باسمه تعالی

پروژه شطرنج، درس اصول طراحی نرم افزار استاد کارگر
کاری از پویان سلمانی و مریم منوریان

در این پروژه بازی شطرنج در بخش فرانت اند پیاده سازی شده است و تمامی قوانین بازی به صورت جاوااسکریپت پیاده سازی شده اند.

تکونولوژی های مورد استفاده: React js, Sass

ساختار کلی پروژه:

پروژه در کامپوننت main شروع میشود، جایی که کامپوننت های app, notif, modal جایگذاری شده اند. علت این کار رعایت سمانتیک صفحه به علت جدا بودن مفاهیم این سه بخش است.

کامپوننت modal: این کامپوننت شامل تمامی اعلانات و صفحاتی است که به صورت مدال نمایش داده میشوند. در حالت عادی این کامپوننت خالی است و محتوایی ندارد و محتوای با با فانکشن modalHandler که همراه کامپوننت اکسپورت میشود تغییر میکند.

کامپوننت notif: این کامپوننت مسئول نمایش دادن خطا هایی است که هنگام بازی رخ میدهد. (مانند زمانی که کاربر حرکت غیر قانونی شطرنج انجام میدهد)

کامپوننت app: این قسمت بخش اصلی پروژه است، که به ترتیب چند کار انجام میدهد. ابتدا با استفاده از modalHandler یک مدال اولیه جهت ورود اطلاعات کاربر مانند نام و ریتینگ کسانی که بازی میکنند است، که بعد از اینکه اطلاعات از این مدال دریافت شد، استیتی که کامپوننت board را نگه داری میکند اپدیت میشود و اطلاعات آنها در صفحه نمایش داده میشود. کامپوننت board جایی است که بازی اصلی اتفاق می افتد.

کامپوننت board: این کامپوننت شامل تمام چیز هایی است که یک صفحه شطرنج نیاز دارد در خود داشته باشد. چیزی که این کامپوننت رندر میکند هشت ردیف است که هر ردیف خود هشت ستون دارد، که در نهایت شکل صفحه شطرنج را تشکیل میدهند.
یک سری متغییر که متغلق به صفحه هستند، مانند نوبت بازی (turn) ساعت بازی (clock) نیز در اینجا تعریف میشوند و به آبجکت board بایند میشوند. در نهایت ابجکت positions و pieces ساخته میشوند که یکی به ردیف ها پاس داده میشود تا خانه هایی که در صفحه تعریف شده اند در ان نگه داری شود و دیگری به مهره ها پاس داده میشوند تا اطلاعات خود را در ان نگه داری کنند.
در نهایت این کامپوننت فانکشن های setUpBoard و setBoardClicks را صدا میزند که اولی مهره ها را در حالت اولیه بازی در صفحه میچیند و دومی اوینت کلیک روی مهره ها را جایگذاری میکند. فانکشن دوم همچنین مسئولیت هندل درگ اند دراپ مهره ها را دارد.

فانکشن setUpBoard: این فانکشن که به positions نیز دسترسی دارد، هر مهره را در جایگاه درست خودش در آبجکت positions بایند میکند و میسازد. مهره ها بر اساس کلاس هایی که برای آنها تعریف شده اند ساخته میشوند و عملیات های حرکت هر مهره، مرده یا زنده بودن آن و رعایت استثناعات شطرنج به عهده هر مهره است. این کلاس از فایل pieces.jsx ایمپورت میشود که در این فایل تمام کلاس های مربوط به مهره ها تعریف شده است.

فایل pieces.jsx: در این فایل یک کلاس اصلی به نام Piece تعریف شده که اساس و خمیر تمامی مهره ها است. کلاس های مربوط به هر مهره (Pawn, Knight, King, Queen, Rook) از این کلاس وراثت میگیرند و تفاوت های مربوط به هر مهره

در کلاس خود به صورت استثنا نوشته میشود. همچنین تفاوت های اصلی مهره ها به عنوان ارگمان به کلاس اصلی Piece پاس داده میشود.

نحوه تعریف هر مهره: هر مهره بر اساس چهار ویژگی شکل میگیرد. موقعیت اولیه آن، سفید بودن یا سیاه بودن، الگوی حرکت و صفحه ای که به آن تعلق دارد. هر کلاس (مانند Pawn) کنستراکتور کلاس اصلی را صدا میزند و این ویژگی ها را اعلام میکند. همچنین تصویر مربوط به هر مهره بر اساس نام کلاس آن در فولدر عکس ها پیدا میشود.
به این صورت کلاس اصلی (Piece) تصویر این مهره را بر اساس نام کانستراکتور و سیاه یا سفید بودن آن پیدا میکند، آن را در موقعیت اولیه که اعلام شده در صفحه ای که اعلام شده جایگذاری میکند، و متد move را برای آن تشکیل میدهد که بر اساس الگوی حرکتی است که هر مهره دارد. با این روش صفحه های کنار هم نیز با یکدیگر تداخل نخواهند داشت.

متد move: این متد مسئول حرکت مهره است. با صدا زدن این متد متد دیگری به نام moveAuthorize نیز صدا زده میشود که مسئول چک کردن این است که آیا این حرکت مجاز است یا نه. در صورت مجاز بودن فانکشن move تصویر مهره را در صفحه جابجا میکند و در آبجکت های pieces و positions موقعیت مهره را اپدیت میکند.

متد moveAuthorize: این متد تمامی قوانین شطرنج را در بر میگیرد. ابتدا چک میکند الگوی حرکت مهره با حرکتی که درخواست آن فرستاده شده تطابق دارد یا نه. سپس چک میشود که در خانه مقصد مهری دیگری هست یا نه. یا اینکه مثلا بین راه مهره دیگری وجود دارد یا نه. سپس بر اساس آن یک true یا false ساده برمیگرداند که نشانه اجازه حرکت مهره است.

الگوی حرکت: الگوی حرکت هر مهره به صورت یک Regex در یک استرینگ نوشته میشود. حرکت مهره ها به صورت استرینگ آنالیز میشوند. به عنوان مثال حرکت دادن یک مهره دو خانه به سمت بالا و یک خانه به سمت چپ در برنامه به عنوان "ulu" شناخته میشود. این الگو ابتدا حرکت مهره را یک خانه در جهت عمودی، سپس یک خانه در جهت افقی، سپس دوباره یک خانه عمودی آنالیز میکند تا زمانی که به خانه مقصد برسدِ و استرینگ تولید شده (که ممکن است به این شکل باشد "ulululuuu") با الگوی حرکت آن مهره تطابق میدهد. اگر این تطابق درست باشد آنگاه آن مهره اجازه حرکت دارد. در غیر این صورت مهره حرکت نمیکند و اررو بر میگرداند.
همچنین رجکس حرکت مهره ها چیزی مانند این است: مثلا برای فیل '+([lr][ud])' به این معنا که عبارات lr یا ud پشت سر هم و به هر تعدادی.

صد البته که فانکشن های بسیار دیگری هم هستند که وظیفه های مختلفی را بر عهده دارند. مانند تولید انیمیشن حرکت، کشته شدن یک مهره، تعویض نوبت بازی و محاسبه زمان هر بازیکن، و المان های دیگری که در بحث نمیگنجد.

با تشکر از استاد