مسابقات کشوری خلاقیت و نوآوری آیوکاپ

شیوه‌نامه اجرایی بخش چالش‌های برنامه‌نویسی و هوش مصنوعی

بهمن ماه ۱۴۰۳ – اردیبهشت ۱۴۰۴

مقدمه:

بخش چالش‌ها به‌عنوان یکی از حرفه‌ای‌ترین قسمت‌های مسابقات آیوکاپ طراحی شده است. این بخش بستری را فراهم می‌کند تا شرکت‌کنندگان با استفاده از دانش و خلاقیت خود، به حل مسائل کاربردی در حوزه‌های وب و هوش مصنوعی بپردازند. هدف این بخش، ارتقاء مهارت‌های شرکت‌کنندگان و آماده‌سازی آن‌ها برای چالش‌های واقعی دنیای فناوری است.

بخش چالش‌های برنامه‌نویسی و هوش مصنوعی چهارمین دوره مسابقات کشوری آیوکاپ شامل ۴ بخش می‌باشد:

۱. **چالش‌های برنامه‌نویسی وب:** شامل پنج چالش که در آن مهارت‌های طراحی و توسعه وب سنجیده می‌شود.
۲. **چالش‌های هوش مصنوعی:** شامل پنج چالش که نیازمند تحلیل داده‌ها و پیاده‌سازی الگوریتم‌های یادگیری ماشین و هوش مصنوعی است.
۳. **چالش‌های سازمانی:** در این بخش، سازمان‌ها پروژه‌ها و مسائل واقعی خود را به‌عنوان چالش مطرح کرده تا توسط شرکت‌کنندگان پیاده‌سازی شوند.
٤. **چالش جایزه بزرگ:** یک چالش منحصربه‌فرد که با هدف رقابت در سطح بالا برگزار می‌شود.

هزینه شرکت در بخش چالش‌ها:

- هزینه شرکت در هر چالش (وب، هوش مصنوعی یا چالش‌سازمانی): مبلغ ۸۰۰,۰۰۰ تومان.

نکته: با پرداخت این هزینه، شرکت‌کننده مجاز به انجام تنها یک چالش است. در صورت تمایل به شرکت در چالش‌های بیشتر، برای هر چالش اضافی مبلغ ۸۰۰,۰۰۰ تومان دریافت می‌شود.

- هزینه شرکت در چالش جایزه بزرگ: مبلغ ۱,۰۰۰,۰۰۰ تومان.

قوانین کلی مسابقه:

- استفاده از سورس‌های آماده موجود در اینترنت مانند (Github) برای الهام گرفتن مجاز است، اما استفاده مستقیم از مولدهای هوش مصنوعی نظیر ChatGPT که کد آماده ارائه می‌دهند ممنوع است.
- کمک گرفتن از برنامه‌نویسان حرفه‌ای یا واگذاری نوشتن کد به افراد دیگر کاملاً ممنوع است. در صورت مشاهده،کد شما بررسی نخواهد شد و از مسابقات حذف خواهید شد.
- نام فایل‌های ارسالی باید به صورت زیر باشد: MahdiGhasri_p۲
- فایل نهایی چالش باید به‌صورت یک فایل فشرده (ZIP) ارسال شود.
- مهلت ارسال کدها تا زمان تعیین‌شده توسط دبیرخانه است و پس از پایان مهلت، هیچ فایلی پذیرفته نخواهد شد.

چالش‌ها و پروژه‌های برنامه‌نویسی وب

چالش اول: پیاده‌سازی سبد خرید (p۱)

در این چالش، شرکت‌کنندگان می‌بایست با استفاده از HTML، CSS و JavaScript ، برنامه‌ای طراحی کنند که در آن، ۱۰ محصول در انبار موجود باشد. کاربر باید قادر باشد محصولات مورد نظر خود را به سبد خرید اضافه کند، همچنین امکان حذف و ویرایش (تعداد سفارش از هر محصول) را داشته باشد.

ویژگی‌های خاص چالش به شرح زیر است:

- اگر قیمت هر محصول بالای ۱۰۰ هزار تومان باشد، هزینه ارسال برای آن محصول محاسبه نخواهد شد.
- اما اگر قیمت محصول پایین‌تر از ۱۰۰ هزار تومان باشد، به ازای هر محصول با قیمت کمتر از این مقدار، مبلغ ۵ هزار تومان هزینه ارسال به جمع کل هزینه‌ها افزوده خواهد شد.
- علاوه بر هزینه‌ها، ۱۰ درصد مالیات بر ارزش افزوده نیز به مجموع هزینه‌های سبد خرید اضافه خواهد شد.

در نهایت، کاربر قادر خواهد بود که سبد خرید خود به همراه تمامی هزینه‌ها را مشاهده و تایید کند.

پیش‌نیازهای حل چالش:

- پیاده‌سازی چالش با HTML، CSS و JavaScript ضروری است.

وظایف شرکت‌کنندگان:

- پیاده‌سازی تمامی اهداف چالش طبق توضیحات ارائه‌شده.
- طراحی رابط کاربری جذاب و کاربرپسند با استفاده از فریم‌ورک.
- طراحی واکنش‌گرا (Responsive) به‌گونه‌ای که برنامه در تمامی دستگاه‌ها به درستی نمایش داده شود.

توضیحات تکمیلی:

- شرکت‌کنندگان می‌توانند از فریم‌ورک‌های محبوبی همچون Bootstrap برای طراحی این چالش استفاده کنند.
- توجه داشته باشید که پیاده‌سازی تمامی منطق چالش تنها باید با استفاده از زبان برنامه‌نویسی JavaScript انجام شود.

نکته مهم: در صورتی که شرکت‌کنندگان چالش خود را با دامنه مشخصی آنلاین کنند، امتیاز بیشتری به آن‌ها تعلق خواهد گرفت.

چالش دوم: تشخیص کد مورس (P۲)

کد مورس یک سیستم رمزنگاری است که برای هر حرف یا عدد، یک کد خاص به‌صورت ترکیب‌هایی از نقطه (.) و خط (-) تعریف می‌شود. در این چالش، شرکت‌کنندگان باید برنامه‌ای طراحی کنند که قادر باشد کد مورس دریافتی را به زبان انگلیسی تبدیل کند. توجه داشته باشید که در این چالش، پنج جمله به زبان کد مورس برای شما طراحی شده است و برنامه شما باید این جملات را به زبان انگلیسی نمایش دهد.

ویژگی‌های خاص چالش به شرح زیر است:

- برنامه شما باید توانایی تبدیل هر جمله به کد مورس به زبان انگلیسی را داشته باشد، نه تنها جملات ارائه‌شده در این چالش، بلکه جملات جدید نیز باید به‌درستی به معادل انگلیسی تبدیل شوند.

جملات مشخص شده که باید به زبان انگلیسی نمایش داده شوند (ترتیب از چپ به راست):

.- .. --- -.-. ..- .--. / -. .- - .. --- -. .- .-.. / -.-. .-. . .- - .. ...- .. - -.-- / .- -. -.. / .. -. -. --- ...- .- - .. --- -. / -.-. ---
-- .--. . - .. - .. --- -.

.- .. --- .-.. . .- .-. -. .- -.-. .- -.. . -- -.-- --- ..-. .--. .-. --- --. .-. .- -- -- .. -. --. .- -. -.. .- .-. - .. ..-. .. -.-. .. .- .-
.. .. -. - . .-.. .-.. .. --. . -. -.-. .

وظایف شرکت‌کنندگان:

- پیاده‌سازی تمامی اهداف چالش طبق توضیحات ارائه‌شده.
- طراحی رابط کاربری جذاب و کاربرپسند با استفاده از فریم‌ورک.
- طراحی واکنش‌گرا (Responsive) به‌گونه‌ای که برنامه در تمامی دستگاه‌ها به درستی نمایش داده شود.

توضیحات تکمیلی:

- شرکت‌کنندگان می‌توانند از فریم‌ورک‌های محبوبی همچون Bootstrap برای طراحی این چالش استفاده کنند.
- توجه داشته باشید که پیاده‌سازی تمامی منطق چالش تنها باید با استفاده از زبان برنامه‌نویسی JavaScript انجام شود.

چالش سوم: بازی هنگ‌من Hangman (P۳)

در این چالش، شرکت‌کنندگان می‌بایست با استفاده از زبان برنامه‌نویسی JavaScript، نسخه‌ای از بازی Hangman(هنگ‌من) را پیاده‌سازی کنند. بازی هنگ‌من یک بازی کلمات است که در آن یک کلمه به صورت مخفی نمایش داده می‌شود و بازیکن باید با حدس زدن حروف، کلمه مخفی را پیدا کند. بازیکن تنها تعداد مشخصی تلاش (حدس حرف) دارد و در صورتی که تعداد تلاش‌ها به پایان برسد، بازی به اتمام می‌رسد.

ویژگی‌های خاص چالش به شرح زیر است:

- در این بازی، کلمه به صورت مخفی (با استفاده از خط تیره یا فضای خالی) نمایش داده می‌شود و بازیکن باید حروف را به ترتیب حدس بزند.
- هر بار که بازیکن یک حرف صحیح را حدس بزند، آن حرف در مکان صحیح خود در کلمه ظاهر می‌شود.
- اگر بازیکن یک حرف اشتباه را حدس بزند، تعداد تلاش‌های باقی‌مانده کاهش می‌یابد.
- پس از تمام شدن تعداد تلاش‌ها یا حدس درست تمام حروف، بازی به اتمام می‌رسد.
- علاوه بر کلمات پیش‌فرض، شرکت‌کنندگان می‌توانند امکانی برای افزودن کلمات جدید به بازی فراهم کنند.

وظایف شرکت‌کنندگان:

- پیاده‌سازی منطق بازی Hangman با استفاده از JavaScript
- طراحی رابط کاربری ساده و جذاب برای نمایش کلمه مخفی، تعداد تلاش‌ها و تلاش‌های انجام‌شده.
- نمایش پیغام‌های مناسب در صورت برد یا باخت بازیکن.
- طراحی قابلیت‌هایی برای شروع بازی جدید و ریست کردن بازی پس از اتمام.

توضیحات تکمیلی:

- شرکت‌کنندگان می‌توانند از فریم‌ورک‌های مختلف مانند Bootstrap برای طراحی ظاهر بازی استفاده کنند.
- تمامی منطق بازی باید به‌طور کامل با استفاده از JavaScript پیاده‌سازی شود.

چالش چهارم: مدیریت پارکینگ (P۴)

در این چالش، شرکت‌کنندگان باید برنامه‌ای طراحی کنند که به مدیریت یک پارکینگ با فضای مشخص بپردازد. در این برنامه، پارکینگ باید به‌گونه‌ای طراحی شود که بر اساس فضای خالی موجود، برای هر نوع ماشین یک جایگاه خاص و منظم انتخاب کند. هدف این است که ماشین‌ها به‌طور هوشمند در کنار یکدیگر قرار گیرند تا پارکینگ به‌صورت کامل و بدون فضای خالی پر شود.

ویژگی‌های خاص چالش به شرح زیر است:

- پارکینگ باید دارای تعداد مشخصی جایگاه باشد که برخی از آن‌ها در ابتدا خالی است و برخی دیگر ممکن است قبلاً توسط ماشین‌های دیگر پر شده باشند.
- برنامه باید به‌گونه‌ای عمل کند که ماشین‌ها به طور خودکار و هوشمند در مکان‌های خالی پارکینگ قرار گیرند.
- برنامه باید فضای خالی را شبیه‌سازی کرده و ماشین‌ها را در کنار یکدیگر به‌گونه‌ای قرار دهد که هیچ فضای خالی در بین ماشین‌ها باقی نماند.
- برای هر نوع ماشین، باید ظرفیت خاصی در پارکینگ تعیین شده باشد (برای مثال: ماشین‌های بزرگ باید جایگاه ۳*۳، ماشین‌های کوچک جایگاه ۱*۱ و غیره).
- اگر یک ماشین وارد پارکینگ شود، سیستم باید بررسی کند که کدام جایگاه‌ها برای آن ماشین مناسب است و سپس آن را در جایگاه مشخص قرار دهد.

وظایف شرکت‌کنندگان:

- پیاده‌سازی منطق مدیریت پارکینگ با استفاده از JavaScript
- طراحی سیستم تخصیص جایگاه به ماشین‌ها بر اساس فضای خالی موجود.
- ایجاد الگوریتمی که به طور خودکار و بدون خطا ماشین‌ها را در جایگاه‌های مناسب قرار دهد.
- نمایش وضعیت پارکینگ در زمان واقعی، به‌گونه‌ای که وضعیت هر جایگاه (خالی یا پر) به‌طور واضح مشخص باشد.

توضیحات تکمیلی:

- شرکت‌کنندگان می‌توانند از فریم‌ورک‌های مختلف مانند Bootstrap برای طراحی ظاهر بازی استفاده کنند.
- تمامی منطق و الگوریتم‌ها باید با استفاده از JavaScript پیاده‌سازی شوند.

چالش پنجم: سیستم ایجاد مدرک دیجیتال با امضای دیجیتال و QRCode (P۵)

در این چالش، شرکت‌کنندگان باید سیستمی را طراحی کنند که به افراد این امکان را بدهد تا مدرک دیجیتال خود را ایجاد کرده و از آن استعلام بگیرند. ابتدا، فرد باید اطلاعات خود را در یک Input وارد کرده و پس از آن، سیستم به طور خودکار یک مدرک دیجیتال ایجاد می‌کند که شامل اطلاعات فرد، یک QR Code و یک امضای دیجیتال یونیک برای آن فرد باشد. این مدرک باید با مختصات مشخص در مدرک قرار گیرد. پس از ایجاد مدرک، در صورت درخواست استعلام از فرد، با وارد کردن شناسه مدرک یا اسکن QR Code، مدرک قبلی با تمامی اطلاعات شامل نام فرد، QR Code و امضای دیجیتال یونیک دوباره نمایش داده شود.

ویژگی‌های خاص چالش به شرح زیر است:

- سیستم باید به گونه‌ای عمل کند که فرد اطلاعات خود را به راحتی وارد کند و سپس مدرک دیجیتال با تمامی جزئیات (نام، QR Code و امضای دیجیتال یونیک) به صورت خودکار ایجاد شود.
- باید مکان‌های مشخص برای نمایش اطلاعات فرد در مدرک وجود داشته باشد.
- پس از ایجاد مدرک، فرد باید بتواند با وارد کردن شناسه مدرک یا اسکن QR Code، مدرک خود را با تمامی جزئیات از جمله امضای دیجیتال یونیک مشاهده کند.
- QR Code باید به‌طور منحصر به فرد برای هر مدرک ایجاد شود.

وظایف شرکت‌کنندگان:

- پیاده‌سازی فرم ورودی برای وارد کردن اطلاعات فرد.
- طراحی سیستم ایجاد مدرک دیجیتال که شامل اطلاعات فرد، QR Codeو امضای دیجیتال یونیک باشد.
- طراحی سیستم استعلام مدرک که با وارد کردن شناسه یا اسکنQR Code ، مدرک دیجیتال با تمامی اطلاعات نمایش داده شود.
- پیاده‌سازی منطق لازم برای ایجاد QR Code و امضای دیجیتال یونیک برای هر مدرک.
- پیاده‌سازی تعامل بین JavaScript (برای تعاملات سمت کاربر) وPHP (برای پردازش اطلاعات و مدیریت پایگاه داده).

توضیحات تکمیلی:

- برای ایجاد QR Code، شرکت‌کنندگان می‌توانند از کتابخانه‌های JavaScript مانند qrcode.js استفاده کنند.
- برای ایجاد امضای دیجیتال یونیک، از روش‌های رمزنگاری موجود در PHP باید استفاده شود.
- تمام اطلاعات باید در یک پایگاه داده ذخیره شده و سیستم باید بتواند از طریق شناسه یاQR Code ، مدرک مورد نظر را بازیابی کند.

نکته مهم: همچنین شرکت کنندگان باید چالش خود را بر روی یک دامنه آنلاین کنند.

چالش جایزه بزرگ آیوکاپ (برنامه‌نویسی وب)

پیاده‌سازی سیستم ساخت رزومه آنلاین

در این چالش، شرکت‌کنندگان باید یک سیستم تحت وب طراحی کنند که به کاربران امکان ایجاد و دانلود رزومه‌های شخصی‌شان را بدهد. این سیستم باید به گونه‌ای طراحی شود که اطلاعات رزومه به‌صورت گرافیکی بر روی یک تصویر رزومه چاپ شود. این فرآیند باید کاملاً خودکار باشد و کاربر تنها با وارد کردن اطلاعات رزومه خود، بتواند نسخه نهایی را به‌صورت تصویر دریافت کند.

ویژگی‌های خاص چالش به شرح زیر است:

۱. **طراحی فرم ورودی اطلاعات**

- ورود اطلاعات رزومه: کاربر باید بتواند اطلاعات شخصی خود را مانند نام و نام خانوادگی، تحصیلات، تجربه کاری، مهارت‌ها و زبان‌ها در فرم وارد کند.
- ذخیره‌سازی اطلاعات در دیتابیس: تمامی اطلاعات وارد شده باید در پایگاه داده ذخیره شوند تا کاربر بتواند در آینده به راحتی به آن‌ها دسترسی داشته باشد و در صورت نیاز، آن‌ها را ویرایش کند.
- ساخت تصویر رزومه: بعد از ورود اطلاعات، سیستم باید به‌صورت خودکار یک تصویر رزومه تولید کند که تمامی اطلاعات وارد شده را در قالب یک رزومه حرفه‌ای و گرافیکی نمایش دهد.
    - چاپ اطلاعات بر روی تصویر: اطلاعات مانند نام و نام خانوادگی، تحصیلات، تجربه کاری و مهارت‌ها باید به‌صورت متنی بر روی تصویر چاپ شوند.

وظایف شرکت‌کنندگان:

۱. **طراحی رابط کاربری با استفاده از** HTML / CSS / BOOTSTRAP
۲. **مدیریت داده‌ها**
۳. **تولید تصویر رزومه**

- طراحی قالب رزومه و چاپ اطلاعات وارد شده در مکان‌های مناسب روی تصویر.

۴. **امکان ویرایش و به‌روزرسانی رزومه**

توضیحات تکمیلی:

- فرمت فایل رزومه: رزومه باید به‌صورت تصویری مانند (PNG یا JPEG) برای دانلود در دسترس باشد.

چالش‌ها و پروژه‌های هوش‌مصنوعی

چالش اول: طراحی دستیار صوتی برای باز کردن ایمیل و جستجوی سایت‌های مرتبط (AI)

در این چالش، شرکت‌کنندگان باید یک برنامه طراحی کنند که با استفاده از دستورات صوتی کاربر، وظایف زیر را انجام دهد: باز کردن صفحه ایمیل کاربر و یا جستجو و باز کردن سایت‌های مرتبط با موضوعی که کاربر به صورت صوتی اعلام می‌کند (مانند سوپرمارکت‌ها).

اهداف چالش:

۱. تبدیل گفتار به متن: تبدیل صدای کاربر به متن با دقت بالا.
۲. تحلیل متن ورودی: شناسایی دستورات مربوط به باز کردن ایمیل یا جستجوی موضوعی.

وظایف شرکت‌کنندگان:

این برنامه باید از کتابخانه‌های Selenium یا Beautiful Soup برای تعامل با مرورگر و جستجو استفاده کند.

- پیاده‌سازی تبدیل گفتار به متن:
  - استفاده از کتابخانه SpeechRecognition برای پردازش گفتار.
- تحلیل متن و شناسایی دستورات:
  - طراحی الگوریتمی برای شناسایی دستور "ایمیل را باز کن" و موضوعات دیگر مانند "سوپرمارکت."
- باز کردن ایمیل:
  - استفاده از Selenium برای باز کردن Gmail یا سرویس ایمیل دیگر.
  - ورود به آدرس https://mail.google.com
  - ارسال اطلاعات ورود (نام کاربری و رمز عبور) به صورت آزمایشی یا هدایت کاربر به صفحه ورود.
- جستجوی موضوعی:
  - دریافت موضوع از متن تبدیل‌شده و استفاده از موتور جستجوی گوگل.

نمونه داده‌ها و سناریوهای آزمایشی:

ورودی صوتی: سوپرمارکت را جستجو کن

- **عملکرد مورد انتظار:** جستجوی "سوپرمارکت" در گوگل.

چالش دوم: تحلیل داده‌های بانکی از فایل CSV (A۲)

شرکت‌کنندگان باید برنامه‌ای طراحی کنند که داده‌های تراکنش‌های بانکی ذخیره‌شده در یک فایل CSV را تحلیل کند. این برنامه باید توانایی استخراج اطلاعات کلیدی از داده‌ها، شناسایی الگوهای تراکنش و ارائه گزارش‌های آماری و تحلیلی را داشته باشد. داده‌ها شامل اطلاعاتی مانند تاریخ تراکنش، مبلغ تراکنش، و نوع تراکنش (پرداخت یا دریافت) هستند.

اهداف چالش:

۱. آشنایی با ابزارهای پردازش داده‌ها در Python مانند pandas

۲. تحلیل دقیق داده‌های تراکنش مالی و شناسایی الگوهای آماری.

وظایف شرکت‌کنندگان:

۱. بارگذاری داده‌ها از فایل CSV

۲. شرکت‌کنندگان باید فایل داده‌ها را با استفاده از pandas بارگذاری کنند. داده‌ها شامل ستون‌های زیر خواهند بود:

- تاریخ تراکنش (Transaction Date)
- مبلغ تراکنش (Amount)
- نوع تراکنش، پرداخت یا دریافت (Type)

۳. تحلیل مبالغ تراکنش‌ها:

- محاسبه فراوانی بیشترین مبلغ ورودی (Receive) و خروجی (Pay)
- بررسی فراوانی بیشترین تراکنش‌های هزینه و خروجی در یک روز مشخص.

٤. تحلیل ماهانه و هفتگی:

چالش‌های اضافی (اختیاری):

- رسم نمودارهای خطی یا ستونی برای نمایش روند مبالغ تراکنش‌ها در روزها، هفته‌ها، و ماه‌ها.
- شناسایی اوج تراکنش‌های مالی در بازه‌های زمانی مشخص (مانند ساعات پرترافیک یا روزهای خاص).
- تحلیل میانگین و انحراف معیار مبالغ تراکنش‌ها.

چالش سوم: شبیه‌سازی صف انتظار بانک با صدای شخصی‌سازی شده (A۳)

شرکت‌کنندگان باید برنامه‌ای طراحی کنند که صف مشتریان در یک بانک را شبیه‌سازی کرده و علاوه بر مدیریت صف، از صدای شخصی‌سازی شده برای اعلام شماره و گیشه مربوط به هر مشتری استفاده کند. به‌طور مثال، وقتی مشتری وارد صف می‌شود، سیستم باید به او بگوید: "شماره ۷۸۹ به گیشه ۳ مراجعه کنید"، و این پیام باید با صدای کاربر پخش شود.

اهداف چالش:

۱. شبیه‌سازی دقیق صف مشتریان با استفاده از الگوریتم‌ها و ساختار داده‌ای مناسب.

۲. پیاده‌سازی سیستم پیام‌دهی صوتی با صدای شخصی‌سازی شده برای هر فرد.

۳. مدیریت بهینه زمان‌بندی سرویس‌دهی و تعداد مشتریان در صف.

وظایف شرکت‌کنندگان

- صف مشتریان باید با استفاده از یک ساختار داده‌ای مانند Queue پیاده‌سازی شود.
- مشتریان به ترتیب وارد صف شوند و هرکدام شماره و گیشه خود را دریافت کنند.
- به هر مشتری یک شماره منحصر به فرد داده می‌شود (مثلاً شماره ۷۸۹).
- به هر مشتری گیشه‌ای که باید به آن مراجعه کند (مثلاً گیشه ۳) تخصیص داده می‌شود.

- **پیاده‌سازی سیستم صوتی برای اعلام شماره و گیشه**
  - از کتابخانه‌های پردازش صوتی مانند GTTS (Google Text-to-Speech) یا pyttsx۳ استفاده کنید تا پیام‌ها با صدای شخصی‌سازی شده پخش شوند.
  - سیستم باید از اطلاعات موجود در صف استفاده کند تا به مشتریان اعلام کند که به کدام گیشه باید مراجعه کنند. مثلاً: "شماره ۷۸۹ به گیشه ۳ مراجعه کنید."
  - صدا می‌بایست شخصی‌سازی شود، برای مثال، با استفاده از صدای افراد مختلف (برای امتیاز بیشتر می‌توانید از صدای خودتان استفاده کنید).
- **شبیه‌سازی زمان سرویس‌دهی به مشتریان**
  - مدت زمان سرویس‌دهی به هر مشتری باید به صورت تصادفی یا ثابت تعیین شود.
  - پس از سرویس‌دهی به مشتری، سیستم باید به‌طور خودکار به مشتری بعدی شماره و گیشه تخصیص دهد.

چالش چهارم: تولید کلمات عبور امن (A٤)

شرکت‌کنندگان باید برنامه‌ای طراحی کنند که کلمات عبور امن و پیچیده تولید کند. این کلمات عبور باید شامل حروف کوچک، حروف بزرگ، اعداد، و نمادها باشند. برنامه باید تعداد و طول کلمات عبور را از کاربر دریافت کرده و آن‌ها را در فایل passwords.txt ذخیره کند. علاوه بر این، برنامه باید امنیت هر کلمه عبور را ارزیابی کند و سطح امنیت آن را گزارش دهد، مانند "متوسط"، "خوب" یا "عالی".

اهداف چالش:

١. مدیریت ورودی‌ها: توانایی دریافت و پردازش ورودی‌ها از کاربر.

٢. تولید داده‌های تصادفی: تولید کلمات عبور امن با استفاده از کاراکترهای تصادفی.

وظایف شرکت‌کنندگان:

١. دریافت ورودی از کاربر

٢. تولید کلمات عبور:

- هر کلمه عبور باید شامل ترکیبی از حروف کوچک، حروف بزرگ، اعداد و نمادها باشد.
- اطمینان از امنیت کلمات عبور (عدم استفاده از ترتیب ساده یا پیش‌بینی‌پذیر).

٣. بررسی امنیت کلمات عبور:

- هر کلمه عبور باید از نظر امنیتی ارزیابی شود و نمره‌ای مانند "متوسط"، "خوب" یا "عالی" دریافت کند.

٤. ذخیره‌سازی کلمات عبور

خروجی:

- فایل passwords.txt که شامل کلمات عبور تولیدشده است. در کنار هر کلمه عبور، ارزیابی سطح امنیتی آن نیز آمده است (عالی، خوب یا متوسط).

چالش پنجم: آموزش حرکات بدنسازی با استفاده از مدل‌سازی حرکت (A۵)

هدف این چالش پیاده‌سازی سیستمی است که از تکنیک‌های مدل‌سازی حرکت برای شناسایی حرکات بدن در ویدیوهای زنده (لایو) استفاده کند و به کاربران در انجام حرکات بدنسازی کمک کند. این سیستم باید قادر باشد حرکت‌های بدن را با استفاده از نقاط کلیدی (Keypoint) شبیه‌سازی کرده و در صورت وجود خطا یا ضعف در فرم حرکت، هشدار دهد یا اصلاحات لازم را پیشنهاد کند.

وظایف شرکت‌کنندگان:

- تشخیص بدن و نقاط کلیدی:
  - استفاده از تکنیک‌های Pose Estimation برای شناسایی نقاط کلیدی بدن در ویدیوهای زنده.
- به‌کارگیری مدل‌هایی مانند OpenPose یا MediaPipe برای استخراج نقاط کلیدی بدن (شانه‌ها، زانوها و ...).
- تشخیص حرکات صحیح و اشتباهات
- شبیه‌سازی حرکات بدنسازی و آموزش آن‌ها بر اساس اطلاعات دریافتی از نقاط کلیدی بدن.

خروجی:

- ارائه فیدبک به کاربر
  - نشان دادن اشتباهات با استفاده از رنگ‌ها (مثل قرمز برای اشتباهات) و یا نمایش متن راهنمایی.
- سیستم باید قابلیت تشخیص حرکت‌های مختلف بدنسازی مانند اسکوات (Squats)، پوش آپ (Push-ups)، ددلیفت ، بارفیکس و ... را داشته باشد.
- نمایش زنده ویدیو با خطوط و نقاط کلیدی:
  - نمایش ویدیو زنده (لایو) که روی آن نقاط کلیدی بدن در حال حرکت است.
  - رسم خطوط بین نقاط کلیدی برای نمایش وضعیت بدن در هر لحظه.

چالش جایزه بزرگ آیوکاپ (هوش مصنوعی)

بررسی صحت فیش‌های بانک

در این چالش، شرکت‌کنندگان باید یک برنامه طراحی کنند که فیش‌های بانکی را به صورت خودکار خوانده و اطلاعات داخل آن‌ها شامل مبلغ و زمان واریز را استخراج کند. سپس این اطلاعات باید با داده‌های موجود در اکسل آماده شده توسط کاربر مقایسه شوند. در نهایت، برنامه باید به کاربر اطلاع دهد که فیش واریزی با داده‌های اکسل هم‌خوانی دارد یا خیر.

اهداف چالش:

۱. آشنایی با پردازش تصویر و استخراج داده‌ها از فیش‌های بانکی.
۲. توانایی مقایسه داده‌ها بین فایل‌های اکسل و منابع آنلاین.
۳. ایجاد سیستمی برای بررسی صحت داده‌ها و اعلام نتایج به کاربر.

وظایف شرکت‌کنندگان:

- استخراج داده‌های فیش بانکی:
  - استفاده از تکنولوژی‌های پردازش تصویر مانند OCR (Optical Character Recognition) برای خواندن فیش‌های بانکی و استخراج اطلاعات مربوط به مبلغ واریزی و زمان واریز.
- مقایسه با داده‌های موجود در فایل اکسل:
  - داده‌های فیش‌های بانکی باید با داده‌های موجود در یک فایل اکسل که شامل جزئیات واریزهای ثبت شده در شرکت است، مقایسه شود.
  - اطلاعاتی مانند مبلغ واریز شده و زمان واریز باید از هر دو منبع استخراج و مقایسه شوند تا هم‌خوانی آن‌ها بررسی شود.

خروجی:

- گزارش نهایی:
  - پس از انجام مقایسه‌ها، برنامه باید به کاربر اطلاع دهد که آیا فیش واریزی با داده‌های فایل اکسل و بانک هم‌خوانی دارد یا خیر.
  - اگر داده‌ها تطابق نداشته باشند، باید علت دقیق (مانند تفاوت مبلغ یا زمان واریز) اعلام شود.
  - ایجاد بانک اطلاعاتی برای ذخیره‌سازی اطلاعات: ایجاد سیستم ذخیره‌سازی داخلی برای فیش‌ها و داده‌های واریزی که در گذشته بررسی شده‌اند.