به نام خدا

پروژه پایانی درس برنامه نویسی پیشرفته

دکتر مجتبی وحیدی اصل

عنوان پروژه : سامانه یکپارچه خدمات بانکی و مدیریت هزینه‌های مشترک

مقدمه و هدف پروژه

هدف این پروژه، پیاده‌سازی یک سامانه کلاینت–سرور برای شبیه‌سازی عملیات بانکی و مدیریت هزینه‌های اشتراکی است. این پروژه به صورت گروه‌های دو نفره تعریف می‌شود و تمرکز اصلی آن بر تقویت مهارت‌های برنامه‌نویسی در سمت سرور با استفاده از زبان جاوا و مفاهیم بنیادین آن است.

سامانه از دو ماژول اصلی و یکپارچه تشکیل شده است:

ماژول خدمات بانکی :کاربران پس از ثبت‌نام، می‌توانند حساب‌های بانکی از انواع مختلف (جاری و پس‌انداز) ایجاد کرده و عملیاتی نظیر انتقال وجه و مشاهده تاریخچه تراکنش‌ها را انجام دهند.

ماژول مدیریت هزینه‌های مشترک :کاربران می‌توانند گروه‌هایی برای مدیریت هزینه‌های اشتراکی ایجاد کنند. در هر گروه، اعضا هزینه‌هایی که پرداخت کرده‌اند را ثبت می‌کنند و سامانه به طور خودکار سهم هر فرد را محاسبه می‌کند. قابلیت کلیدی سامانه، امکان تسویه حساب بدهی‌ها از طریق ماژول بانکی داخلی است.

قابلیت‌ها و صفحات پیشنهادی

لیست زیر یک پیشنهاد کلی برای ساختار و قابلیت‌های صفحات اپلیکیشن است. گروه‌ها می‌توانند این ساختار را به شرط حفظ تمام قابلیت‌های اصلی، مطابق با سلیقه و خلاقیت خود تغییر دهند.

۱ .صفحه ورود و ثبت‌نام(Authentication)

شرح :صفحات استاندارد برای ورود و ثبت‌نام کاربر.

۲ .داشبورد اصلی(Home Page)

شرح :این صفحه نقطه ورود اصلی کاربر به سامانه است.

قابلیت‌ها:

- نمایش خلاصه وضعیت کاربر (مانند نام و موجودی حساب‌ها).
- دسترسی‌های سریع به بخش‌های اصلی سامانه (مانند پروفایل، مدیریت حساب‌ها و گروه‌های مشترک).
- میان‌برهایی برای خدمات پرکاربرد حساب (مانند کارت به کارت و انتقال پول).
- امتیازی :پیاده‌سازی خدمات اضافی مانند پرداخت قبوض، خرید شارژ یا فرم شبیه‌سازی شده افتتاح حساب جدید.

۳ .صفحه مدیریت حساب‌ها(Accounts Management)

شرح :نمایش متمرکز تمام حساب‌های بانکی کاربر.

قابلیت‌ها:

- نمایش لیستی از حساب‌های بانکی کاربر به همراه موجودی هرکدام.
- گزینه‌ای برای ایجاد یک حساب بانکی جدید.
- قابلیت جستجو و فیلتر کردن حساب‌ها (مثلاً بر اساس نوع حساب: جاری/پس‌انداز).

۴ .صفحه جزئیات حساب و عملیات بانکی(Account Details & Operations)

شرح :نمایش اطلاعات کامل و جامع یک حساب بانکی مشخص.

قابلیت‌ها:

- **نمایش اطلاعات حساب :شماره حساب، شماره کارت، مانده و غیره .**

راهنمایی : زمان افتتاح حساب، سرور باید خودش یک شماره کارت ۱۶ رقمی و یک شماره حساب برای کاربر تولید کند (مثلاً رندوم یا ترتیبی) و مطمئن شود که تکراری نیست. همچنین برای موجودی اولیه حساب ها میتوانید در زمان افتتاح حساب یک موجودی اولیه به کاربر ها دهید یا یک گزینه واریز وجه داشته باشید که بتواند مبلغ مشخصی را به هر حساب اضافه کند.

- **تاریخچه تراکنش‌ها:**
  - **نمایش لیست کامل تراکنش‌های حساب.**
  - **جستجوی پیشرفته :امکان فیلتر کردن تراکنش‌ها بر اساس نوع (واریز/برداشت) و انتخاب بازه زمانی مشخص.**
  - **دسته‌بندی تراکنش‌ها :امکان اختصاص دادن دسته‌بندی به هر تراکنش (مثلاً دستمزد، اجاره، سود، مخارج زندگی و...).**
- **عملیات حساب :دسترسی به فرم‌هایی برای عملیات اصلی مانند تغییر رمزهای کارت (شبیه‌سازی شده) و انتقال وجه.**

موارد امتیازی :فراهم کردن امکان دانلود صورت‌حساب تراکنش‌ها مثلاً در قالب یک فایل متنی ساده یاPDF

۵ .منطق انتقال وجه

شرح :در پیاده‌سازی منطق انتقال وجه (که در داشبورد و صفحه جزئیات حساب قابل دسترسی است)، روش‌های مختلفی با قوانین متفاوت باید شبیه‌سازی شوند. به عنوان مثال:

1. **کارت به کارت :این روش باید سقف انتقال مشخصی داشته باشد (مثلاً ۱۰ میلیون تومان).**
2. **کارت به کارت (بانک مشترک) :اگر شماره کارت مبدأ و مقصد مربوط به یک بانک (شبیه‌سازی شده) باشند، سقف انتقال باید برداشته شود.**
3. **انتقال با شماره شبا :این روش باید سقف انتقال بسیار بالاتری داشته باشد یا بدون سقف در نظر گرفته شود.**

4. **و موارد دیگر به صلاحدید شما.**

۶ .صفحه مدیریت گروه‌های مشترک(Shared Groups Management)

شرح :مرکز مدیریت تمام گروه‌های هزینه کاربر.

قابلیت‌ها:

- **نمایش لیستی از گروه‌هایی که کاربر در آنها عضو است.**
- **قابلیت ایجاد یک گروه جدید.**
- **امکان افزودن اعضای جدید (از بین کاربران ثبت‌شده در سیستم) به گروه‌های موجود.**

۷ .صفحه جزئیات گروه مشترک(Shared Group Details)

شرح :نمایش اطلاعات کامل و وضعیت یک گروه هزینه مشخص.

قابلیت‌ها:

- **نمایش لیست کامل اعضا و تمام هزینه‌های ثبت‌شده.**
- **فرمی برای افزودن یک هزینه جدید با جزئیات کامل.**
- **دکمه «مشاهده نتیجه نهایی» برای نمایش شفاف بدهی‌ها و طلب‌های هر یک از اعضا.**
- **گزینه ای برا تسویه حساب با افراد گروه.**

۸ .صفحه پروفایل کاربری(User Profile)

شرح :مدیریت اطلاعات شخصی کاربر.

قابلیت‌ها:

- **نمایش اطلاعات کاربر (نام، نام کاربری و...).**
- **امکان ویرایش اطلاعات پایه (مانند تغییر نام).**
- **گزینه‌ای برای خروج از حساب کاربری**(Logout)

فناوری‌های مورد نیاز پروژه :

در این پروژه باید از مفاهیمی که در درس برنامه نویسی پیشرفته (سوکت، نتورک، ترد، فایل و مباحث مرتبط با شی گرایی) که آموخته اید استفاده کنید.

- **بک اند پروژه به زبان جاوا توسعه داده میشود. استفاده از هرگونه فریمورک آماده مانند Spring ممنوع است.**
- **سرور باید به ازای هر کلاینت متصل، یک نخ (Thread) مجزا ایجاد کند یعنی سرور برنامه باید توانایی پاسخگویی به چندین کاربر مختلف در زمان واحد را داشته باشد.**
- **فرانت اند پروژه با استفاده از فریمورک فلاتر و زبان دارت توسعه داده میشود.**
- **دیتاهای مورد نیاز این پروژه در فایل های متنی با فرمت json. ذخیره میشوند و استفاده از دیتابیس های دیگر مانند mysql , nosql و ... ممنوع میباشد. همچنین با بسته شدن برنامه سرور هیچ داده ای نباید پاک شود و با اجرای دوباره سرور داده ها باید دوباره بارگذاری شده و از حالت قبلی ادامه باید.**

امتیازی : در صورت پیاده سازی ذخیره سازی داده ها در پایگاه داده ای غیر از دیتابیس های ذکر شده باید هر دو شیوه موجود باشد (مثلا اگر به جای دیتابیس فایل json برای ذخیره سازی اطلاعات از دیتابیس دیگری استفاده شود باید هم ذخیره سازی به صورت فایل json و هم دیتابیس جدید هر دو موجود باشند . برای اینکار میتوان در برنچ دیگری در گیتهاب نسخه جدید را اضافه کرد که به عنوان مورد امتیازی لحاظ گردد.)

- **پروتکل ارتباطی از نوع سوکت‌های TCP/IP اس و استفاده از HTTP یا API های آماده ممنوع است.**

امتیازی : استفاده از api ، فایربیس و .... مجاز است اما باید هر دو ارتباط موجود باشد. به طور مثال در گیتهاب پروژه ی خود دو برنچ نهایی داشته باشید یکی با api و دیگری که نسخه ی پایه ی آن با سوکت است .

موارد امتیازی دیگر :

می‌توانید از قابلیت‌های زیر ایده گرفته یا موارد خلاقانه خود را پیاده‌سازی کنید:

امتیازات سمت بک‌اند :محاسبه سود بانکی، جستجوی پیشرفته تراکنش‌ها، تقسیم هزینه غیرمساوی.

امتیازات سمت فرانت‌اند :قابلیت تغییر تم (روشن/تاریک)، استفاده از انیمیشن‌های مناسب ، responsive **بودن اپلیکیشن.**

امتیازات یکپارچه‌سازی :تنها در صورتی که پروژه با تمام موارد خواسته شده پیاده‌سازی شود، می‌توانید در یک شاخه (Branch) مجزا در گیت، از فریمورک‌ها، APIها یا دیتابیس‌های دیگر برای توسعه بیشتر استفاده کنید.

قوانین، ابزارهای توسعه و سیاست های پروژه :

- **توجه داشته باشید که اجباری برای پیاده سازی صد درصدی مدل های ذکر شده نیست اما پروژه شما باید دارای امکانات و صفحات خواسته شده باشد و هرگونه خلاقیت اضافه چه در قابلیت های نرم افزار و چه در طراحی رابط کاربری مجاز میباشد و در صورت صلاحدید تحویل گیرنده به عنوان امتیاز مثبت منظور می شود.**
- **همانگونه که میدانید پروژه در قالب گروه های یک الی دو نفره انجام میشود. سعی شود پیش از شروع پیاده سازی بین اعضای گروه تقسیم کار صورت گیرد و مشخص باشد که هر فرد چه بخش هایی را قرار است پیاده سازی کند .**
  - **برای تقسیم بندی و تسک بندی پروژه از ابزارهای زیادی میتوانید استفاده کنید. از گیتهاب برای تسک بندی میتوانید کمک بگیرد. یکی از سایت هایی که در این زمینه کارایی بالایی دارد و استفاده از آن آسان است Trello نام دارد.**
- **تقسیم کار به شما بستگی دارد که به چه شکل صورت میگیرد ولی اکیدا توصیه میشود که هر دو نفر هم در بخش فرانت مشارکت داشته باشند و هم در بخش بک اند. (تقسیم بندی هایی که به این صورت هستند که یکی از اعضای گروه بخش فرانت را به عهده بگیرد و دیگری بخش بک اند را مورد قبول نمیباشند)**
- **توجه شود که در نهایت هر دو نفر باید به کلیات بخش هایی که توسط همگروهیشان زده میشود تسلط کلی داشته باشند .**
- **در هنگام تحویل پروژه در صورتی که موارد خواسته شده پیاده سازی نشده باشند نمره آن بخش از تمامی اعضای گروه کم میشود، اما تسلط هر فرد به کلیات بخش هایی که هم گروهی او پیاده سازی کرده نیز سنجیده میشود و این درصد تسلط برای هر شخص متفاوت است و تاثیر زیادی در نمره نهایی شما دارد.**
- **استفاده از GitHub اجباری است و میزان استفاده از گیت استفاده یا عدم استفاده تعداد کامیت و کامیتهای هر فرد و برنچ ها شامل نمره اختصاصی است و عدم استفاده به منزله دریافت نکردن نمره آن بخش است. پیشنهاد میشود در هر مرحله برای پیشرفت کار تمامی تغییرات خود را با کامیتهای مناسب در ریپوی خود پوش کنید.**

- **اضافه کردن توضیحات و تصاویر مناسب و گزارشات تهیه شده از اپلیکیشن به** repository **پروژه در گیتهاب الزامی است.**
- **تمیزی کد که شامل نام گذاریها و کامنت گذاری ها و .... مورد ارزیابی قرار میگیرد.**
- **بخش قابل توجهی از نمره ی دریافتی از پروژه مربوط به تسلط شما بر پروژه است.**
- **استفاده از ابزارهای هوش مصنوعی به عنوان یک دستیار برای یادگیری، رفع اشکال یا ایده‌پردازی مجاز است. اما، تولید مستقیم و کپی کردن بلوک‌های اصلی کد بدون درک کامل آن، منجر به کسر نمره خواهد شد. انتظار می‌رود دانشجویان قادر به توضیح کامل کدی که نوشته‌اند باشند.**
- **هرگونه تقلب یا کپی‌برداری از پروژه‌های دیگران یا منابع آنلاین، در صورت مشاهده، منجر به نمره صفر یا منفی برای کل پروژه خواهد شد.**

فازبندی پروژه و ددلاین ها:

پروژه در دو فاز اصلی تعریف می‌شود .انتظار می‌رود هر دو عضو گروه در تمام بخش‌های پروژه، شامل بک‌اند و فرانت‌اند، مشارکت فعال و تقریبا یکسانی داشته باشند.

✓ فاز اول: پیاده‌سازی رابط کاربری با داده‌های نمایشی

ددلاین :

هدف این فاز ساخت کامل ظاهر و تجربه کاربری اپلیکیشن در فلاتر، بدون نیاز به سرور است.

برای پیاده سازی این فاز با استفاده از لیست‌های ثابت (Hardcoded Lists) در کد فلاتر، صفحات را با داده‌های نمایشی پر کنید تا ظاهر برنامه کامل شود. در این فاز، نیازی به پیاده‌سازی منطق محاسباتی نیست.

خروجی تحویلی :سورس کد کامل پروژه فلاتر و دموی فرانت‌اند پروژه.

✓ فاز دوم: پیاده‌سازی سرور و یکپارچه‌سازی نهایی

ددلاین :

هدف این فاز ساخت موتور اصلی سامانه در جاوا و اتصال اپلیکیشن فلاتر به آن است.

در این فاز پیاده‌سازی کامل سرور جاوا و جایگزینی داده‌های نمایشی در فلاتر با ارتباط واقعی از طریق سوکت از شما انتظار میرود.

خروجی تحویلی :سورس کد کامل سرور (جاوا)، سورس کد نهایی کلاینت (فلاتر) و دموی زنده از عملکرد کامل سامانه.

توجه کنید تحویل هر 2 فاز به صورت حضوری انجام میشود . لذا برنامه ریزی مناسب برای تحویل پروژه را در هر 2 فاز داشته باشید.

سوالات خود را در گروه پروژه مطرح کنید.

موفق باشید

پیوست شماره 1

معماری پروژه و راهنمای تفکیک وظایف

بخش اول: فرانت‌اند- (Front-end)

فرانت‌اند، چهره‌ی اپلیکیشن شما و تنها بخشی است که کاربر نهایی به طور مستقیم با آن در ارتباط است. در این پروژه، این نقش توسط یک اپلیکیشن موبایل که با فریمورک فلاتر و زبان دارت توسعه می‌یابد، ایفا می‌شود. وظیفه اصلی فرانت‌اند، ارائه یک تجربه کاربری (UX) روان و یک رابط کاربری (UI) زیبا و کارآمد است. این بخش مسئولیت نمایش اطلاعات دریافت شده از سرور، مانند لیست حساب‌های بانکی، تاریخچه تراکنش‌ها و جزئیات گروه‌های هزینه را در قالبی قابل فهم و جذاب بر عهده دارد. همچنین، تمامی ورودی‌های کاربر، از جمله اطلاعات فرم‌های ثبت‌نام، ورود، انتقال وجه و ثبت هزینه‌ها، از طریق این لایه دریافت می‌شود.

نکته‌ی کلیدی در طراحی فرانت‌اند این است که این لایه نباید هیچ‌گونه منطق تجاری اصلی را اجرا کند. فرانت‌اند مانند یک پیام‌رسان هوشمند عمل می‌کند؛ درخواست‌های کاربر را دریافت کرده، آن‌ها را در قالب یک پیام استاندارد بسته‌بندی می‌کند و برای سرور ارسال می‌نماید. پس از آنکه سرور درخواست را پردازش کرد، فرانت‌اند پاسخ دریافتی (مانند پیام موفقیت‌آمیز بودن تراکنش یا خطای کمبود موجودی) را تفسیر کرده و به شکل مناسبی، مثلاً از طریق یک پیغام یا به‌روزرسانی صفحه، به کاربر نمایش می‌دهد.

نحوه پیاده‌سازی در فاز اول: برای تسهیل فرآیند توسعه و امکان کار موازی، در فاز اول پروژه نیازی به اتصال به سرور واقعی نیست. شما با استفاده از داده‌های نمایشی و ثابت (Hardcoded Data)، یعنی تعریف لیست‌هایی از کاربران، حساب‌ها و تراکنش‌ها به صورت دستی در کد فلاتر، می‌توانید تمام صفحات و جریان کاری اپلیکیشن را به طور کامل طراحی و پیاده‌سازی کنید. در فاز دوم، این داده‌های موقت با فراخوانی‌های واقعی به سرور جایگزین خواهند شد.

بخش دوم: بک‌اند - (Back-end)

بک‌اند، موتور قدرتمند و مرکز فرماندهی سامانه شماست که به زبان جاوا پیاده‌سازی می‌شود. این برنامه بر روی یک سرور اجرا شده و به طور مداوم در انتظار دریافت و پردازش درخواست‌ها از سوی کلاینت‌ها (اپلیکیشن‌های فلاتر) است. تمامی قوانین، محاسبات و عملیات حساس در این لایه متمرکز شده‌اند. بک‌اند مسئولیت‌های زیر را بر عهده دارد:

اجرای منطق تجاری: تمامی قوانین اصلی سامانه، از جمله اعتبارسنجی اطلاعات کاربر هنگام ورود، بررسی موجودی حساب پیش از تایید انتقال وجه، و الگوریتم‌های پیچیده محاسبه سهم هر فرد در هزینه‌های مشترک، در این بخش پیاده‌سازی می‌شوند (عملیات هایی مانند چک کردن انتخاب پسورد قدرتمند در سمت فرانت اند و نحوه نمایش لیست های کامل ارسال شده و ...در سمت فرانت اند انجام میشوند و جزء منطق تجاری به حساب نمی ایند و انجام ان ها در بخش بک اند موجب کندی سرور و فازایس بار سیستم میشود).

مدیریت داده‌ها: بک‌اند تنها بخشی است که مجوز دسترسی مستقیم به پایگاه داده پروژه(در اینجا، فایل‌های با فرمت .json) را دارد. این لایه مسئولیت خواندن، نوشتن، به‌روزرسانی و حذف داده‌ها را به طور ایمن بر عهده می‌گیرد.

مدیریت کاربران همزمان: یک سرور واقعی باید قادر به پاسخگویی به چندین کاربر به صورت همزمان باشد. طبق الزامات پروژه، سرور شما باید با استفاده از چند نخی (Multi-threading) طراحی شود، به طوری که برای هر کلاینت متصل، یک نخ (Thread) مجزا ایجاد گردد. این مکانیزم تضمین می‌کند که درخواست‌های یک کاربر، باعث مسدود شدن یا کندی سرویس‌دهی به سایر کاربران نشود.

بخش سوم: پروتکل ارتباطی

از آنجایی که کلاینت و سرور دو برنامه مستقل هستند که بر روی دستگاه‌های مختلفی اجرا می‌شوند، برای تبادل اطلاعات نیازمند یک زبان و کانال ارتباطی مشترک و تعریف‌شده هستند. در این پروژه، این ارتباط از طریق سوکت‌های TCP/IP برقرار می‌شود که یک کانال ارتباطی پایدار و قابل اعتماد بین دو نقطه در شبکه ایجاد می‌کند. داده‌هایی که در این کانال رد و بدل می‌شوند، باید ساختار مشخصی داشته باشند که برای هر دو طرف قابل فهم باشد. این ساختار توسط فرمت JSON (JavaScript Object Notation)، که یک استاندارد سبک و خوانا برای تبادل داده است، تعریف می‌شود.

روند یک تعامل نمونه (درخواست تاریخچه تراکنش‌ها):

فرآیند ارتباط را می‌توان به یک مکالمه‌ی پرسش و پاسخ تشبیه کرد. برای مثال، وقتی کاربر قصد مشاهده تراکنش‌های یک حساب را دارد، این گفتگو به شکل زیر انجام می‌شود:

شروع از کلاینت: اپلیکیشن فلاتر، بر اساس درخواست کاربر، یک پیام متنی با فرمت JSON می‌سازد که حاوی نوع دستور و اطلاعات لازم است. برای مثال

```
{
 "command": "GET_TRANSACTIONS",
 "data": {
        "account_number": "123456789"
        }
 }
```

ارسال به سرور: این پیام از طریق سوکت به سرور ارسال می‌شود.

پردازش در سرور: سرور پیام را دریافت کرده و آن را تحلیل می‌کند. با شناسایی دستور GET_TRANSACTIONS، به لایه‌های داخلی خود مراجعه کرده، به فایل‌های داده دسترسی پیدا می‌کند و لیست تراکنش‌های مربوط به حساب مشخص‌شده را استخراج می‌نماید.

ساخت و ارسال پاسخ: سرور نتیجه عملیات را در قالب یک پیام JSON جدید بسته‌بندی می‌کند که حاوی وضعیت عملیات (موفق یا ناموفق) و داده‌های درخواستی است. برای مثال

```
{
 "status": "SUCCESS",
 "data": [...]
 }
```

این پاسخ از طریق همان سوکت به کلاینت بازگردانده می‌شود.

نمایش در کلاینت: کلاینت پاسخ را دریافت، تحلیل کرده و اطلاعات موجود در آن را به صورت یک لیست خوانا و زیبا به کاربر نمایش می‌دهد.