پروژه شماره ۳ [نهایی]

نو سکّه

NO SEKE

ترم اول

مدرسه برنامه نویسی امیرکبیر

ترم تابستان ۹۸

مقدمه

در کشور نوور (Nowhere) واحد پول مردم سکه میباشد. پس از گذر زمان سکه‌های نزد برخی از افراد از پارو بالا رفته و خیلی زیاد شد. به طوری که این افراد دیگر فضایی برای نگهداری سکه هایشان نداشتند.

"پس از گذر زمان سکه‌های نزد برخی از افراد از پارو بالا رفت"

بنابراین طبق معمول عده ای برنامه نویس باهوش تصمیم گرفتند این مشکل را حل کنند.

تاسیس اولیه نو سکّه

برنامه نویسان کشور نوور تصمیم گرفتند واحد پول جدیدی را برای حل این مشکل ایجاد کنند به طوری که این واحد پول فضایی را اشغال نکند. آنها نام این واحد پول را نوسکه (سکه‌ی جدید) گذاشته و نام شرکت خود را نوسکه (no seke - بدون سکه) گذاشتند.

"هر نوسکه معادل یک سکه است."

لیست مشکلات سکّه

از طرفی تنها مشکل سکه های قدیمی فضای اشغالی آنها نبود، بلکه امنیت نگهداری از آن برای دزدیده نشدن و به علاوه زمان زیادی که طول میکشید تا سکه ها را انتقال دهند نیز از مشکلات سکه بود.

بنابراین برنامه نویسان شرکت نوسکه تصمیم گرفتند این مشکلات سکه را نیز حل کنند.

ایجاد اپلیکیشن نو سکّه

پس از تفکرات و ایده‌زنی‌های بسیار، تصمیم آنها براین شد که سیستمی بسازند که هر فرد در کشور نوور، یک حساب در سیستم نوسکه داشته باشد که مقدار اعتبار آن شخص را ذخیره نماید.

بنابراین هر شخص باید در سیستم نوسکه یک نام کاربری یکتا داشته باشد که با استفاده از آن بتوان به موجودی حساب او دسترسی پیدا کرد.

همچنین برای داشتن امنیت حسابها نیاز است که هر حساب کاربری یک رمز عبور هم داشته باشد.

"حساب کاربری"

قسمت های مختلف نوسکّه

برنامه نوسکه شامل چند بخش مختلف میباشد که در زیر آمده است:

- مشاهده مشخصات همه کاربران فعلی
- مشاهده میزان موجودی در یک حساب با داشتن نام کاربری و رمز عبور
- انتقال وجه از حساب "الف" به حساب "ب" با داشتن رمز عبور حساب "الف"
- افزایش وجه برای یک حساب (بدون نیاز به رمز عبور)
- ایجاد حساب کاربری جدید با نام کاربری و رمز عبور
- مشاهده تعداد حساب های کاربری موجود (فقط توسط ادمین)
- مشاهده مجموع موجودی همه حسابها (فقط توسط ادمین)

پیاده سازی نوسکّه

برنامه‌نویسان شرکت نوسکّه توانسته اند یک برنامه پایه برای سیستم مورد نظرشان پیاده سازی کنند. همچنین کارایی‌های مورد نیاز سیستمشان را نیز ذکر کرده اند.

آنها برای نگهداری هر حسابهای کاربری یک کلاس با نام User طراحی کرده اند که دارای سه مشخصه (attribute) میباشد.

User	
username	“Jack”
password	“abcd”
credit	1000

همچنین برای نگهداری همه کاربران و انجام پردازشهای مختلف روی آنها یک کلاس دیگر به نام Database پیاده سازی کرده اند که در یک لیست به نام users همه کاربران را نگهداری میکند. این کلاس یک ویژگی دیگر به نام admin_password دارد که برای بعضی از عملیات نیاز میشود.

همچنین همه اطلاعات کاربران در یک فایل متنی ذخیره شده اند که ارتباط با آن فایل نیز از وظایف این کلاس میباشد.

در فایل متنی accounts.ck اطلاعات کاربری همه کاربران موجود میباشد و کلاس Database در هر بار اجرا ابتدا محتویات آن فایل را خوانده و لیست حسابهای کاربری را در users نگهداری میکند.

قسمت های مختلف نوسکّه

برنامه نوسکه شامل چند بخش مختلف میباشد که در زیر آمده است.

خواندن اطلاعات کاربران

در ابتدای اجرای برنامه، کلاس database اطلاعات همه کاربران را از فایل accounts.ck خوانده و درself.users ذخیره میکند. این کار از طریق اجرای متود read انجام میشود.

این قسمت توسط برنامه نویسان نوسکه پیاده سازی شده و نیازی به انجام شما نیست.

ذخیره اطلاعات کاربران

همانطور که گفته شد اطلاعات کاربران بایستی در فایل accounts.ck ذخیره شود. با انجام هر تغییر روی اطلاعات کاربران فعلی و یا اضافه و حذف کردن کاربران نیاز به انجام عمل ذخیره سازی است. این کار با اجرای متود save انجام میشود. پیاده سازی متود save هم توسط برنامه نویسان نوسکه انجام شده ولی نیاز است شما در صورت تغییر اطلاعات کاربران یک بار متود save را فراخوانی کنید تا اطلاعات درون فایل ذخیره شوند.

۱. مشاهده مشخصات همه کاربران

در این قسمت نیاز است شما یک متود برای کلاس Database پیاده سازی کنید به طوری که اطلاعات همه کاربران فعلی را چاپ کند. نام این متود print_all میباشد و نیاز است شما بدنه آنرا بنویسید.

این عمل مخصوص ادمین میباشد و برنامه در صورت درخواست این عمل، از شما رمز عبور ادمین را میپرسد. بنابراین شما باید در این متود ابتدا برابر بودن رمز عبور وارد شده با admin_password را چک کنید.

۲. مشاهده تعداد کل کاربران

در این قسمت لازم است درون متود get_number_of_users را طوری کامل کنید که تعداد همه کاربران موجود را چاپ کند. (کاربران درون self.users میباشند.)

این عمل مخصوص ادمین میباشد و برنامه در صورت درخواست این عمل، از شما رمز عبور ادمین را میپرسد. بنابراین شما باید در این متود ابتدا برابر بودن رمز عبور وارد شده با admin_password را چک کنید.

۳. مشاهده مجموع موجودی همه حسابها

بدنه متود get_total_amount_of_credits را طوری کامل کنید که مجموع همه موجودی های کاربران را چاپ کند.

این عمل مخصوص ادمین میباشد و برنامه در صورت درخواست این عمل، از شما رمز عبور ادمین را میپرسد. بنابراین شما باید در این متود ابتدا برابر بودن رمز عبور وارد شده با admin_password را چک کنید.

۴. مشاهده موجودی یک حساب خاص

در این قسمت شما باید از کاربر نام کاربری و رمز عبور را گرفته و سپس در صورت وجود نام کاربری او و همچنین درست بودن رمز عبور موجودی او را نمایش دهید.

برای اینکار لازم است ابتدا تابع user_exists را کامل کنید. وظیفه این تابع چک کردن وجود کاربری با نام کاربری username میباشد. این تابع در صورت وجود داشتن کاربر مقدارTrue و در غیر اینصورت False برمیگرداند.

سپس متود username_matches_password را کامل کنید. این متود نام کاربری و رمز عبور را گرفته و اگر همخوانی داشته باشند مقدار True برمیگرداند. در غیر اینصورت مقدار False برمیگرداند.

پس از انجام این دو کار تابع show_credit_of_user را کامل کنید. این تابع ابتدا دو شرط قبلی را چک میکند و سپس در صورت درست بودن آنها مقدار موجودی کاربر را نمایش میدهد.

۵. اضافه کردن یک کاربر جدید

این عمل را باید در متود add_user انجام دهید. به این صورت که ابتدا چک کنید که کاربر با نام کاربری گرفته شده موجود نباشد. در صورت موجود نبودن آن نام کاربری یک User جدید بسازید و آنرا به لیست کاربران فعلی اضافه کنید. سپس عمل ذخیره سازی را انجام دهید.

۶. افزودن موجودی یک حساب کاربری

در این قسمت لازم است با گرفتن نام کاربری و رمز عبور یک کاربر و همچنین مقدار مبلغ، آن مبلغ را به موجودی آن کاربر خاص اضافه کنید و موجودی او را بعد از اضافه کردن مبلغ نمایش دهید.

این کار را باید در بدنه متود add_credit_to_user پیاده سازی کنید به این صورت که با گرفتن نام کاربری و رمز عبور ، ابتدا مانند قسمت نمایش موجودی، وجود داشتن کاربر و مطابقت رمزعبور را چک کنید. سپس کاربر را پیدا کرده و موجودی او را افزایش دهید. سپس موجودی جدید اورا چاپ کنید و سپس عمل ذخیره سازی را انجام دهید.

۷. انتقال وجه بین دو حساب

این کار را باید در متود send_credit_from_user_to_user پیاده سازی کنید.

این تابع قرار است مبلغ انتقالی، نام کاربری و رمز عبور کاربر اول، و نام کاربری کاربر دوم را بگیرد و آن مبلغ را از حساب کاربر اول به حساب کاربر دوم انتقال دهد.

برای این کار لازم است ابتدا وجود داشتن دو کاربر را چک نمایید. سپس همخوانی رمز عبور و نام کاربری را برای کاربر اول چک کنید. پس از آن درصورتی که کاربر اول حداقل به اندازه مبلغ انتقالی در حساب خود پول داشته باشد، آن مبلغ را از حساب او کم کرده و به حساب شخص دوم واریز کنید. در صورت موفقیت آمیز بودن عملیات موجودی حساب کاربر اول را چاپ کنید و در غیر این صورت خطای مناسب را چاپ کنید. سپس فراموش نکنید که عمل ذخیره سازی را انجام دهید.

ورودی و خروجی کاربر

قسمت خواندن ورودی و خروجی کاربر توسط برنامه نویسان نوسکه پیاده سازی شده. برای اجرا و تست برنامه کافی است شما فایل server.py را اجرا نمایید. این فایل از محتویات فایل db.py استفاده میکند که قرار است توسط شما پیاده سازی شود. نیاز به تغییر محتویات فایل server.py توسط شما نمیباشد.

توجه کنید که شما تنها باید کدهای موجود در db.py را تکمیل کنید و در بقیه فایلها چیزی ننویسید.

امتیازی

سعی کنید قسمتهایی که توسط برنامه نویسان نوسکه پیاده سازی شده را بخوانید و نحوه اجرای آنرا توضیح دهید.

نکات قابل توجه

- مهلت زمانی انجام این پروژه دو هفته میباشد.
- برنامه شما باید به زبان پایتون۳ نوشته شده باشد.
- پس از نوشتن برنامه، برنامه خود را با ورودی‌های مختلف تست کنید تا در هنگام تحویل به مشکل نخورید.
- این پروژه، یک پروژه انفرادی میباشد و در صورت مشابه بودن کد دو دانش‌آموز، نمره این پروژه برای هر دو آنها از بین میرود.
- در صورت داشتن هر گونه سوال، ابتدا به گوگل، سپس به دوستان خود و در پایان به استاد خود مراجعه نمایید.