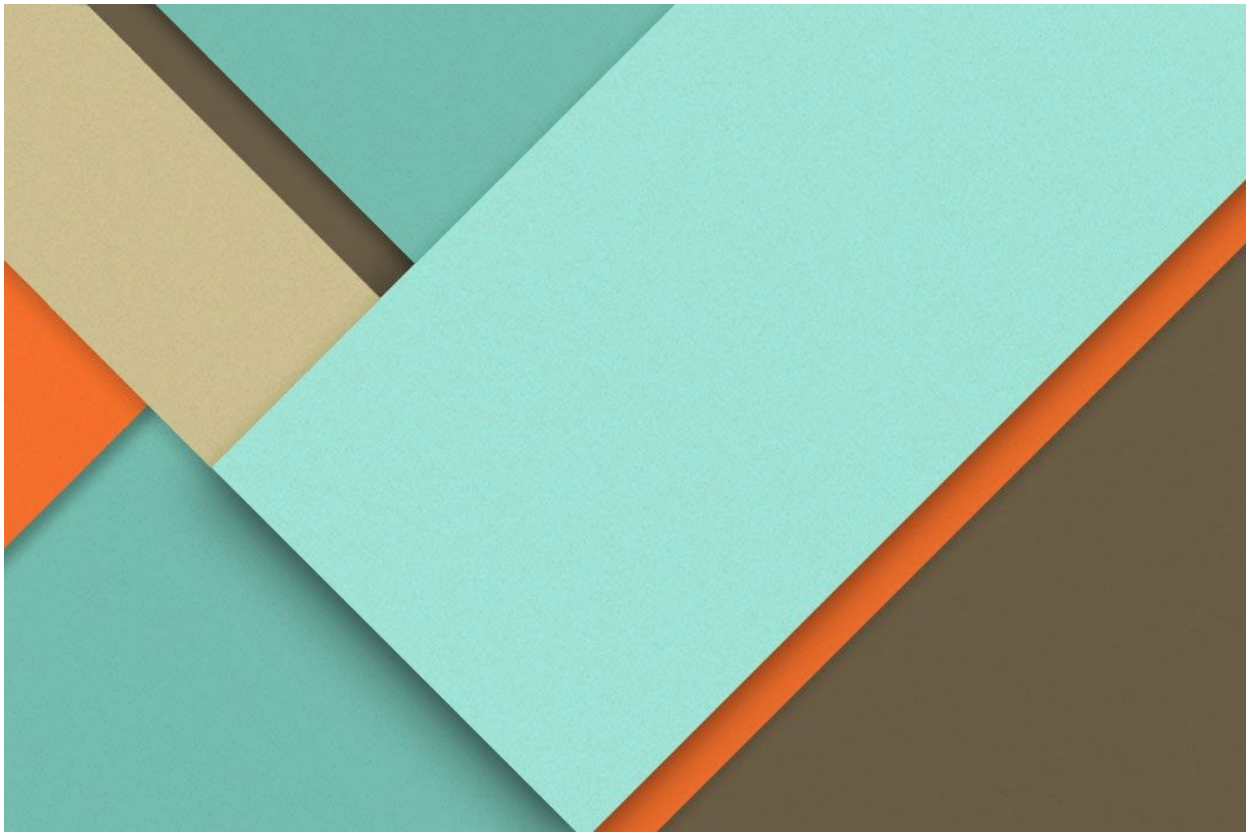

# Code Analyzer

محمد بابایی 96521074

مبین داریوش همدانی

استاد سعید پارسا

## بررسی کلی

امروزه برنامه نویس های زیادی وجود دارند که ادعای حرفه ای بودن و داشتن مهارت را دارند اما نکته اینجا است که چیزی که این برنامه نویس ها را از هم جدا میکند یعنی شغلشان و سطح مهارت آنها را مشخص میکند کیفیت کد و نرم افزاری است که می نویسند و اینکه اصول و قواعدی که امروزه به دست آورده ایم مثل اصول کتاب هایی که وجود دارد را رعایت کنند مثل کتاب code complete.

ما در این پروژه به بررسی دقیق اینکه کد شما چقد این اصول و قواعد را رعایت میکند پرداخته ایم.

## اهداف

1. کاربر بتواند کدی که میخواهد بررسی کند را وارد کند.
2. طراحی پروژه به صورت سایت که کاربران به راحتی بتوانند به این ابزار دست پیدا کرده و از آن استفاده کنند.
3. آنالیز و بررسی کردن کد به صورت کاملا موشکافانه و گرفتن ایراد هایی که با اصول موجود در کتاب code clean مغایرت دارد.
4. نشان دادن جای دقیق اینکه ایراد گرفته شده دقیقا در کجای کد قرار دارد و اینکه خط آن را به صورت دقیق به کاربر داده شود.
5. نوشتن دقیق جملات افراد بزرگ مثل robert c martin یا همان uncle bob.

## مشخصات فنی

تمامی موارد ذکر شده در بالا به صورت کامل پیاده سازی شده اند.

ما قادر به گرفتن ۲۰ ایراد از کد وارد شده هستیم.

قسمتی از پروژه با coco و قسمتی با استفاده از roslyn پیاده سازی شده است.

لیست ایراد هایی که به کاربر اطلاع داده میشود:

1. تشخیص nested loop های فراوان و متعدد.
2. تشخیص nested if statement های فراوان و متعدد.
3. تشخیص توابع طولانی و بلند.
4. تشخیص کلاس های طولانی و بلند.
5. تشخیص catch هایی که کاربر آن ها را خالی رها کرده است.
6. try statement هایی که دارای پیچیدگی زیادی هستند(try ها خود گیج کننده هستند نباید کد داخل آن ها نیز پیچیده باشد).
7. اسامی توابع و کلاس ها باید با معنی باشند.
8. توابع static بهتر است void تعریف نشوند.
9. if statement ها نباید دارای کد پیچیده باشند و بهتر است داخل آن ها fucntion call داشته باشیم تا یک کد پیچیده.
10. خط های افقی کد نباید خیلی طولانی باشند چون خوانایی کد را خدشه دار میکند.
11. آرگمان های توابع نباید از چهار تا بیشتر باشند.
12. آرگمان های توابع بهتر است صفر باشد اگر نشد یکی و اگر نشد به ترتیب ۲ و ۳ باشند و از ۴ تا باید اجتناب کرد.
13. نسبت تعداد خط comment های موجود در کد به کل خط های کد. این نسبت باید رعایت شود نه زیاد باشد و نه کم.
14. unit test هایی که در کد وجود دارد فقط و فقط باید یک چیز را test کنند و به عبارتی single responsible باشند و یک چیز را assert کنند.
15. اعداد بزرگی که در کد استفاده میشوند بهتر است به صورت constant تعریف شوند.(magic number)

...

تمامی موارد بالا بررسی میشوند و در صورت نقض شدن آن ها به کاربر اطلاع رسانی شده و حتی خط آن را در کد به کاربر اطلاع رسانی خواهیم کرد.