‌دهد.

به عنوان نمونه :

2.7 کاربردهای متنوع

2.7.1 سرشماری عمومی نفوس و مسکن

کاربرد: این تابع برای شناسایی افراد تکراری در داده‌های سرشماری به کار می‌رود تا از خطاهای ثبت جلوگیری کند و دقت نتایج را افزایش دهد.

```
library(janitor)
# داده‌های سرشماری (نمونه)
census_data <- data.frame(
  full_name = c("علی محمدی" ,"فاطمه کریمی" ,"احمد رضایی" ,"احمد رضایی"),
  national_id = c("0123456789", "0123456788", "9876543210", "5555444433"),
  birth_year = c(1370, 1350, 1365, 1360),
  father_name = c("رضا" ,"حسن" ,"محمد" ,"محمد"),
  address = c("شیراز" ,"اصفهان" ,"تهران، منطقه 12" ,"تهران، منطقه 5"),
  household_head = c("خود" ,"دختر" ,"پسر" ,"خود"),
  province_code = c("08", "08", "03", "07")
)

# تشخیص افراد تکراری در سرشماری
duplicate_citizens <- census_data %>%
  get_dupes(full_name, father_name) %>%
  mutate(
    issue_type = case_when(
      national_id != lead(national_id) | national_id != lag(national_id) ~ "🔴 کد ملی متفاوت",
      address != lead(address) | address != lag(address) ~ "🟡 آدرس متفاوت",
      birth_year != lead(birth_year) | birth_year != lag(birth_year) ~ "🟠 سال تولد متفاوت",
      TRUE ~ "🟢 احتمالاً عادی"
    )
  )
kable(duplicate_citizens,caption = "افراد تکراری در سرشماری")
```

Table 2.8: افراد تکراری در سرشماری

household_head	address	birth_year	national_id	dupe_count	father_name	full_name
خود	تهران، منطقه 5	1370	0123456789	2	محمد	احمد رضایی
پسر	تهران، منطقه 12	1350	0123456788	2	محمد	احمد رضایی

تکرار نام «احمد رضایی» با تفاوت در کد ملی و سال تولد شناسایی شده که نشان‌دهنده احتمال خطای ثبت است.

نکات مهم و بهینه‌سازی‌ها

در فصل نمونه، بخشی برای توصیه‌های اجرایی و بهینه‌سازی در نظر گرفته شده است؛ از جمله انتخاب ستون‌های ضروری برای کاهش مصرف حافظه و انجام پردازش به‌صورت قطعه‌ای برای داده‌های بزرگ. وجود این بخش باعث می‌شود آموزش از سطح «اجرا کردن تابع» به سطح «استفاده حرفه‌ای» ارتقا پیدا کند.

2.8 نکات مهم و توصیه‌ها

2.8.1 مدیریت حافظه

برای داده‌های بزرگ، ستون‌های ضروری را انتخاب کنید تا حافظه کمتری مصرف شود و فرآیند سریع‌تر اجرا گردد. این کار از خطاهای حافظه (مانند "out of memory") جلوگیری می‌کند.

```
# فرض کنید large_data یک data frame بزرگ است
large_data %>%
  select(key_columns) %>%  # انتخاب فقط ستون‌های ضروری (یا نام واقعی جایگزین key_columns)
  get_dupes() %>%
  collect()  # جمع‌آوری نتایج (مفید برای محیط‌های توزیع‌شده)
```

2.8.2 پردازش متوالی

داده‌های خیلی بزرگ را به قطعات کوچک‌تر تقسیم کنید تا پردازش گام‌به‌گام انجام شود و سیستم بیش از حد بارگذاری نشود.

```
# تعریف تابع برای پردازش قطعات
process_chunks <- function(data, chunk_size = 10000) {
  split(data, ceiling(seq_len(nrow(data)) / chunk_size)) %>%
    lapply(get_dupes)  # اعمال get_dupes() روی هر قطعه
}

# استفاده: process_chunks(large_data)  # large_data را با داده واقعی جایگزین کنید
```

2.8.3 ترکیب با سایر توابع

get_dupes() را با توابع group_by() و summarise() از dplyr ترکیب کنید تا تحلیل‌های پیشرفته‌تری مانند خلاصه‌سازی تعداد تکرارها انجام دهید.

```
# فرض کنید data یک data frame است
data %>%
  get_dupes(column1, column2) %>%  # شناسایی تکرارها
  group_by(column1) %>%  # گروه‌بندی
  summarise(total_dupes = n(), max_dupe_count = max(dupe_count))  # خلاصه‌سازی
```

2.8.4 مقایسه با روش‌های دیگر

get_dupes() از روش‌های سنتی مانند duplicated() برای داده‌های بزرگ بهتر عمل می‌کند، زیرا خروجی کامل‌تری ارائه می‌دهد و آسان‌تر با pipelineها ترکیب می‌شود.

```
# روش janitor (پیشنهادی)
data %>% get_dupes(column)  # خروجی کامل با dupe_count

# روش سنتی (غیربهینه)
data[duplicated(data$column) | duplicated(data$column, fromLast = TRUE), ]  # ای تکراری بدون شمارش
```

خطاهای رایج و مدیریت آن‌ها

در فصل نمونه، خطاهای رایج مانند اثر مقادیر گمشده (NA) یا تفاوت‌های متنی/کدگذاری در داده‌های فارسی بررسی شده است. این بخش نقش مهمی در کاهش خطاهای عملی کاربران و واقعی‌تر شدن آموزش دارد.

2.9.2 مدیریت خطاهای متن فارسی و کدگذاری

وقتی ستون‌های متنی حاوی فاصله‌های اضافی، تغییرات نوشتار یا مسائل کدگذاری باشند، get_dupes() ممکن است تکرارهای واقعی را شناسایی نکند، زیرا تفاوت‌های جزئی را به عنوان مقادیر متفاوت تلقی می‌کند. این می‌تواند منجر به خروجی خالی یا ناقص شود.

```
# بارگذاری بسته‌های لازم
if (!require(janitor)) { install.packages("janitor"); library(janitor) }
if (!require(dplyr)) { install.packages("dplyr"); library(dplyr) }

# ایجاد داده‌های نمونه با متن فارسی ناپاک
persian_data <- data.frame(
  name = c("احمد "," محمد"," محمد "," محمد"),  # فاصله‌های مختلف
  city = c("شیراز"," تهران"," تهران"," تهران")
)

# بررسی تکرارها - ممکن است هیچ تکراری شناسایی نشود
result <- persian_data %>% get_dupes(name)

# نمایش تعداد ردیف‌های نتیجه (ممکن است صفر باشد)
print(nrow(result))
```

```
## [1] 0
```

این نشان می‌دهد که تکرارهای واقعی (مانند "محمد" با فاصله‌های متفاوت) شناسایی نشده‌اند، زیرا تابع تفاوت‌های جزئی را نادیده نمی‌گیرد.

راه‌حل: پاک‌سازی متن فارسی قبل از بررسی

برای حل این مشکل، متن را پاک‌سازی کنید: فاصله‌های اضافی را حذف نمایید و optionally به حروف کوچک تبدیل کنید (اگر داده‌ها ترکیبی از فارسی و انگلیسی باشند). این کار با استفاده از توابع stringr انجام می‌شود و تکرارهای واقعی را آشکار می‌سازد. این روش ساده و قابل ادغام در pipelineهای dplyr است.

```
# بارگذاری بسته اضافی برای پردازش رشته‌ها
if (!require(stringr)) { install.packages("stringr"); library(stringr) }

# پاک‌سازی متن فارسی و بررسی تکرارها
clean_persian <- persian_data %>%
  mutate(
    name_clean = str_trim(name),  # حذف فاصله ابتدا و انتها
    name_clean = str_squish(name_clean),  # حذف فاصله‌های اضافی میانی
    name_clean = str_to_lower(name_clean)  # (اختیاری: تبدیل به حروف کوچک (برای داده‌های ترکیبی
  ) %>%
  get_dupes(name_clean)

kable(clean_persian,caption = "بعد از پاک‌سازی متن")
```

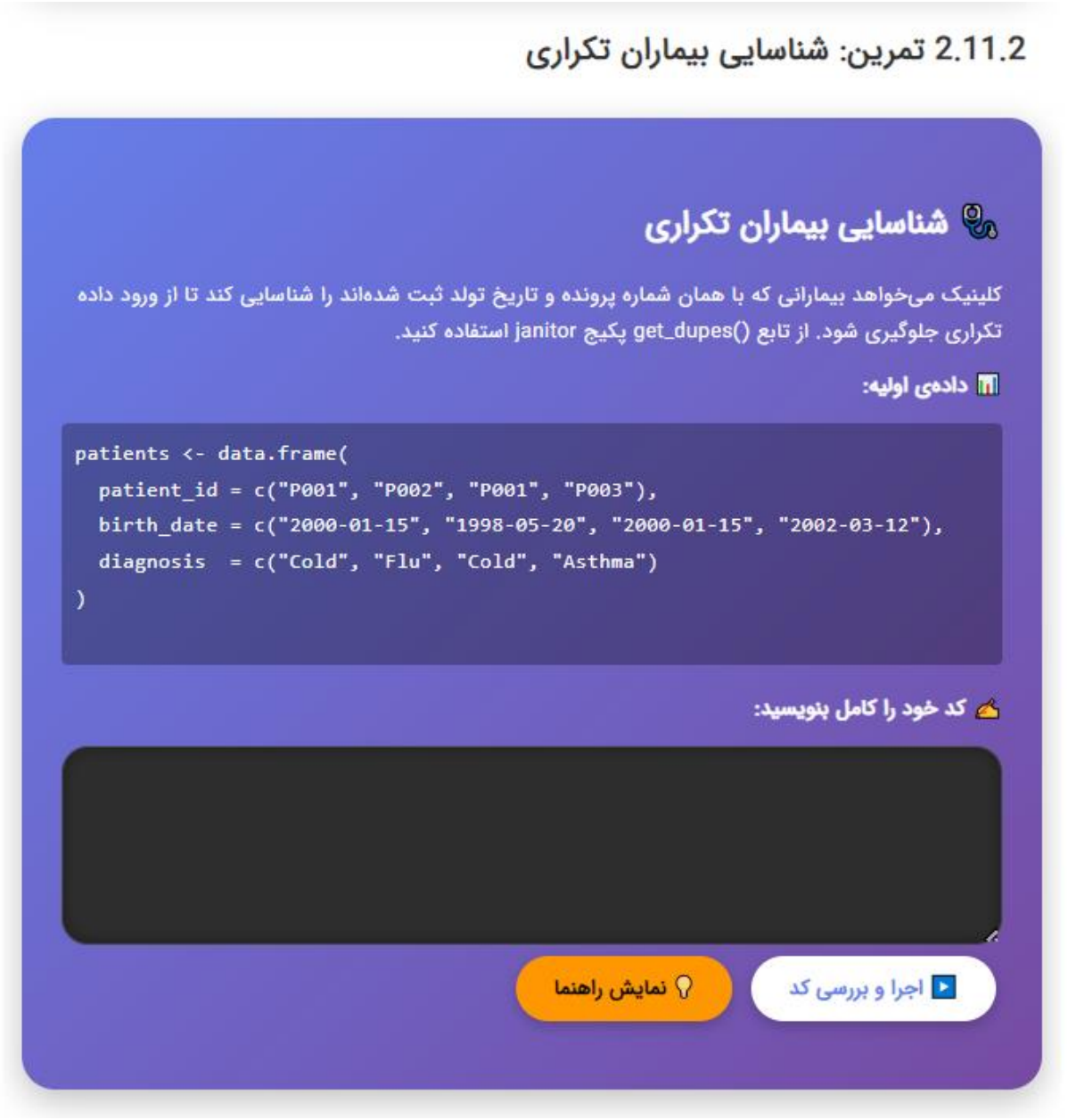
Table 2.10: بعد از پاک‌سازی متن

city	name	dupe_count	name_clean
تهران	محمد	3	محمد
تهران	محمد	3	محمد
تهران	محمد	3	محمد

این خروجی نشان می‌دهد که پس از پاک‌سازی، سه مورد "محمد" به عنوان تکراری شناسایی شده است .

تمرین تعاملی و سنجش‌پذیری یادگیری

بخش تمرینی فصل به‌صورت تعاملی طراحی شده است؛ به‌گونه‌ای که کاربر کد را در همان صفحه وارد می‌کند، دکمه اجرا را می‌زند، خروجی را می‌بیند و در صورت اشتباه، بازخورد دریافت می‌کند. علاوه بر این، راهنمای مرحله‌ای (Hint) برای هدایت کاربر به سمت پاسخ درست فراهم شده است.



شکل ۳-۴. معرفی تمرین تعاملی مرتبط با تابع get_dupes

2.11.2 تمرین: شناسایی بیماران تکراری

شناسایی بیماران تکراری

کلینیک می‌خواهد بیمارانی که با همان شماره پرونده و تاریخ تولد ثبت شده‌اند را شناسایی کند تا از ورود داده تکراری جلوگیری شود. از تابع get_dupes() پکیج janitor استفاده کنید.

داده‌ی اولیه:

```
patients <- data.frame(
  patient_id = c("P001", "P002", "P001", "P003"),
  birth_date = c("2000-01-15", "1998-05-20", "2000-01-15", "2002-03-12"),
  diagnosis  = c("Cold", "Flu", "Cold", "Asthma")
)
```

کد خود را کامل بنویسید:

```
03-12"),
  diagnosis  = c("Cold", "Flu", "Cold", "Asthma")
)
dupes_patients <- patients %>% get_dupes(patient_id, birth_date)
dupes_patients
```

اجرا و بررسی کد | نمایش راهنما

داده اولیه:

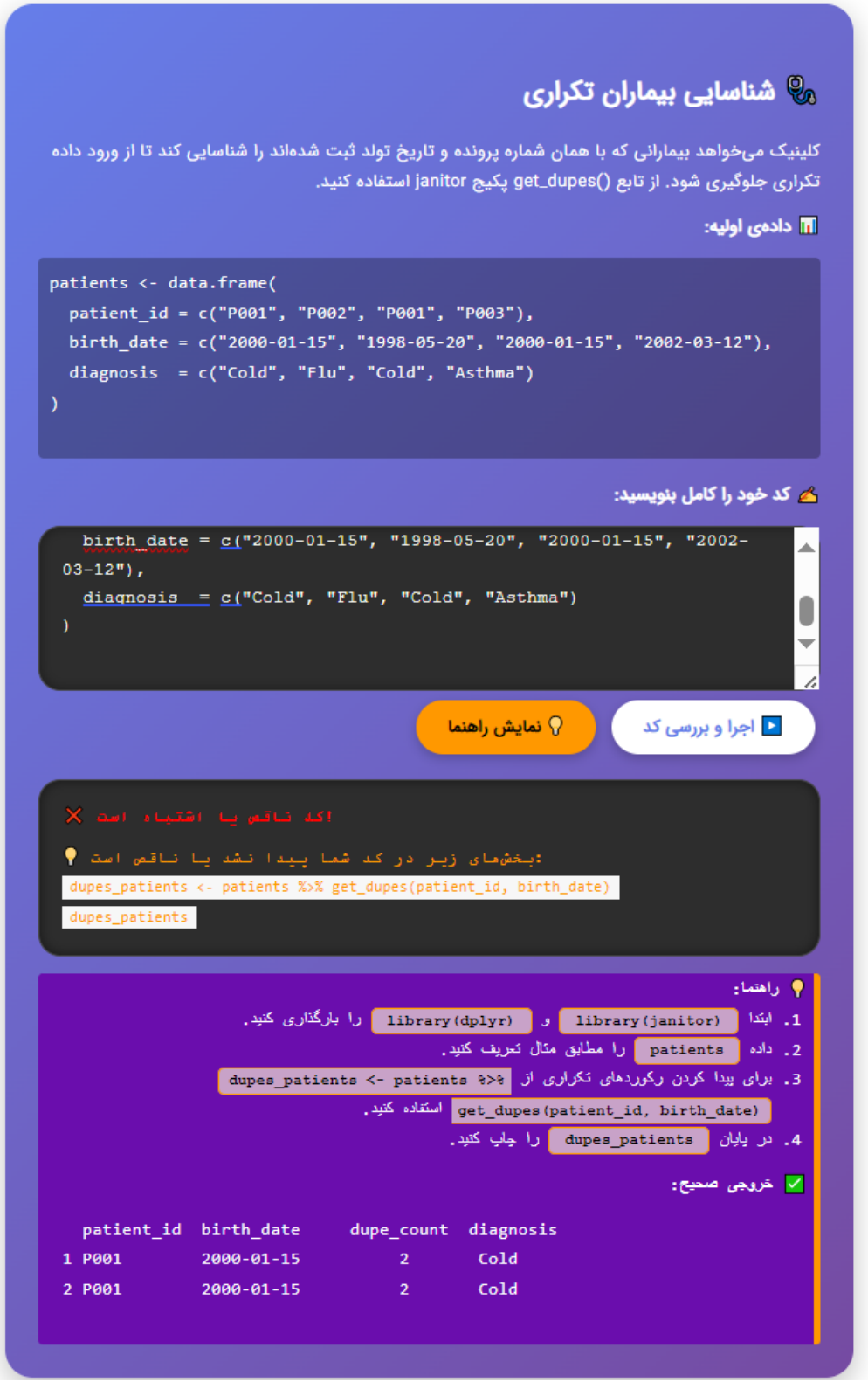
	patient_id	birth_date	diagnosis
1	P001	2000-01-15	Cold
2	P002	1998-05-20	Flu
3	P001	2000-01-15	Cold
4	P003	2002-03-12	Asthma

بیماران تکراری:

	patient_id	birth_date	dupe_count	diagnosis
1	P001	2000-01-15	2	Cold
2	P001	2000-01-15	2	Cold

شکل ۴-۴. نمایش خروجی کاربر پس از اجرای صحیح کد

2.11.2 تمرین: شناسایی بیماران تکراری



شکل ۴-۱۷. نمونه خطای کاربر و پیام بازخورد سیستم

راهنمای (Hint) ارائه‌شده برای اصلاح خطا

۳-۴ تحلیل فنی کوتاه از چرخه تعامل کاربر در فصل نمونه

چرخه تعامل در تمرین‌های سامانه به‌صورت «نوشتن کد ← اجرا ← مشاهده خروجی ← اصلاح» انجام می‌شود. در این چرخه، موتور WebR اجرای کد را در سمت کاربر انجام می‌دهد و خروجی بدون ارسال به سرور در همان صفحه نمایش داده می‌شود. همچنین وجود بازخورد (خطا/موفقیت و راهنما) باعث می‌شود تمرین‌ها صرفاً نمایشی نباشند و نقش آموزشی فعال داشته باشند.

این مدل تعامل، مشابه محیط REPL در R عمل می‌کند، با این تفاوت که در بستر WebAssembly و بدون وابستگی به سرور اجرا می‌شود.

فصل پنجم

نتیجه‌گیری نهایی و دستاورد پژوهش

این پژوهش با هدف پاسخ به یک مسئله مشخص شکل گرفت: آیا می‌توان آموزش عملی پاکسازی داده در زبان R را به‌گونه‌ای طراحی کرد که اجرای واقعی کد، ساختار آموزشی منسجم و استقلال از زیرساخت سرور به‌صورت هم‌زمان محقق شوند؟ نتیجه حاصل نشان می‌دهد که این هدف نه‌تنها امکان‌پذیر است، بلکه می‌تواند به چارچوبی پایدار و توسعه‌پذیر منجر شود.

سامانه طراحی‌شده در این پژوهش نشان داد که اجرای واقعی زبان R در مرورگر، بدون اتکا به محیط‌های نصب‌شده محلی یا سرورهای پردازشی مرکزی، قابلیت تحقق دارد. این دستاورد صرفاً یک پیاده‌سازی فنی نیست؛ بلکه تغییری در منطق ارائه آموزش عملی برنامه‌نویسی محسوب می‌شود. در مدل‌های متعارف، یادگیری عملی وابسته به پیکربندی نرم‌افزار، مدیریت وابستگی‌ها یا دسترسی به زیرساخت اجرایی است. چارچوب حاضر نشان می‌دهد که می‌توان این وابستگی‌ها را حذف کرد و تجربه‌ای تعاملی و واقعی را در قالب یک سامانه وب ایستا فراهم ساخت.

از منظر آموزشی، پژوهش حاضر نشان داد که انسجام ساختاری ماژول‌ها، پیوند نظام‌مند میان مبانی نظری و تمرین‌های تعاملی، و ارائه بازخورد فوری به کاربر می‌تواند فرایند یادگیری را از حالت انتقال محتوا به تجربه فعال تبدیل کند. تمرکز هدفمند بر پاکسازی داده، که بخش عمده‌ای از زمان پروژه‌های تحلیلی را به خود اختصاص می‌دهد، جایگاه راهبردی این مهارت را در آموزش تحلیل داده برجسته می‌سازد. بدین ترتیب، خروجی پژوهش نه مجموعه‌ای از صفحات آموزشی، بلکه یک چارچوب ساختاریافته برای پرورش مهارت عملی مدیریت کیفیت داده است.

از دیدگاه معماری نرم‌افزار، این پژوهش نشان داد که تفکیک لایه اجرا از لایه محتوا، مدیریت وضعیت اجرای موتور و کنترل خطا در دو سطح می‌تواند به سامانه‌ای پایدار، مقیاس‌پذیر و قابل نگهداری منجر شود. حذف وابستگی به سرور، توزیع بار محاسباتی در سمت کاربر و امکان استقرار به‌صورت وب‌سایت استاتیک، سامانه را از منظر عملیاتی نیز به گزینه‌ای کارآمد برای محیط‌های آموزشی و سازمانی تبدیل می‌کند. این ترکیب از طراحی مهندسی و منطق آموزشی، چارچوبی منسجم ایجاد کرده است که قابلیت تعمیم به حوزه‌های دیگر آموزش برنامه‌نویسی را دارد.

در عین حال، محدودیت‌هایی همچون وابستگی به توان پردازشی دستگاه کاربر، محدودیت حافظه مرورگر و نیاز به ارزیابی تجربی گسترده‌تر وجود دارد که می‌تواند در ادامه مسیر توسعه مورد بررسی قرار گیرد. با این حال، این محدودیت‌ها ماهیت چارچوب را زیر سؤال نمی‌برند، بلکه مسیرهای پژوهشی آینده را مشخص می‌کنند.

در جمع‌بندی نهایی می‌توان گفت این پژوهش نشان داد که آموزش عملی تحلیل داده می‌تواند از وابستگی‌های زیرساختی رها شود و به محیطی یکپارچه، تعاملی و مستقل تبدیل گردد. چارچوب ارائه‌شده نمونه‌ای عملی از همگرایی فناوری‌های نوین وب و طراحی آموزشی نظام‌مند است و می‌تواند مبنایی برای توسعه نسل جدید سامانه‌های آموزش برنامه‌نویسی در بستر مرورگر باشد. این دستاورد، گامی در جهت کاهش فاصله میان یادگیری نظری و تجربه عملی در آموزش داده‌محور به شمار می‌آید.