Веб апликација за шах која овозможува трансформација на вистинска слика од шаховска табла во дигитален формат и евалуација на позицијата со помош на шаховски алгоритам

Содржина

[Апстракт 2](#_Toc150793626)

[Вовед 2](#_Toc150793627)

[Бекенд 5](#_Toc150793628)

[Менаџирање со пајтон пакети 5](#_Toc150793629)

[Фронтенд 6](#_Toc150793630)

[Податочно множество 9](#_Toc150793631)

[Машински модели 9](#_Toc150793632)

[Модел за препознавање шаховска табла 9](#_Toc150793633)

[Модели за детекција 9](#_Toc150793634)

[Евалуација на шаховска позиција 11](#_Toc150793635)

[Проширување 12](#_Toc150793636)

[Заклучок 13](#_Toc150793637)

[Референци 13](#_Toc150793638)

## Апстракт

## Вовед

Во рамките на овој дипломски труд, се изработува целосна веб апликација со цел да им обезбедиме на шахистите вреден софтвер наменет за почетници и професионалци. Веб апликацијата овозможува анализа на шаховски позиции во реално време и дава совети за подобрување на потегата.

Целта на овој проект е да ги научи шахистите почетници да ги разбираат и анализираат шаховските позиции, како и да ги води кон подобри потега. Со употреба на алгоритми за машинско учење и машинска визија, нашата апликација овозможува трансформација на слика од физичкиот шах во дигитален формат и евалуација на шаховската позиција.

Преку три различни начини за внес на шаховската позиција, корисниците можат да ја стартуваат анализата. По праќањето на сликата до бекендот, апликацијата резултира со FEN (Forsyth-Edwards Notation) од шаховската позиција. Сепак, бидејќи не може да се постигне 100% прецизност на нетривијални слики, дозволуваме рачна корекција на детектираните грешки.

Потоа, корисниците имаат можност да внесат дополнителни информации за состојбата на играта и да го изградат целосниот FEN, вклучувајќи го информации како кој е на потег и дали има можност за рокада. Овие дополнителни информации се критични за градење на FEN.

По успешното завршување на процесот на внесување и корекција на информациите, корисниците се пренасочуваат кон страницата за евалуација на шаховската позиција. Тука, апликацијата им нуди графички приказ на најдобрите 3 потега на играчот кој е на потег, како и информации за моменталната предност на играчот, добиена со помош на шаховскиот алгоритам Stockfish верзија 16.

Со овој проект, нашата цел е да го обезбедиме шахот како забавна и образователна игра за почетници, истовремено им обезбедувајќи инструмент за подобрување на нивните шаховски вештини. Преку комбинирање на технологии и алгоритми од областите на веб дизајн, машинско учење и машинска визија, сме создале платформа која овозможува раст на шаховските способности кај корисниците.

Во продолжение на овој дипломски труд, ќе ги истражиме деталите на нашиот проект, вклучувајќи го описот на проблемот, технологиите кои ги користиме, архитектурата на бекенд и фронтенд, податочното множество, моделите за детекција на фигурите, описот на целата веб апликација, и проширувањето на истата. Ќе се фокусираме на нашите решенија и ќе објасниме како функционираат.

Во рамките на овој дипломски труд ќе разгледаме развој на целосна веб апликација кое крајниот софтверско решение ќе спои полиња од веб дизајн, машинско учење, машинска визија. На почетокот на апликацијата имаме 3 опции доаѓање до евалуација на шаховска позиција каде првиот начин е сликање на моменталниот физички шах, вториот е импортирање на веќе постоечка слика и третиот начин е директно пренасочување до страната за манипулација на шаховската табла. По праќање на сликата до бекендот, истата навлага во процес кој крајниот одговор (response) до фронтендот ќе биде FEN (Forsyth-Edwards Notation) од шаховската позиција дадена на сликата. Но бидејќи не е возможно да се направи модел со 100% прецизност на нетривијални слики, по евалуација на шаховската табла и шаховските фигури се пренасочува до страна која овозможува промена на грешно-детектирани полиња/фигури. По завршување на рачната промена на фигурите, се внесуваат дополнителни информации за состојбата на шаховската игра како: кој е на потег и дали белата и црната страна имаат можност за рокада, овие информации битни се бидејќи за целосно градење на FEN. По завршување на рачното внесување податоци, корисникот е пренасочен на страната за евалуација на шаховската позиција каде графички се прикажуваат најдобрите 3 потега на играчот што е на потег, исто така се прикажува моментално која страна води и за колку, добиена со помош на stockfish 16 шаховскиот алгоритам со помош на centi-pawn функцијата. Од тука играчите можат да продолжат со игра каде после секој потег се ажурира евалуацијата и стрелките за 3-те најдобри потега.

Опис на проблемот

Проблемот

Шахот е комплексна игра, која предизвикува играчите да размислуваат стратегиски и точно да разберат шаховските позиции. За почетниците, проникнувањето во светот на шахот и учењето на правилата и техниките може да биде предизвик.

Потребата:

Проектот има за цел да одговори на потребите на почетниците кои сакаат да учат и подобруваат своите шаховски вештини. Одличниот начин за учење и усовршување на шаховските способности е преку анализа на шаховските позиции и размислување за најдобрите потега.

Изазовите:

Играњето на шах во реално време, со способност за евалуација на потегата, може да биде комплексно и изазовно. Извршувањето на трансформација од физичка табла во дигитален формат и евалуација на шаховската позиција може да се среќе со известни технички предизвици и прецизност на детекцијата на фигурите.

Решението:

Проектот користи технологии од областите на машинско учење и машинска визија за да го решат овој проблем. Со алгоритми за детекција и класификација на фигурите, како и со користење на шаховски алгоритми за евалуација на потегата, апликацијата овозможува анализа на шаховската позиција и нуди совети за најдобрите потега.

Искористени технологии

Django: Како бекенд технологија, користите Django - популарен Python фрејмворк. Django ви овозможува да го развиете бекенд делот на вашиот проект, вклучувајќи ги операциите за детекција на фигурите, евалуација на позициите, и комуникацијата со фронтендот.

React: Фронтендот на вашиот проект е развиен во React, што е моќен JavaScript библиотека за создавање кориснички интерфејси. React ви дозволува да креирате интерактивен и прилагодлив интерфејс кој ќе го олесни корисниците да го следат процесот на анализа.

Stockfish 16: За шаховската евалуација, користите Stockfish 16, кој е јач шаховски алгоритам. Stockfish има висока репутација во светот на шахот како еден од најмоќните и точни алгоритми за анализа на шаховски позиции.

Машинска Визија: За детекција на фигурите на шаховската табла, се користат алгоритми од областа на машинската визија. Ова ви овозможува да ги идентификувате и класификувате шаховските фигури на сликата.

Python: Како јазик за развој, користите Python, кој е популарен и лесен за учење. Python ви овозможува да го интегрирате различни технологии и да изработите комплексни операции на бекендот.

Webpack: За компресирање на фронтенд решението, користите Webpack. Webpack го олеснува управувањето со зависности и генерирањето на оптимизирани фронтенд фајлови.

React Router: За управување со рутите на фронтендот, користите React Router. Ова ви овозможува да го контролирате прелистувањето помеѓу различните делови на вашата апликација.

Pipenv: За менаџирање со Python библиотеките и средината, користите Pipenv. Ова ви овозможува да ефикасно управувате со зависностите и да го одржувате околината на вашиот проект чиста и организирана.

Babel: За компилација на JavaScript кодот, користите Babel. Babel го преведува вашиот JavaScript код во формат кој е разбирлив за различни веб прелистувачи, што ги олеснува интерпретацијата на фронтендот во различни околини.

## Бекенд

### Менаџирање со пајтон пакети

pipenv е мојот избор за менаџирање на зависности во Python проектот поради изолацијата на зависностите, точното одредување на верзии, лесната репродукција на околината и јасната структура за користење. Ова обезбедува стабилно и конзистентно развојно окружување на целиот проект. Со користење на Pipfile и Pipfile.lock, се чуваат точни информации за зависностите, што спречува не совпаѓање на верзии. Интеграцијата со инструменти за тестирање како pytest го прави pipenv уште поедноставен за користење во развојниот процес.

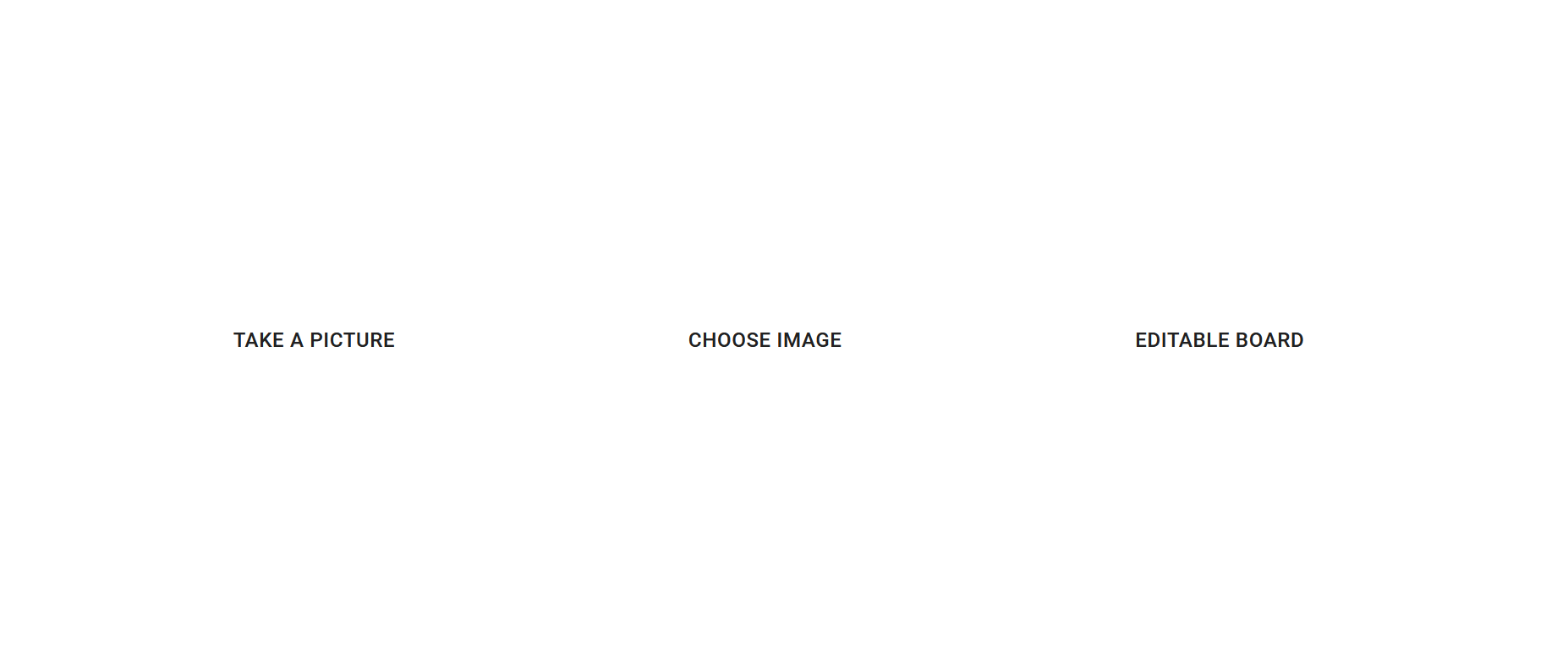
Бекендот е организиран во 3 суб-апликации од Django

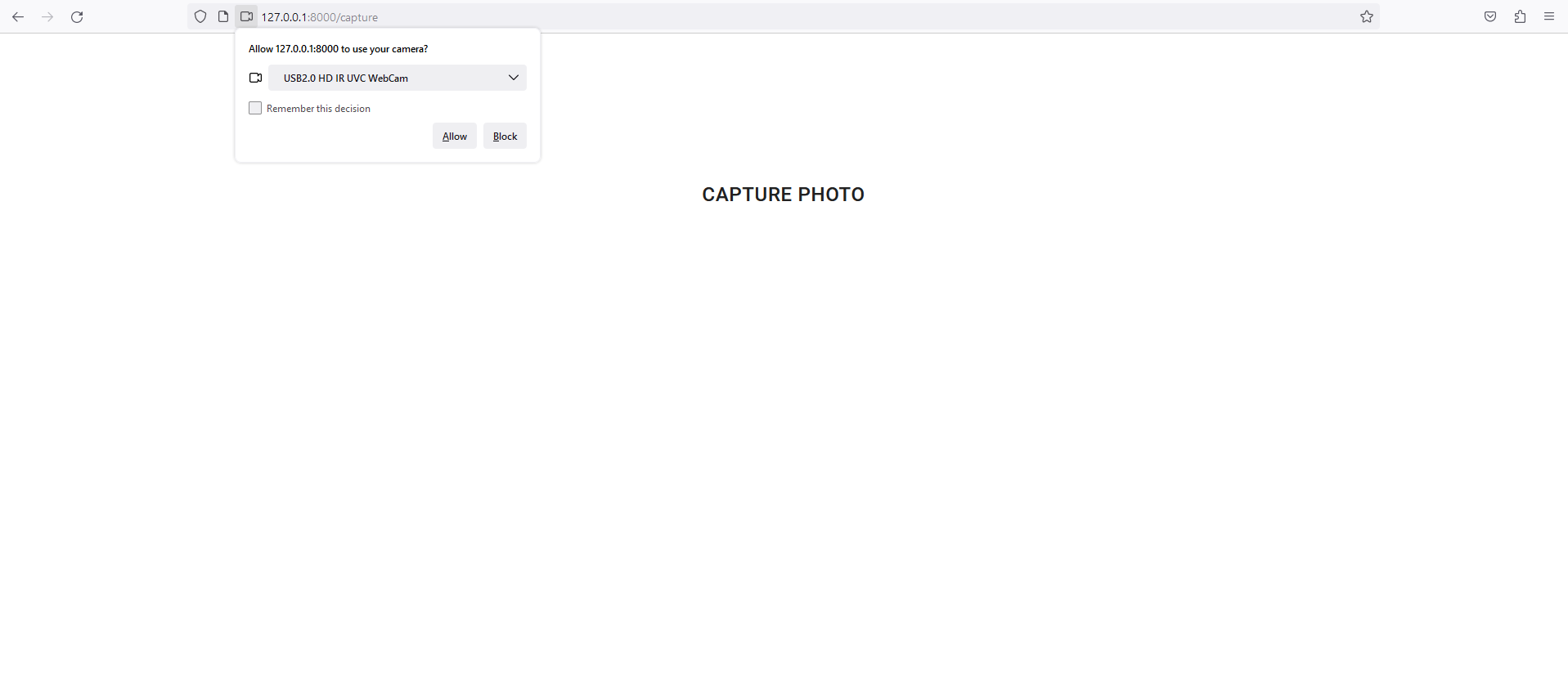
* Chess\_app

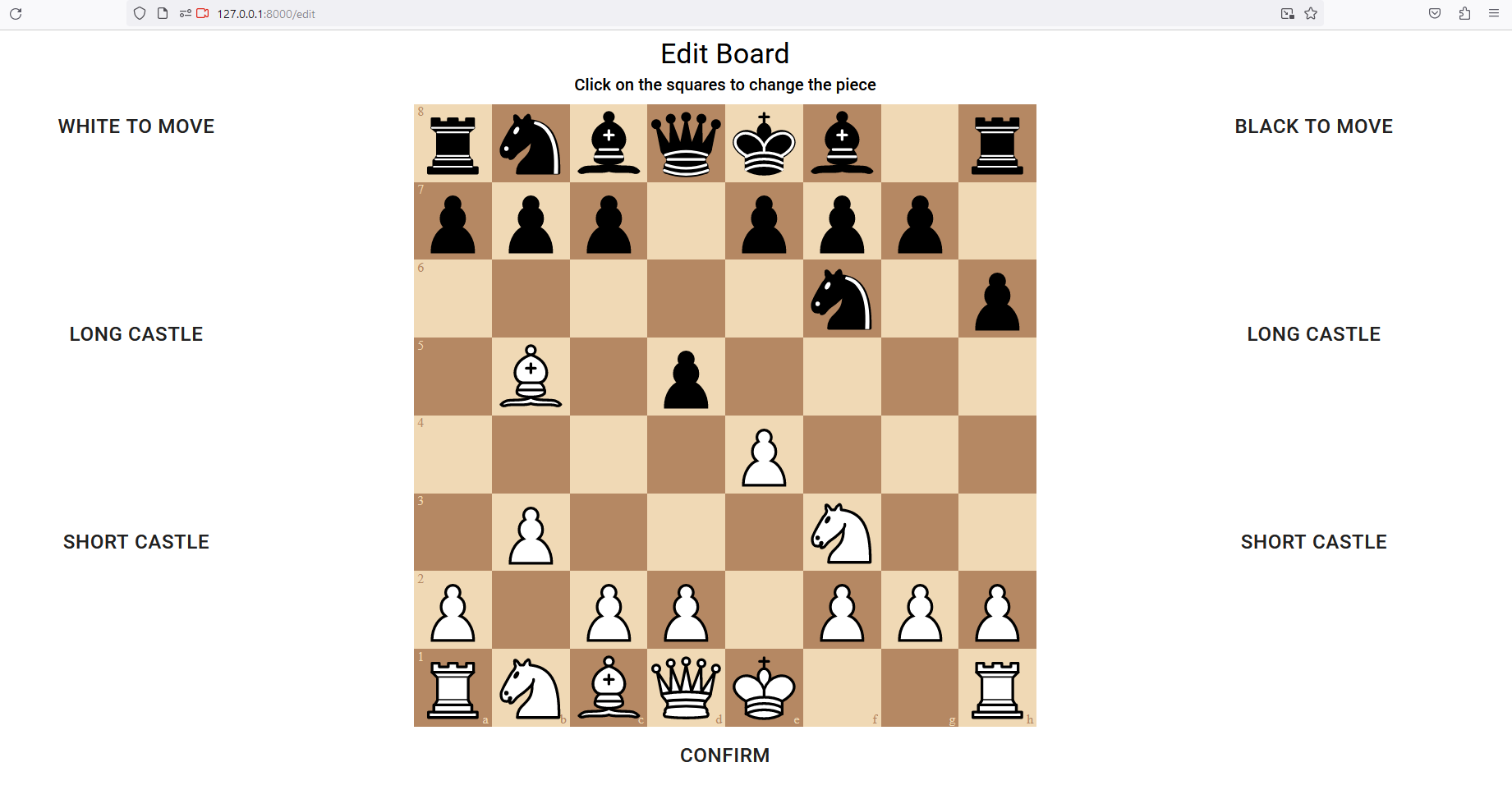
Во овој модул се сите потребни вјуа за да може апликацијата да функциоира, како првиот вју што се корисити за праќање на слика од фронтенд-от на апликацијата до бекендот, оваа слика. Вториот вју се користи за праќање на резултатот

* Api
* Frontend

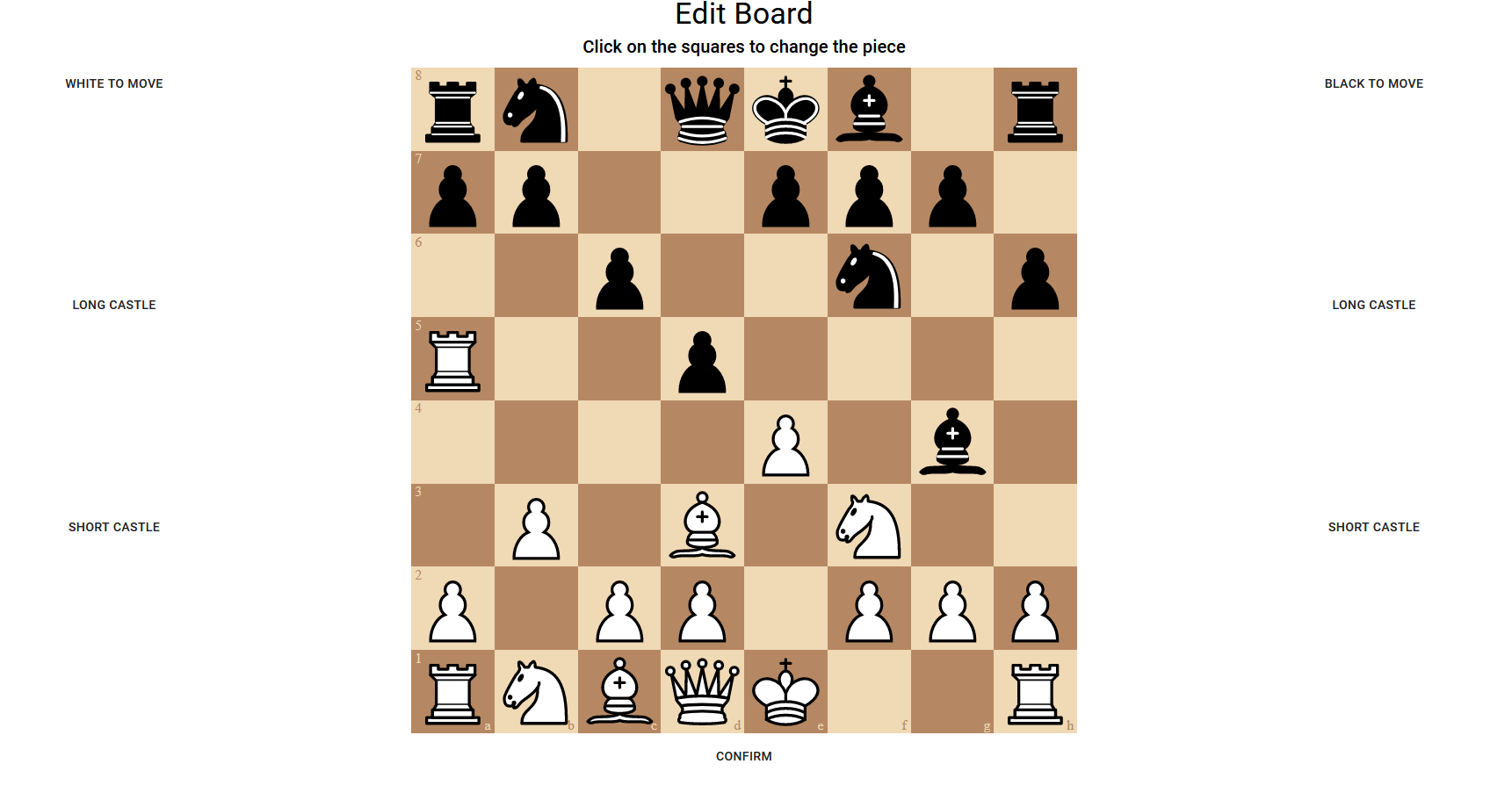
## Фронтенд

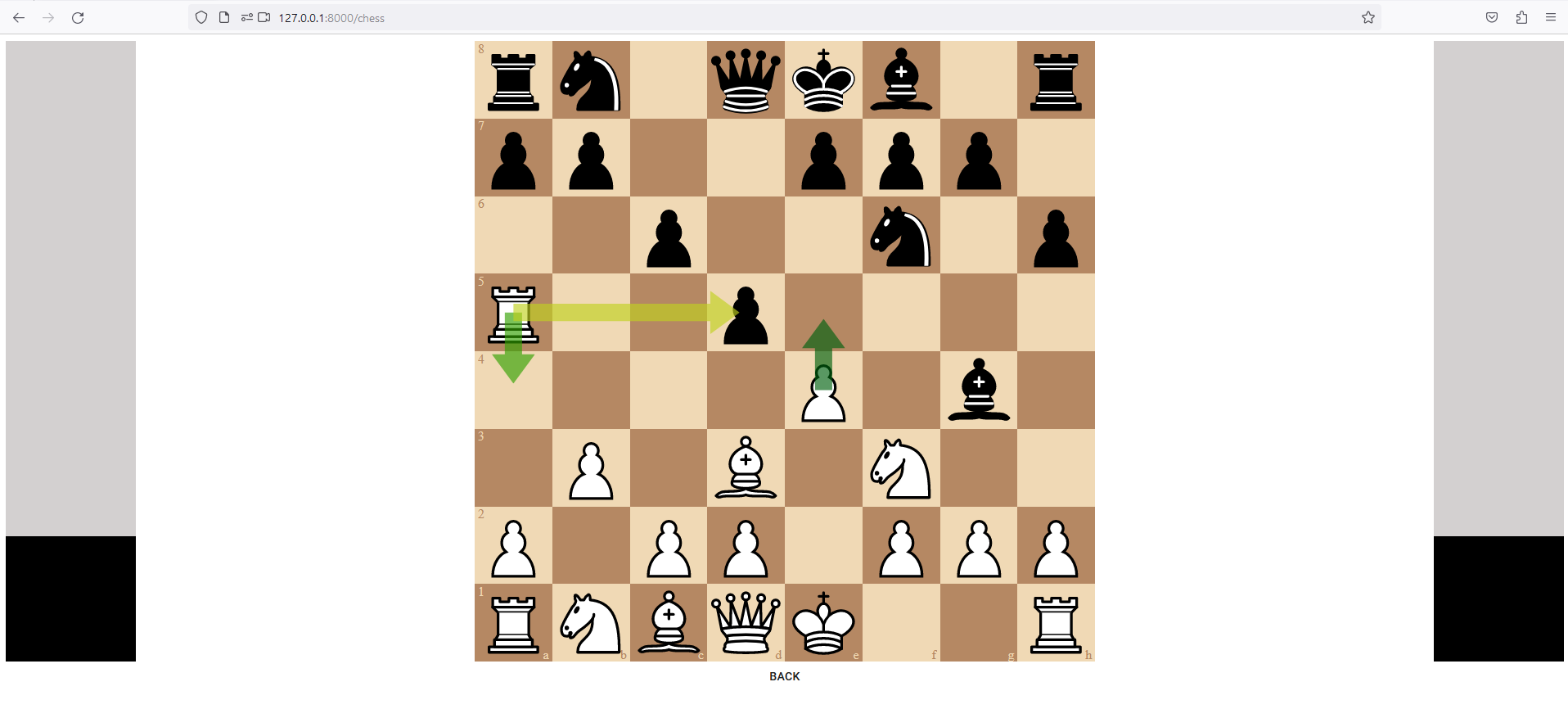
****

****

****

****

****

****

Тука ставаш слики од дизајнот на страната, почетниот workflow diagram што го нацрта и дедално објаснување на компонентите напишани во реакт (ќе пополниш бајаѓи место) со слики се разбира

Почетна страна: при активирање на апликацијата се доведува до почетната страна на апликацијата која има 3 главни и едини копчињаѕ. Тие копчиња служат за внес на слика директно од камера или превземена слика од галерија, а пак 3тото копче носи директно до страната за едитирање на почетна шаховска позиција каде корисникот ќе може мануелно да внесе било каква шаховска позиција и истата да се евалуира.

А пак ако се прати слика, корисникот се носи на страна кое е како лоби (се чека хехе) да завршат сите потребни алгоритми за да вратам ФЕН стринг која ќе се искористи да сетира шаховската табла на страната за едитирање. Потоа корисникот конфигурира уште неколку копчиња за шаховската позиција како: дали двете страни имаат можност да направат рокада на секоја страна посебно и кој играч е на потег. При внес на овие информации и притискање на копчето, корисникот се носи на главната страна која опишува кој играч е во водство и кои се 3те најдобри потега според stockfish 16 шаховскиот алгоритам со длабочина од 15 од дрвото за пребарување на најдобар потег.

## Податочно множество

Податочното множество кое е искористено за тренирање на моделот за детекција на шаховски фигури е превземено од roboflow страната, потоа со додавање на слики од мојите шаховската фигури и лабелирање на истите,

A table with numbers and letters

Description automatically generated

## Машински модели

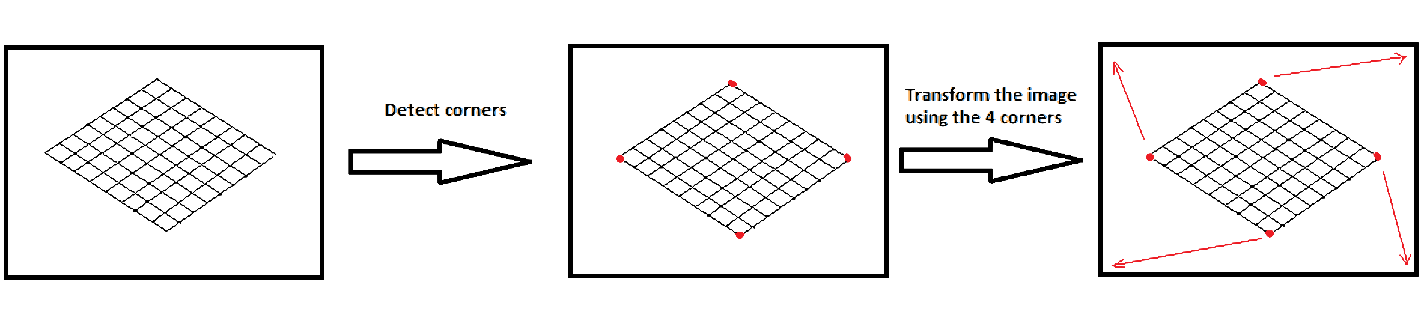
### Модел за препознавање шаховска табла

За ова искористено е готов модел превземен од github страната, и со помош на пајтон библиотеката OpenCV прелагоден е моделот за зголемена ефикасност на детекција на шаховска табла.

Самата природа на алгоритмот доведува до зголемени преформаси ако аголот на сликата е од 90 степени поставена над шаховската табла. Бидејќи при делење на шаховската табла на 64 делови, се доведува до намалување на оверлап помеѓу полињата и со тоа се зголемува ефикасноста на моделот за препознавање фигури

Дигитализација на физичката шаховска табла се извршува со помош на два модели.

Првиот модел е задолжен за детекција на четирите ќошеви на шаховската табла, така што при детекција на 4 ќошеви се извршува низа на трансформации која доведува шаховската табла целосно да ја зафаќа сликата така што потоа со едноставно делење на сликата на 8 еднакви делови по x и y оската се добиваат 64 слики. Со првична претпоставка дека белиот играч се наоѓа на долната страна од сликата, можеме да ги лабелирање сите слики со соодветниот квадрат каде што се наоѓа почнувајќи од А1 квадратот од долната-лева страна на сликата.

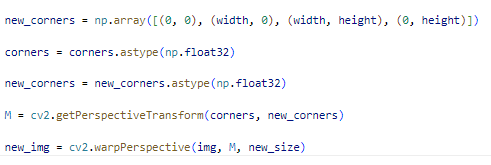


Пред да се детектираат кошевите на шаховската табла, мора сликата да се трансформира да има должина 512 и ширина 384 за компатибилност со моделот. Но со тоа, координатите на детектираните кошеви на шаховската табла се одвиваат од опсег на квалитетот од трансформираната слика, но бидејќи не сакаме да се изгуби квалитетот на сликите на поединечните слики се прави следнава трансформација на детектираните кошеви:

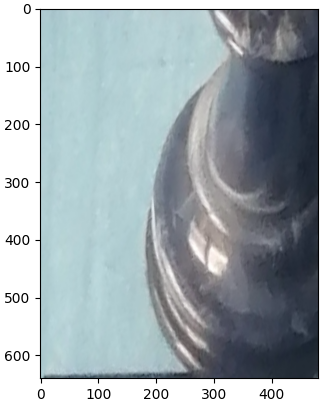
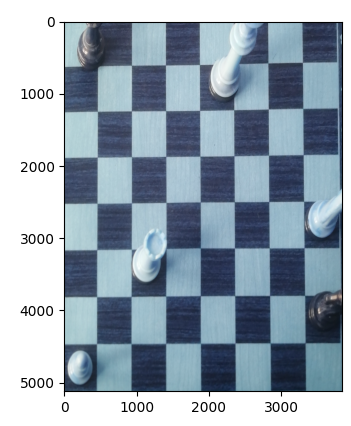


Со оваа трансформација се добиваат координатите на шаховската табла во релација со висината и ширината на оригиналната слика.

Потоа се прави транформацијата на сл горе користејќи го следниов код со помош на OpenCV пајтон библиотеката:

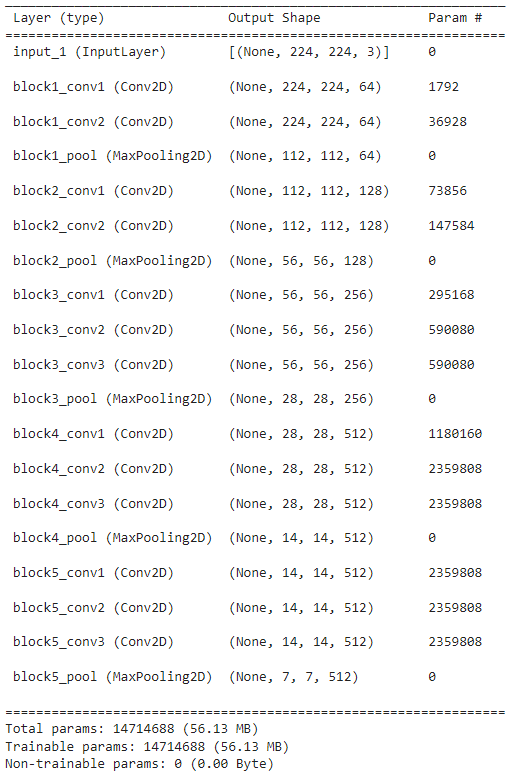


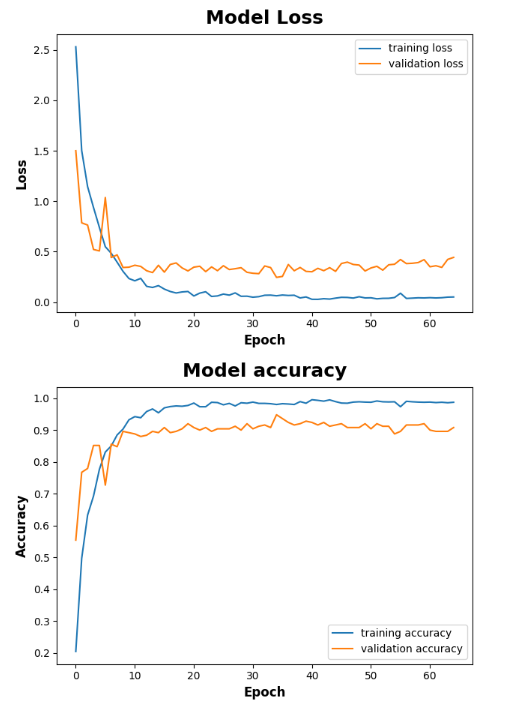
Пример за горенаведената трансформација:

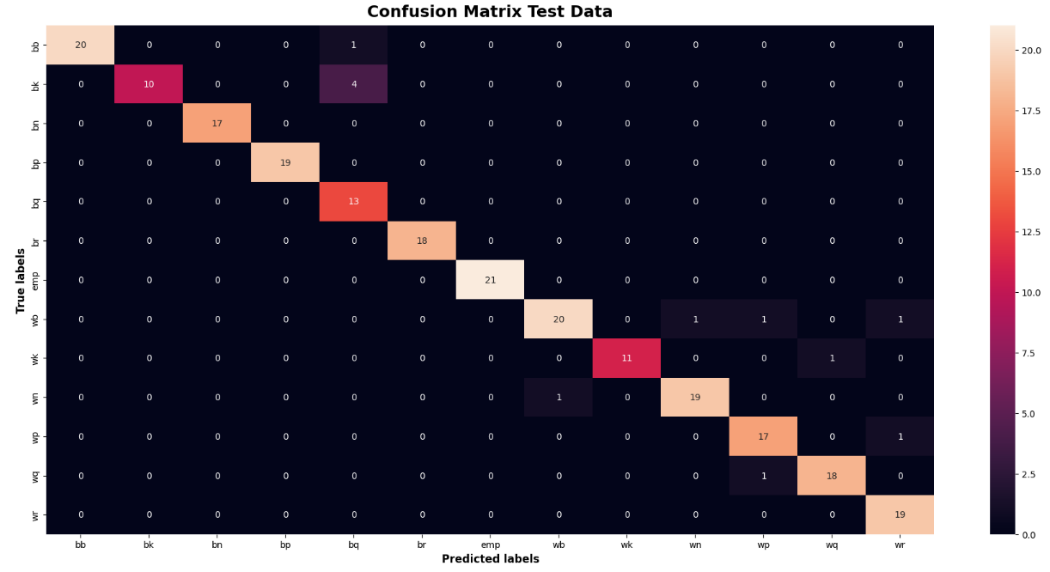


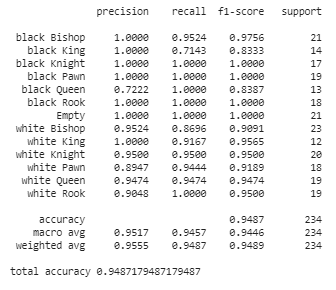
По добивање на координатите на сите 64 полиња од шаховската табла, истите можат да се искористат со трансформацијата за да се добие низа од 64 слики кои кореспондираат со полето од шаховската табла. Со ова лесно ќе може да се добие FEN стрингот која ќе се прати како response од request-от на пратената слика.

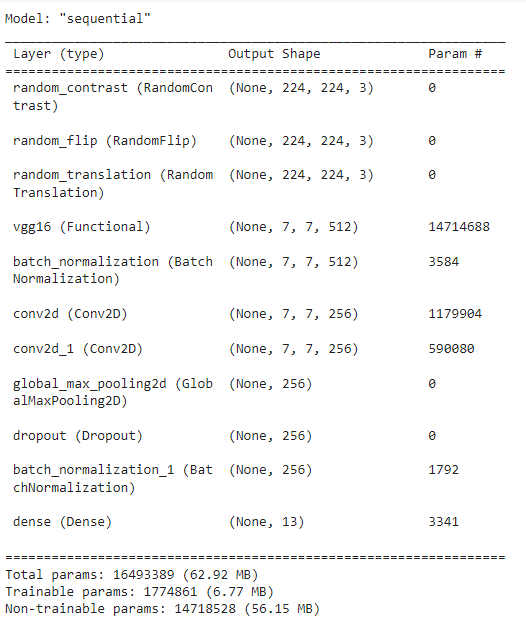
### Модели за детекција









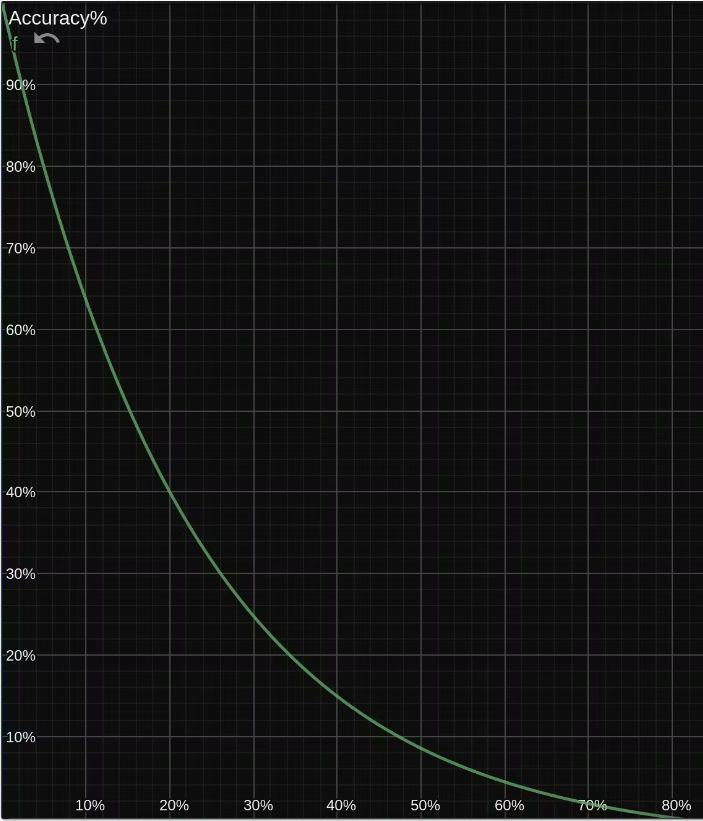


## Евалуација на шаховска позиција

Евалуација на шаховската позиција се остварува преку користење на еден од помоќните шаховски алгоритми наречен stockfish со верзија 16. Во рами на апликацијата се постава целосниот код потребен за stockfish да функционира, и со помош на пајтон библиотеката stockfish се интегрира во бекендот. За одредување на 3-те најдобри потега од дадена шаховска позиција, потребно е само да пратиме FEN. По евалуација на

A graph with a blue line

Description automatically generated



Опис на целосна веб апликација

## Проширување

Во рамки на целосната апликација има можности за проширување во секој аспект по апликацијата. Како една од поинтересните делови за проширување на апликацијата но и значително потежок за имплементација е додавање начин корисниците да можат да внесат одреден број на слики од нивната уникатна шаховска табла со шаховските фигури, која при процес наречен transfer learning ќе го надгради моделот со спесифичните шаховски фигури и ќе ја зголеми прецизноста на моделот. Секако со оваа функционалност би било конвенционално да се направи функционалност за логирање на корисниците за можност на зачувување на уникатниот модел.

Друг аспект на проширување на апликацијата е додавање имиња и објаснувања на познати шаховски отварања. Исто така би било корисно и интересно, во случај на одредена шаховска позиција да има секвенција на шаховски потези која доведува до форсиран шах мат, истите потега да се покажат на корисникот.

Додавање на функционалност во случај кралот да не е на стартната позиција, копчињата за рокада од соодветната страна да бидат автоматски селектирани, поточно деселектирани

## Заклучок

## Референци