به نام خدا

تمرین سری هشتم

مائده بادان فیروز - ۹۸۲۲۲۰۰۹

بهار ۱۴۰۲

مقدمه:

در این برنامه سعی شده تا با استفاده از socket programming یک برنامه‌ی شامل Server و Client نوشته شود که Steam با functionalityهای خیلی کمتر شبیه‌سازی شده است.

طراحی و پیاده سازی:

طراحی برنامه به گونه‌ای است که تبادل اطلاعات بین کلاینت و سرور با رد و بدل کردن درخواست و پاسخ انجام می شود. اطلاعات در attribute به نام data از جنس Object ذخیره می‌شوند و در attribute به نام title نوع درخواست‌ها و پاسخ‌ها تعیین می‌شود که پاسخ‌ها دقیقا به نظیر درخواست‌ها فرستاده می‌شوند. دلیل اینکه data از نوع Object تعریف شده این است که هر شیء دیگری که از آن ارث‌بری کرده بتواند جای آن بنشیند. چون شیءهای ما برای انتقال LoginDto, RegisterDto و ... هستند که بسته به نیاز برای انتقال اطلاعات تعریف شده‌اند.

در سمت سرور برنامه در کلاس main سرور آغاز می‌شود و همواره منتظر رسیدن درخواستی برای وصل شدن به سوکت می‌ماند تا کلاینت جدید بخواهد وصل شود و وقتی وصل شد برای آن یک ترد جدید می‌سازد و کلاینت روی همان ترد کار می‌کند. به این صورت برنامه به صورت multiThread کار می‌کند و چندین کلاینت می‌توانند همرمان به سرور وصل شوند و هر کدام عملیات مورد نظر خودشان را انجام بدهند.

در کلاس Manager اصل کار مدیریت می‌شود. بسته به نوع درخواستی که آمده (با توجه به title هر request) ابتدا نوع دیتایی که همراهش هست را می‌داند و این تبدیل را انجام می‌دهد و بعد برای دادن پاسخ مناسب این اطلاعات استخراج شده را به تابع مورد نظر می‌دهد تا پاسخ را بگیرد. هر کدام از این توابع به لایه‌های عمیق‌تری از برنامه (ابتدا به Serviceها و سپس به Repositoryها) می‌روند تا جایی که از database اطلاعات را استخراج کنند یا در آن ثبت کنند و سپس با گزارش موفقیت یا شکست به همراه اطلاعات ضمیمه شده (در صورت وجود) به Manager برگردند تا به JSON تبدیل شوند و از طریق سوکت به کلاینت فرستاده شوند. توجه داریم که در ابتدای کار هم درخواست به صورت JSON فرستاده شده بود و خود درخواست هم ابتدا از JSON به Request تبدیل شده تا بتواند با آن کار شود.

تبدیل به JSON و برعکس خود با چالش‌هایی مواجه بود که با سرچ حل شد. مثلا برای اینکه بتوانیم اطلاعات تاریخ تولد را که از نوع LocalDate بود درتبدیل درخواست به JSON بفرستیم و بعد در مقصد بخوانیم نیاز بود که mapper ما mudole به نام JavaTimeModule() را بشناسد که هم در سمت سرور (خط ۴۵ کلاس Manager) و هم در سمت کلاینت (خط ۴۴ کلاس ClientMain) کد مربوط به آن اضافه شده است.

در سمت کلاینت با طراحی منوهای مناسب به کاربر این قابلیت داده شده که لیست تمام بازی‌ها و جزئیات یک بازی انتخابی را بتواند ببیند و نیازی به ورود به حساب کاربری خودش را ندارد اما وقتی بخواهد بازی‌ای را دانلود کند باید حتما وارد حساب کاربری خودش شود که طبیعتا اگر اولین بار باشد ابتدا باید یک حساب بسازد.

برای هندل کردن اینکه تاریخ تولد حتما درست وارد شود از رجکس استفاده کرده‌ایم و برای آن یک Exception خاص ساخته‌ایم که اگر تاریخ به صورت ناصحیح وارد شود این Exception رخ می‌دهد و دوباره از کاربر خواسته می‌شود که به صورت صحیح تاریخ را وارد کند.

سنجش و ارزیابی:

تمام functionalityهای ساخته شده چک شده‌اند و بدون ارور کار می‌کنند. بخش‌های زیادی حین کد زدن دیباگ شده که توضیح مفصل هر کدام در این مقال نمی‌گنجد!

نتیجه‌گیری:

در ابعاد واقعی، برنامه‌ها ورژن گشترده‌تری از اینچنین برنامه‌ای هستند که functionalityهای بیشتری دارند، و البته حتما database های بزرگتر و پیچیده‌تر. به نسبت بزرگتر شدن پروژه و اضافه کردن هر قابلیت به آن زمان بسیار بیشتر برای پیاده سازی آن مورد نیاز است.