KARIITHI ANNE WANJIKU

kariithiwanjiku3@gmail.com

Malaria Classification model

Task TensorFlow is an open-source machine learning framework developed by Google for the sole purpose of building and training machine learning models.

One of the datasets under the Tensorflow Image libraries is a malaria dataset. Using this dataset, we would like you to create a model which is able to classify whether a blood smear is uninfected or parasitized.

Abstract TensorFlow Datasets has many datasets that can be loaded and be used to learn more about image classification and various computer vision machine learning pipelines. One of the datasets hosted here is the "Malaria Dataset".

The Malaria dataset contains a total of 27,558 cell images with equal instances of parasitized and uninfected cells from the thin blood smear slide images of segmented cells.

Remark

This dataset consists of two directories, one is the parasitized cell images and the other is the uninfected cell images. Each of these directories has a number of images and labels. The label denotes the class to which that image belongs.

To Do

- Create a model to classify the different blood smears into eith parasitized or uninfected
- Transform the best model into a tflite model
- Load it into a mobile application
- Use the model to correctly classify the different images selected from the image list

Import Libraries

In [1]:
```
!pip install tensorflow
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: tensorflow in c:\users\admin\appdata\roaming\python
\python311\site-packages (2.12.0rc0)
Requirement already satisfied: tensorflow-intel==2.12.0-rc0 in c:\users\admin\appd
ata\roaming\python\python311\site-packages (from tensorflow) (2.12.0rc0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\admin\appdata\roaming\py
thon\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (1.4.
0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\admin\appdata\roaming
\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow)
(1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in c:\users\admin\appdata\roaming
\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (2
3.1.21)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in c:\users\admin\appdata\roami
ng\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow)
(0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\admin\appdata\roami
ng\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow)
(0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\admin\appdata\roaming\pytho
n\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (3.8.0)
Requirement already satisfied: jax>=0.3.15 in c:\users\admin\appdata\roaming\pytho
n\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (0.4.4)
Requirement already satisfied: libclang>=13.0.0 in c:\users\admin\appdata\roaming
\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (1
5.0.6.1)
Requirement already satisfied: numpy<1.24,>=1.22 in c:\users\admin\appdata\roaming
\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow)
(1.23.5)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\admin\appdata\roaming
\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow)
(3.3.0)
Requirement already satisfied: packaging in c:\users\admin\appdata\roaming\python
\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (23.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.
4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\users\admin\appdata\roaming\python\python311\s
ite-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (4.25.3)
Requirement already satisfied: setuptools in c:\program files\python311\lib\site-p
ackages (from tensorflow-intel==2.12.0-rc0->tensorflow) (65.5.0)
Requirement already satisfied: six>=1.12.0 in c:\users\admin\appdata\roaming\pytho
n\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\admin\appdata\roaming
\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow)
(2.2.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\admin\appdata
\roaming\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensor
flow) (4.8.0)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\admin\appdata\roaming\pyt
hon\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow) (1.14.
1)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\admin\appdata\roami
ng\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflow)
(1.51.1)
Requirement already satisfied: tensorboard<2.13,>=2.12 in c:\users\admin\appdata\r
oaming\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorfl
```

```
ow) (2.12.0)
Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0rc0 in c:\users\a
dmin\appdata\roaming\python\python311\site-packages (from tensorflow-intel==2.12.0
-rc0->tensorflow) (2.12.0rc0)
Requirement already satisfied: keras<2.13,>=2.12.0rc0 in c:\users\admin\appdata\ro
aming\python\python311\site-packages (from tensorflow-intel==2.12.0-rc0->tensorflo
w) (2.12.0rc0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in c:\users\ad
min\appdata\roaming\python\python311\site-packages (from tensorflow-intel==2.12.0-
rc0->tensorflow) (0.30.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\admin\appdata\roamin
g\python\python311\site-packages (from astunparse>=1.6.0->tensorflow-intel==2.12.0
-rc0->tensorflow) (0.38.4)
Requirement already satisfied: scipy>=1.5 in c:\users\admin\appdata\roaming\python
\python311\site-packages (from jax>=0.3.15->tensorflow-intel==2.12.0-rc0->tensorfl
ow) (1.10.0)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\admin\appdata\roa
ming\python\python311\site-packages (from tensorboard<2.13,>=2.12->tensorflow-inte
l==2.12.0-rc0->tensorflow) (2.16.1)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in c:\users\admin
\appdata\roaming\python\python311\site-packages (from tensorboard<2.13,>=2.12->ten
sorflow-intel==2.12.0-rc0->tensorflow) (0.4.6)
Requirement already satisfied: markdown>=2.6.8 in c:\users\admin\appdata\roaming\p
ython\python311\site-packages (from tensorboard<2.13,>=2.12->tensorflow-intel==2.1
2.0-rc0->tensorflow) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\admin\appdata\roami
ng\python\python311\site-packages (from tensorboard<2.13,>=2.12->tensorflow-intel=
=2.12.0-rc0->tensorflow) (2.28.2)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in c:\users\a
dmin\appdata\roaming\python\python311\site-packages (from tensorboard<2.13,>=2.12-
>tensorflow-intel==2.12.0-rc0->tensorflow) (0.7.0)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in c:\users\admin\app
data\roaming\python\python311\site-packages (from tensorboard<2.13,>=2.12->tensorf
low-intel==2.12.0-rc0->tensorflow) (1.8.1)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\admin\appdata\roaming\p
ython\python311\site-packages (from tensorboard<2.13,>=2.12->tensorflow-intel==2.1
2.0-rc0->tensorflow) (3.0.0)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\admin\appdata\ro
aming\python\python311\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.1
3,>=2.12->tensorflow-intel==2.12.0-rc0->tensorflow) (5.3.0)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\admin\appdata\roa
ming\python\python311\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.13,
>=2.12->tensorflow-intel==2.12.0-rc0->tensorflow) (0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\admin\appdata\roaming\pyt
hon\python311\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->
tensorflow-intel==2.12.0-rc0->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\admin\appdata
\roaming\python\python311\site-packages (from google-auth-oauthlib<0.5,>=0.4.1->te
nsorboard<2.13,>=2.12->tensorflow-intel==2.12.0-rc0->tensorflow) (1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\appdata
\roaming\python\python311\site-packages (from requests<3,>=2.21.0->tensorboard<2.1
3,>=2.12->tensorflow-intel==2.12.0-rc0->tensorflow) (3.0.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\appdata\roaming\pyth
on\python311\site-packages (from requests<3,>=2.21.0->tensorboard<2.13,>=2.12->ten
sorflow-intel==2.12.0-rc0->tensorflow) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\admin\appdata\roa
```

```
ming\python\python311\site-packages (from requests<3,>=2.21.0->tensorboard<2.13,>=
2.12->tensorflow-intel==2.12.0-rc0->tensorflow) (1.26.14)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\appdata\roamin
g\python\python311\site-packages (from requests<3,>=2.21.0->tensorboard<2.13,>=2.1
2->tensorflow-intel==2.12.0-rc0->tensorflow) (2022.12.7)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\admin\appdata\roaming
\python\python311\site-packages (from werkzeug>=1.0.1->tensorboard<2.13,>=2.12->te
nsorflow-intel==2.12.0-rc0->tensorflow) (2.1.2)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\admin\appdata\roam
ing\python\python311\site-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.
6.3->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0-rc0->tensorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\admin\appdata\roaming\p
ython\python311\site-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib
<0.5,>=0.4.1->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0-rc0->tensorflow)
(3.2.2)
DEPRECATION: Loading egg at c:\program files\python311\lib\site-packages\vboxapi-
1.0-py3.11.egg is deprecated. pip 24.3 will enforce this behaviour change. A possi
ble replacement is to use pip for package installation.. Discussion can be found a
t https://github.com/pypa/pip/issues/12330
```

In [2]:
```
!pip install tensorflow-datasets
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: tensorflow-datasets in c:\users\admin\appdata\roami
ng\python\python311\site-packages (4.9.4)
Requirement already satisfied: absl-py in c:\users\admin\appdata\roaming\python\py
thon311\site-packages (from tensorflow-datasets) (1.4.0)
Requirement already satisfied: click in c:\users\admin\appdata\roaming\python\pyth
on311\site-packages (from tensorflow-datasets) (8.1.7)
Requirement already satisfied: dm-tree in c:\users\admin\appdata\roaming\python\py
thon311\site-packages (from tensorflow-datasets) (0.1.8)
Requirement already satisfied: etils>=0.9.0 in c:\users\admin\appdata\roaming\pyth
on\python311\site-packages (from etils[enp,epath,etree]>=0.9.0->tensorflow-dataset
s) (1.8.0)
Requirement already satisfied: numpy in c:\users\admin\appdata\roaming\python\pyth
on311\site-packages (from tensorflow-datasets) (1.23.5)
Requirement already satisfied: promise in c:\users\admin\appdata\roaming\python\py
thon311\site-packages (from tensorflow-datasets) (2.3)
Requirement already satisfied: protobuf>=3.20 in c:\users\admin\appdata\roaming\py
thon\python311\site-packages (from tensorflow-datasets) (4.25.3)
Requirement already satisfied: psutil in c:\users\admin\appdata\roaming\python\pyt
hon311\site-packages (from tensorflow-datasets) (5.9.4)
Requirement already satisfied: requests>=2.19.0 in c:\users\admin\appdata\roaming
\python\python311\site-packages (from tensorflow-datasets) (2.28.2)
Requirement already satisfied: tensorflow-metadata in c:\users\admin\appdata\roami
ng\python\python311\site-packages (from tensorflow-datasets) (1.15.0)
Requirement already satisfied: termcolor in c:\users\admin\appdata\roaming\python
\python311\site-packages (from tensorflow-datasets) (2.2.0)
Requirement already satisfied: toml in c:\users\admin\appdata\roaming\python\pytho
n311\site-packages (from tensorflow-datasets) (0.10.2)
Requirement already satisfied: tqdm in c:\users\admin\appdata\roaming\python\pytho
n311\site-packages (from tensorflow-datasets) (4.66.1)
Requirement already satisfied: wrapt in c:\users\admin\appdata\roaming\python\pyth
on311\site-packages (from tensorflow-datasets) (1.14.1)
Requirement already satisfied: fsspec in c:\users\admin\appdata\roaming\python\pyt
hon311\site-packages (from etils[enp,epath,etree]>=0.9.0->tensorflow-datasets) (20
23.10.0)
Requirement already satisfied: importlib_resources in c:\users\admin\appdata\roami
ng\python\python311\site-packages (from etils[enp,epath,etree]>=0.9.0->tensorflow-
datasets) (6.4.0)
Requirement already satisfied: typing_extensions in c:\users\admin\appdata\roaming
\python\python311\site-packages (from etils[enp,epath,etree]>=0.9.0->tensorflow-da
tasets) (4.8.0)
Requirement already satisfied: zipp in c:\users\admin\appdata\roaming\python\pytho
n311\site-packages (from etils[enp,epath,etree]>=0.9.0->tensorflow-datasets) (3.1
8.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\appdata
\roaming\python\python311\site-packages (from requests>=2.19.0->tensorflow-dataset
s) (3.0.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\appdata\roaming\pyth
on\python311\site-packages (from requests>=2.19.0->tensorflow-datasets) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\admin\appdata\roa
ming\python\python311\site-packages (from requests>=2.19.0->tensorflow-datasets)
(1.26.14)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\appdata\roamin
g\python\python311\site-packages (from requests>=2.19.0->tensorflow-datasets) (202
2.12.7)
Requirement already satisfied: colorama in c:\users\admin\appdata\roaming\python\p
```

```
ython311\site-packages (from click->tensorflow-datasets) (0.4.6)
Requirement already satisfied: six in c:\users\admin\appdata\roaming\python\python
311\site-packages (from promise->tensorflow-datasets) (1.16.0)
Requirement already satisfied: googleapis-common-protos<2,>=1.56.4 in c:\users\adm
in\appdata\roaming\python\python311\site-packages (from tensorflow-metadata->tenso
rflow-datasets) (1.63.0)
```

```
DEPRECATION: Loading egg at c:\program files\python311\lib\site-packages\vboxapi-
1.0-py3.11.egg is deprecated. pip 24.3 will enforce this behaviour change. A possi
ble replacement is to use pip for package installation.. Discussion can be found a
t https://github.com/pypa/pip/issues/12330
```

In [3]:
```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
import tensorflow_datasets as tfds
```

In [4]:
```
# List all the available datasets and load them into the notebook then select the m
tfds.list_builders()
```

Out[4]:
```
['abstract_reasoning',
 'accentdb',
 'aeslc',
 'aflw2k3d',
 'ag_news_subset',
 'ai2_arc',
 'ai2_arc_with_ir',
 'amazon_us_reviews',
 'anli',
 'answer_equivalence',
 'arc',
 'asqa',
 'asset',
 'assin2',
 'asu_table_top_converted_externally_to_rlds',
 'austin_buds_dataset_converted_externally_to_rlds',
 'austin_sailor_dataset_converted_externally_to_rlds',
 'austin_sirius_dataset_converted_externally_to_rlds',
 'bair_robot_pushing_small',
 'bc_z',
 'bccd',
 'beans',
 'bee_dataset',
 'beir',
 'berkeley_autolab_ur5',
 'berkeley_cable_routing',
 'berkeley_fanuc_manipulation',
 'berkeley_gnm_cory_hall',
 'berkeley_gnm_recon',
 'berkeley_gnm_sac_son',
 'berkeley_mvp_converted_externally_to_rlds',
 'berkeley_rpt_converted_externally_to_rlds',
 'big_patent',
 'bigearthnet',
 'billsum',
 'binarized_mnist',
 'binary_alpha_digits',
 'ble_wind_field',
 'blimp',
 'booksum',
 'bool_q',
 'bot_adversarial_dialogue',
 'bridge',
 'bucc',
 'c4',
 'c4_wsrs',
 'caltech101',
 'caltech_birds2010',
 'caltech_birds2011',
 'cardiotox',
 'cars196',
 'cassava',
 'cats_vs_dogs',
 'celeb_a',
 'celeb_a_hq',
 'cfq',
```

```
                'cherry_blossoms',
                'chexpert',
                'cifar10',
                'cifar100',
                'cifar100_n',
                'cifar10_1',
                'cifar10_corrupted',
                'cifar10_h',
                'cifar10_n',
                'citrus_leaves',
                'cityscapes',
                'civil_comments',
                'clevr',
                'clic',
                'clinc_oos',
                'cmaterdb',
                'cmu_franka_exploration_dataset_converted_externally_to_rlds',
                'cmu_play_fusion',
                'cmu_stretch',
                'cnn_dailymail',
                'coco',
                'coco_captions',
                'coil100',
                'colorectal_histology',
                'colorectal_histology_large',
                'columbia_cairlab_pusht_real',
                'common_voice',
                'conll2002',
                'conll2003',
                'controlled_noisy_web_labels',
                'coqa',
                'corr2cause',
                'cos_e',
                'cosmos_qa',
                'covid19',
                'covid19sum',
                'crema_d',
                'criteo',
                'cs_restaurants',
                'curated_breast_imaging_ddsm',
                'cycle_gan',
                'd4rl_adroit_door',
                'd4rl_adroit_hammer',
                'd4rl_adroit_pen',
                'd4rl_adroit_relocate',
                'd4rl_antmaze',
                'd4rl_mujoco_ant',
                'd4rl_mujoco_halfcheetah',
                'd4rl_mujoco_hopper',
                'd4rl_mujoco_walker2d',
                'dart',
                'databricks_dolly',
                'davis',
                'deep1b',
                'deep_weeds',
                'definite_pronoun_resolution',
```

```
'dementiabank',
'diabetic_retinopathy_detection',
'diamonds',
'div2k',
'dlr_edan_shared_control_converted_externally_to_rlds',
'dlr_sara_grid_clamp_converted_externally_to_rlds',
'dlr_sara_pour_converted_externally_to_rlds',
'dmlab',
'doc_nli',
'dolphin_number_word',
'domainnet',
'downsampled_imagenet',
'drop',
'dsprites',
'dtd',
'duke_ultrasound',
'e2e_cleaned',
'efron_morris75',
'emnist',
'eraser_multi_rc',
'esnli',
'eth_agent_affordances',
'eurosat',
'fashion_mnist',
'flic',
'flores',
'food101',
'forest_fires',
'fractal20220817_data',
'fuss',
'gap',
'geirhos_conflict_stimuli',
'gem',
'genomics_ood',
'german_credit_numeric',
'gigaword',
'glove100_angular',
'glue',
'goemotions',
'gov_report',
'gpt3',
'gref',
'groove',
'grounded_scan',
'gsm8k',
'gtzan',
'gtzan_music_speech',
'hellaswag',
'higgs',
'hillstrom',
'horses_or_humans',
'howell',
'i_naturalist2017',
'i_naturalist2018',
'i_naturalist2021',
'iamlab_cmu_pickup_insert_converted_externally_to_rlds',
```

```
'imagenet2012',
'imagenet2012_corrupted',
'imagenet2012_fewshot',
'imagenet2012_multilabel',
'imagenet2012_real',
'imagenet2012_subset',
'imagenet_a',
'imagenet_lt',
'imagenet_pi',
'imagenet_r',
'imagenet_resized',
'imagenet_sketch',
'imagenet_v2',
'imagenette',
'imagewang',
'imdb_reviews',
'imperialcollege_sawyer_wrist_cam',
'irc_disentanglement',
'iris',
'istella',
'jaco_play',
'kaist_nonprehensile_converted_externally_to_rlds',
'kddcup99',
'kitti',
'kmnist',
'kuka',
'laion400m',
'lambada',
'lfw',
'librispeech',
'librispeech_lm',
'libritts',
'ljspeech',
'lm1b',
'locomotion',
'lost_and_found',
'lsun',
'lvis',
'malaria',
'maniskill_dataset_converted_externally_to_rlds',
'math_dataset',
'math_qa',
'mctaco',
'media_sum',
'mlqa',
'mnist',
'mnist_corrupted',
'movie_lens',
'movie_rationales',
'movielens',
'moving_mnist',
'mrqa',
'mslr_web',
'mt_opt',
'mtnt',
'multi_news',
```

```
'multi_nli',
'multi_nli_mismatch',
'natural_instructions',
'natural_questions',
'natural_questions_open',
'newsroom',
'nsynth',
'nyu_depth_v2',
'nyu_door_opening_surprising_effectiveness',
'nyu_franka_play_dataset_converted_externally_to_rlds',
'nyu_rot_dataset_converted_externally_to_rlds',
'ogbg_molpcba',
'omniglot',
'open_images_challenge2019_detection',
'open_images_v4',
'openbookqa',
'opinion_abstracts',
'opinosis',
'opus',
'oxford_flowers102',
'oxford_iiit_pet',
'para_crawl',
'pass',
'patch_camelyon',
'paws_wiki',
'paws_x_wiki',
'penguins',
'pet_finder',
'pg19',
'piqa',
'places365_small',
'placesfull',
'plant_leaves',
'plant_village',
'plantae_k',
'protein_net',
'q_re_cc',
'qa4mre',
'qasc',
'quac',
'quality',
'quickdraw_bitmap',
'race',
'radon',
'real_toxicity_prompts',
'reddit',
'reddit_disentanglement',
'reddit_tifu',
'ref_coco',
'resisc45',
'rlu_atari',
'rlu_atari_checkpoints',
'rlu_atari_checkpoints_ordered',
'rlu_control_suite',
'rlu_dmlab_explore_object_rewards_few',
'rlu_dmlab_explore_object_rewards_many',
```

```
'rlu_dmlab_rooms_select_nonmatching_object',
'rlu_dmlab_rooms_watermaze',
'rlu_dmlab_seekavoid_arena01',
'rlu_locomotion',
'rlu_rwrl',
'robomimic_mg',
'robomimic_mh',
'robomimic_ph',
'robonet',
'robosuite_panda_pick_place_can',
'roboturk',
'rock_paper_scissors',
'rock_you',
's3o4d',
'salient_span_wikipedia',
'samsum',
'savee',
'scan',
'scene_parse150',
'schema_guided_dialogue',
'sci_tail',
'scicite',
'scientific_papers',
'scrolls',
'segment_anything',
'sentiment140',
'shapes3d',
'sift1m',
'simpte',
'siscore',
'smallnorb',
'smartwatch_gestures',
'snli',
'so2sat',
'speech_commands',
'spoken_digit',
'squad',
'squad_question_generation',
'stanford_dogs',
'stanford_hydra_dataset_converted_externally_to_rlds',
'stanford_kuka_multimodal_dataset_converted_externally_to_rlds',
'stanford_mask_vit_converted_externally_to_rlds',
'stanford_online_products',
'stanford_robocook_converted_externally_to_rlds',
'star_cfq',
'starcraft_video',
'stl10',
'story_cloze',
'summscreen',
'sun397',
'super_glue',
'svhn_cropped',
'symmetric_solids',
'taco_play',
'tao',
'tatoeba',
```

```
'ted_hrlr_translate',
'ted_multi_translate',
'tedlium',
'tf_flowers',
'the300w_lp',
'tiny_shakespeare',
'titanic',
'tokyo_u_lsmo_converted_externally_to_rlds',
'toto',
'trec',
'trivia_qa',
'tydi_qa',
'uc_merced',
'ucf101',
'ucsd_kitchen_dataset_converted_externally_to_rlds',
'ucsd_pick_and_place_dataset_converted_externally_to_rlds',
'uiuc_d3field',
'unified_qa',
'universal_dependencies',
'unnatural_instructions',
'usc_cloth_sim_converted_externally_to_rlds',
'user_libri_audio',
'user_libri_text',
'utaustin_mutex',
'utokyo_pr2_opening_fridge_converted_externally_to_rlds',
'utokyo_pr2_tabletop_manipulation_converted_externally_to_rlds',
'utokyo_saytap_converted_externally_to_rlds',
'utokyo_xarm_bimanual_converted_externally_to_rlds',
'utokyo_xarm_pick_and_place_converted_externally_to_rlds',
'vctk',
'viola',
'visual_domain_decathlon',
'voc',
'voxceleb',
'voxforge',
'waymo_open_dataset',
'web_graph',
'web_nlg',
'web_questions',
'webvid',
'wider_face',
'wiki40b',
'wiki_auto',
'wiki_bio',
'wiki_dialog',
'wiki_table_questions',
'wiki_table_text',
'wikiann',
'wikihow',
'wikipedia',
'wikipedia_toxicity_subtypes',
'wine_quality',
'winogrande',
'wit',
'wit_kaggle',
'wmt13_translate',
```

```
 'wmt14_translate',
 'wmt15_translate',
 'wmt16_translate',
 'wmt17_translate',
 'wmt18_translate',
 'wmt19_translate',
 'wmt_t2t_translate',
 'wmt_translate',
 'wordnet',
 'wsc273',
 'xnli',
 'xquad',
 'xsum',
 'xtreme_pawsx',
 'xtreme_pos',
 'xtreme_s',
 'xtreme_xnli',
 'yahoo_ltrc',
 'yelp_polarity_reviews',
 'yes_no',
 'youtube_vis']
```

In [9]:
```
#load the datasets and print the different labels and categories in the dataset

malaria_ds, malaria_info = tfds.load(
     'malaria',
     split = 'train',
     shuffle_files = True, # to increase randomness in a large dataset
     as_supervised = True, # returns a tuple(image, label) instead of a dictionary
     with_info = True
)
```

Parameters

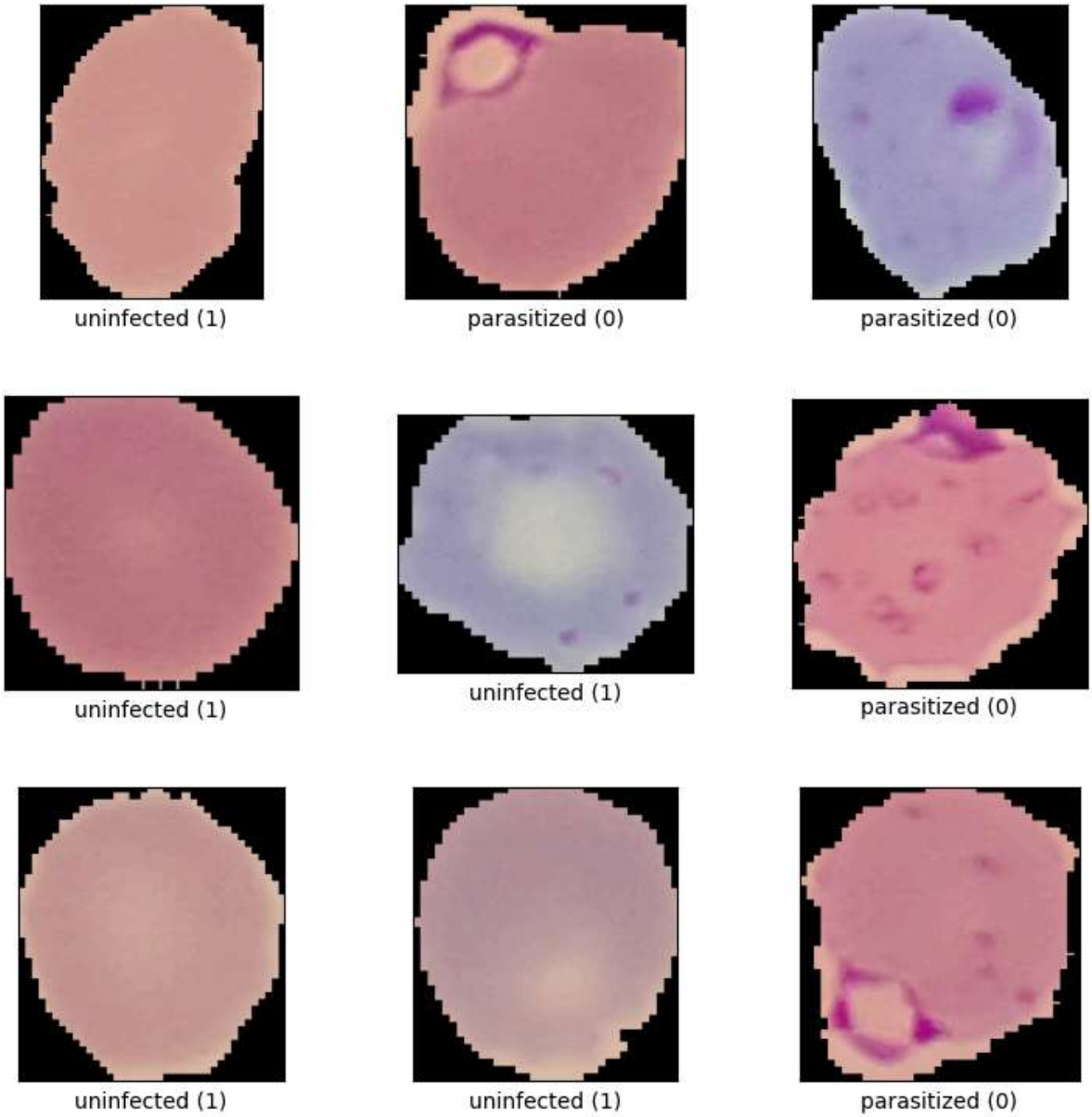
In [10]:
```
categories = 2
no_epochs = 5
batch_size = 32
image_size = [200, 200]
```

Visualize the data and check the number of classes belogining to the dataset

In [7]:
```
print("Categories: " + str(malaria_info.features['label'].num_classes))
print("Class_names: " + str(malaria_info.features['label'].names))
```

```
Categories: 2
Class_names: ['parasitized', 'uninfected']
```

In [8]:
```
# print out some of the images of the cells
malaria_examples = tfds.visualization.show_examples(malaria_ds, malaria_info)
```



Visualize the modified images out of 16 random images in the modified train set

Data Exploration

Into Train, test and validation sets which we will use in our model in the ratio 75:15:10

In [11]:
```
batch_size = 32
image_size = [200, 200]

# split the datasets into train, validation and test sets in the ratio 75:15:10
malaria_train_ds, malaria_val_ds, malaria_test_ds = tfds.load(
    'malaria',
```

```
    split=['train[:75%]', 'train[75%:90%]', 'train[90%:]'],
    shuffle_files=True, as_supervised=True
)
```

In [12]:
```
num_train_mages = tf.data.experimental.cardinality(malaria_train_ds).numpy()
print("Total training images: " + str(num_train_mages ))

num_val_mages = tf.data.experimental.cardinality(malaria_val_ds).numpy()
print("Total validating images: " + str(num_val_mages ))

num_test_mages = tf.data.experimental.cardinality(malaria_test_ds).numpy()
print("Total testing images: " + str(num_test_mages ))
```

```
Total training images: 20668
Total validating images: 4134
Total testing images: 2756
```

Class Distribution

See the number of images per class

In [13]:
```
def get_classes_distribution(data):
    # Create a dictionary for each type of label
    labels = {0: "parasitized", 1: "uninfected"}

    # Initialize an empty dictionary to store label counts
    label_counts = {}

    # Iterate through the dataset and count the occurrences of each label
    for image, label in data:
        label_name = labels[label.numpy()]
        if label_name in label_counts:
            label_counts[label_name] += 1
        else:
            label_counts[label_name] = 1

    # Get total number of samples
    total_samples = len(data)

    # Print the distribution of classes
    for label, count in label_counts.items():
        percent = (count / total_samples) * 100
        print("{:<20s}:   {} or {:.2f}%".format(label, count, percent))

# Call the function with the malaria training dataset
get_classes_distribution(malaria_train_ds)
```

```
uninfected          :   10342 or 50.04%
parasitized         :   10326 or 49.96%
```

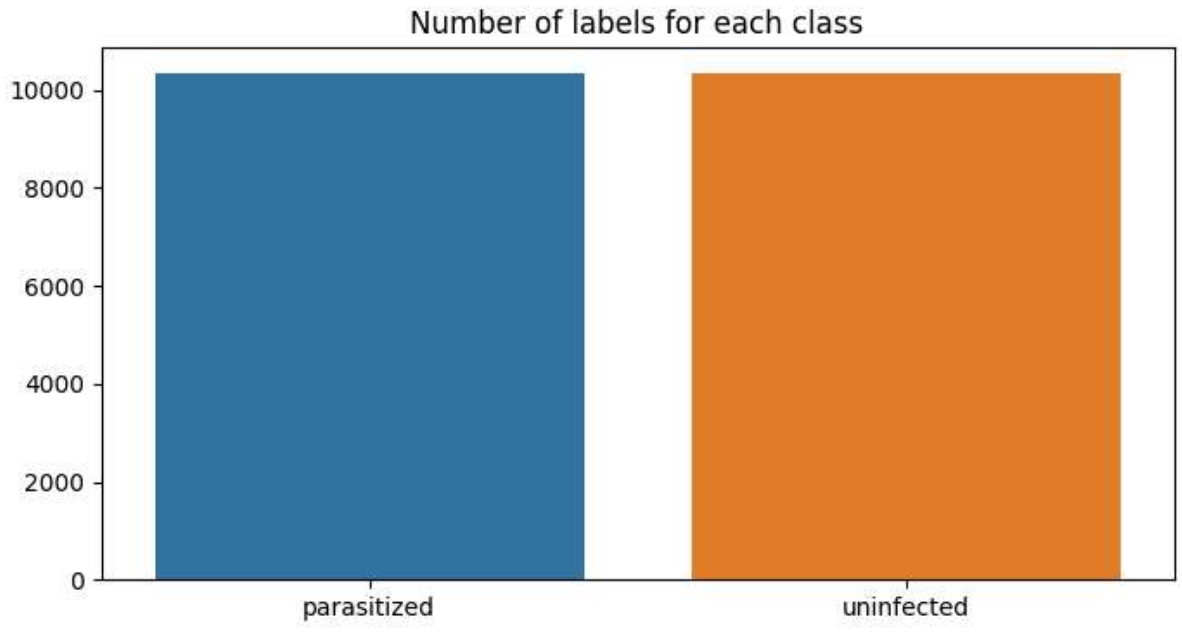
In [12]:
```
# class Distribution in the test data
get_classes_distribution(malaria_test_ds)
```

```
uninfected          :   1385 or 50.25%
parasitized         :   1371 or 49.75%
```

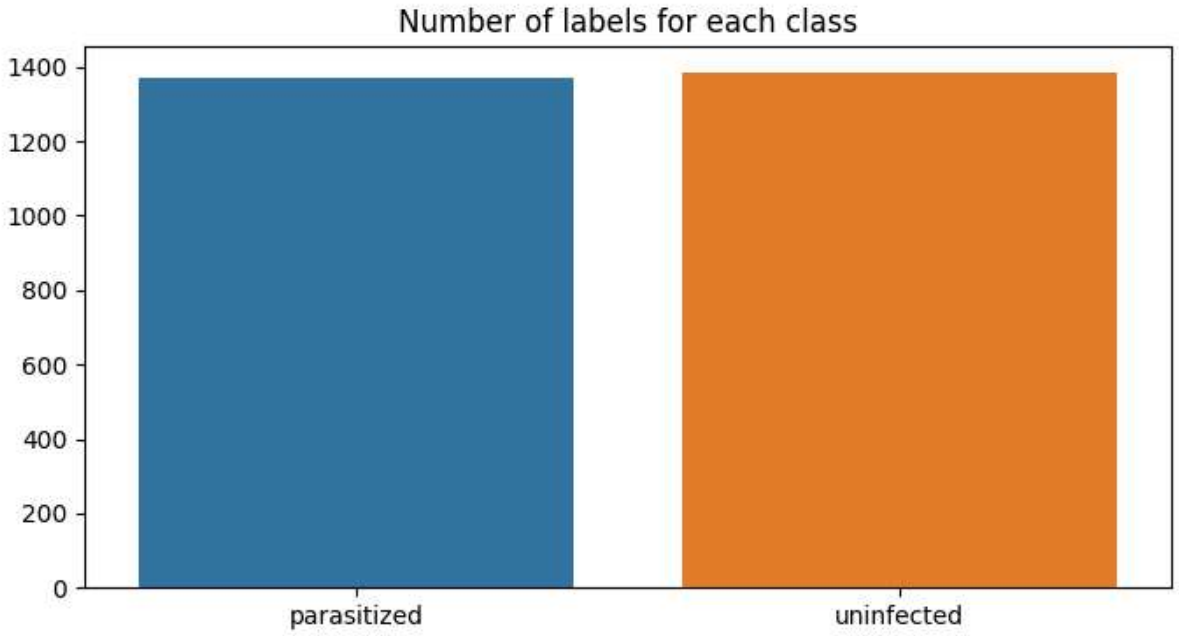
Plot the class Distribution of both test and train data

In [14]:
```
import seaborn as sns
import pandas as pd
```

In [14]:
```
def plot_label_per_class(data):
    # Extract labels and convert them to a list
    labels = [label.numpy() for _, label in data]

    # Define class names
    class_names = ["parasitized", "uninfected"]

    # Calculate label counts
    label_counts = [labels.count(i) for i in range(len(class_names))]

    # Plotting
    f, ax = plt.subplots(1, 1, figsize=(8, 4))  # Reduced figure size
    g = sns.barplot(x=class_names, y=label_counts)
    g.set_title("Number of labels for each class")
    plt.show()

# Call the function with the malaria training dataset
plot_label_per_class(malaria_train_ds)
```



In [15]:
```
plot_label_per_class(malaria_test_ds)
```



Visulaize sample images from the train and test sets

In [16]:
```
for image, label in malaria_train_ds.take(1):
    print("Image shape: ", image.numpy().shape)
    print("Label: ", label.numpy())
```

```
Image shape:  (103, 103, 3)
Label:  0
```

Resize Image to a uniform scale

From the Images shown above on the visualization sector, it is clear that the images are of different sizes. This section will crop big images and padd the small images. Using pad() and .map() functions

In [15]:
```
def convert(image, label):
  image = tf.image.convert_image_dtype(image, tf.float32)
  return image, label

def pad(image,label):
  image,label = convert(image, label)
  image = tf.image.resize_with_crop_or_pad(image, 150, 150)
  return image,label
```

In [16]:
```
# train set modified
mod_train_ds = (
    malaria_train_ds
    .cache()
    .map(pad)
)
```

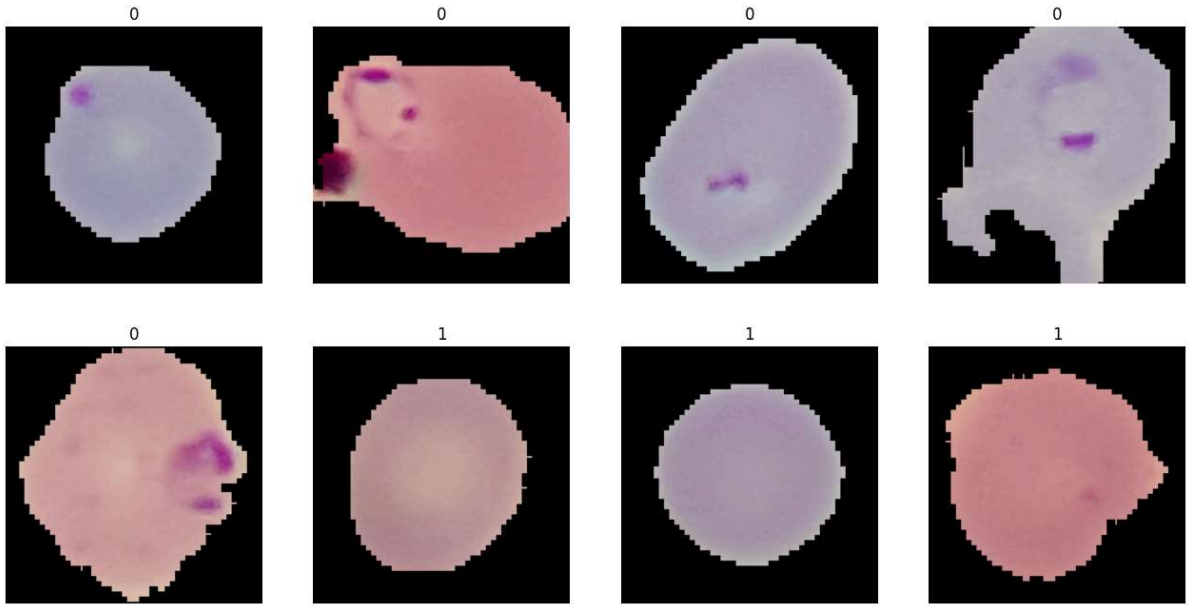
```
# validation set modified
mod_val_ds = (
    malaria_val_ds
    .cache()
    .map(pad)
)

# validation set modified
mod_test_ds = (
    malaria_test_ds
    .cache()
    .map(pad)
)
```

In [17]:
```
for image, label in mod_train_ds.take(1):
    print("Image shape: ", image.numpy().shape)
    print("Label: ", label.numpy())
```

```
Image shape:  (150, 150, 3)
Label:  1
```

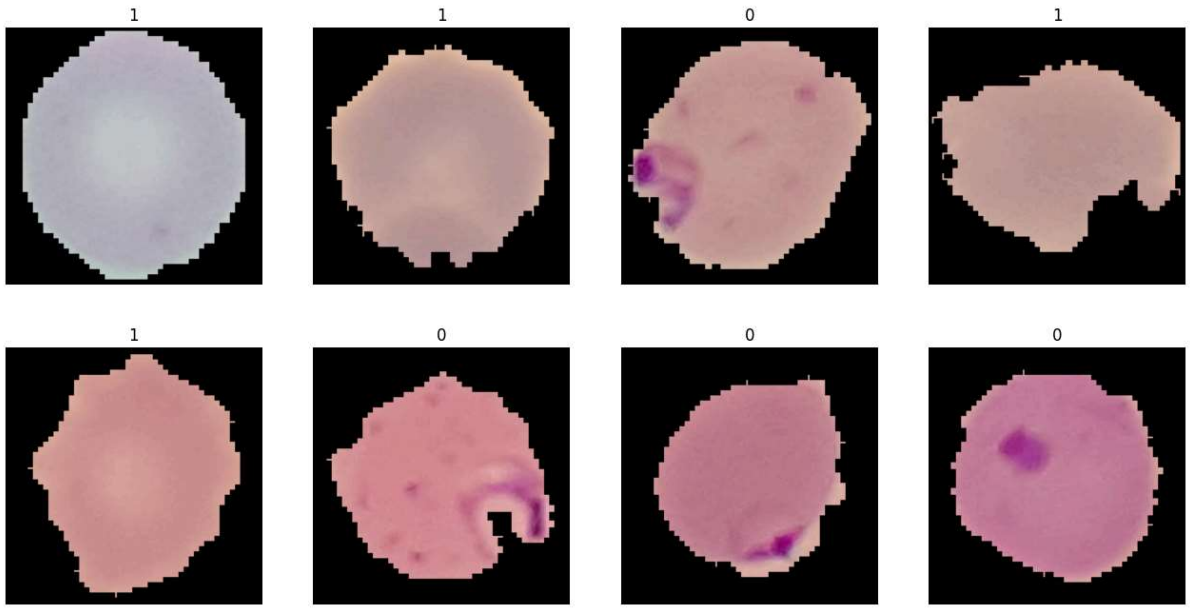
Visualize the images in the train dataset

In [18]:
```
# Sample images and labels from the modified training dataset
sample_images = []
sample_labels = []

for image, label in mod_train_ds.take(8):
    sample_images.append(image.numpy())
    sample_labels.append(label.numpy())

# Plot the sampled images
fig, ax = plt.subplots(2, 4, figsize=(16, 8))
for i, img in enumerate(sample_images):
    ax[i//4, i%4].imshow(img)
    ax[i//4, i%4].axis('off')
    ax[i//4, i%4].set_title(sample_labels[i])
plt.show()
```



Visualize the images in the train dataset

In [21]:
```
# Sample images and labels from the modified training dataset
sample_images = []
sample_labels = []

for image, label in mod_test_ds.take(8):
    sample_images.append(image.numpy())
    sample_labels.append(label.numpy())

# Plot the sampled images
fig, ax = plt.subplots(2, 4, figsize=(16, 8))
for i, img in enumerate(sample_images):
    ax[i//4, i%4].imshow(img)
    ax[i//4, i%4].axis('off')
    ax[i//4, i%4].set_title(sample_labels[i])
plt.show()
```



Data Preprocessing

Libraries

In [19]:
```
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

from tensorflow.python import keras
from tensorflow.python.keras.models import Sequential
from keras.utils import plot_model

from tensorflow.python.keras.layers import Dense, Flatten, Conv2D, Dropout, MaxPool
from tensorflow.keras.utils import to_categorical

import seaborn as sns
```

Feature Extraction

SEperate images and labels in the datset

- Split data into 70:15:15 test, train and validation sets
- X represent features(images)
- y represents classes/labels

In [20]:
```
X = []
y = []
for image, label in mod_train_ds:
    X.append(image.numpy())
    y.append(label.numpy())

# Split the data into train_full and test sets (15% test)
X_train_full, X_test, y_train_full, y_test = train_test_split(X, y, test_size=0.15,

# Split the train_full set into train and validation sets (15% validation)
X_train, X_val, y_train, y_val = train_test_split(X_train_full, y_train_full, test_
```

Normalise the split dataset by /255

In [21]:
```
#images/features
X_train = np.array(X_train) / 255.0
X_val = np.array(X_val) / 255.0
X_test = np.array(X_test) / 255.0

# Labels
y_train = np.array(y_train)
y_val = np.array(y_val)
y_test = np.array(y_test)
```

In [22]:
```
# print the shapes of the train set
print("The model performance for training set")
print("-------------------------------------")
print('The shape of X_train is:',X_train.shape)
print("\n")

# print the shapes of the train set
print("The model performance for training set")
print("-------------------------------------")
print('The shape of y_train is:',y_train.shape)
print("\n")

# print the shapes of the train set
print("The model performance for training set")
print("-------------------------------------")
print('The shape of X_test is:',X_test.shape)
print("\n")

# print the shapes of the train set
print("The model performance for training set")
print("-------------------------------------")
print('The shape of y_test is:',y_test.shape)
print("\n")

# print the shapes of the train set
print("The model performance for training set")
print("-------------------------------------")
print('The shape of X_val is:',X_val.shape)
print("\n")

# print the shapes of the train set
print("The model performance for training set")
print("-------------------------------------")
print('The shape of y_val is:',y_val.shape)
print("\n")
```

```
The model performance for training set
--------------------------------------
The shape of X_train is: (14931, 150, 150, 3)


The model performance for training set
--------------------------------------
The shape of y_train is: (14931,)


The model performance for training set
--------------------------------------
The shape of X_test is: (3101, 150, 150, 3)


The model performance for training set
--------------------------------------
The shape of y_test is: (3101,)


The model performance for training set
--------------------------------------
The shape of X_val is: (2636, 150, 150, 3)


The model performance for training set
--------------------------------------
The shape of y_val is: (2636,)
```

MODELS(Supervised Learning)

Target classes(Parasitized: 0 and Uninfected:1)

I will build different models and use them to come up with an ensemble model to perform prediction on the blood smears.

Both the train and validation sets are imbalanced in comparison to the distribution of the 'malaria' dataset

Convolution Neural Network

We will use the sequential model which is a linear stack of layers. It can be first initialized and then we add layers using add method or we can add all layers at init stage.

The different layers are the :

- 2D Convolution layer

- Maxpooling layer which will be used for feature extraction(reduce spatial dimensions)
- Flatten : this will act as our input
- Dense : Which will act as the output layer
- Finally, we will use optimizers to help improve the accuracy of the model and reduce the losses.

In [56]:
```
def conv_block(filters):
    block = tf.keras.Sequential([
        tf.keras.layers.SeparableConv2D(filters, 3, activation='relu', padding='sam
        tf.keras.layers.SeparableConv2D(filters, 3, activation='relu', padding='sam
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.MaxPool2D()
    ]
    )

    return block

def dense_block(units, dropout_rate):
    block = tf.keras.Sequential([
        tf.keras.layers.Dense(units, activation='relu'),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Dropout(dropout_rate)
    ])

    return block
```

In [57]:
```
def build_model():
    cnn_model = tf.keras.Sequential([
        tf.keras.Input(shape=(150, 150, 3)),

        tf.keras.layers.Conv2D(16, 3, activation='relu', padding='same'),
        tf.keras.layers.Conv2D(16, 3, activation='relu', padding='same'),
        tf.keras.layers.MaxPool2D(),

        conv_block(32),
        conv_block(64),

        conv_block(128),
        tf.keras.layers.Dropout(0.2),

        conv_block(256),
        tf.keras.layers.Dropout(0.2),

        tf.keras.layers.Flatten(),
        dense_block(512, 0.7),
        dense_block(128, 0.5),
        dense_block(64, 0.3),

        tf.keras.layers.Dense(1, activation='sigmoid')
    ])

    return cnn_model
```

In [58]:
```
cnn_model = build_model()

cnn_model.compile(
    optimizer='adam',
    loss='binary_crossentropy',
    metrics=['accuracy']
)
```

In [59]:
```
cnn_model.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_27 (Conv2D)          (None, 150, 150, 16)      448

 conv2d_28 (Conv2D)          (None, 150, 150, 16)      2320

 max_pooling2d_27 (MaxPoolin  (None, 75, 75, 16)       0
 g2D)

 sequential_1 (Sequential)   (None, 37, 37, 32)        2160

 sequential_2 (Sequential)   (None, 18, 18, 64)        7392

 sequential_3 (Sequential)   (None, 9, 9, 128)         27072

 dropout (Dropout)           (None, 9, 9, 128)         0

 sequential_4 (Sequential)   (None, 4, 4, 256)         103296

 dropout_1 (Dropout)         (None, 4, 4, 256)         0

 flatten_9 (Flatten)         (None, 4096)              0

 sequential_5 (Sequential)   (None, 512)               2099712

 sequential_6 (Sequential)   (None, 128)               66176

 sequential_7 (Sequential)   (None, 64)                8512

 dense_21 (Dense)            (None, 1)                 65

=================================================================
Total params: 2,317,153
Trainable params: 2,314,785
Non-trainable params: 2,368
_________________________________________________________________
```

Callbacks

We need to define some hyper parameters including the learning rates, in order to design a model that does not overfit. High learning rates will make the model slow, therefore, I will approach the exponential approach. In this type of approach, our learning rate will be changing with each epoch.

In [60]:
```
checkpoint_cb = tf.keras.callbacks.ModelCheckpoint("malaria_classification_model.h5
                                                   save_best_only=True)

early_stopping_cb = tf.keras.callbacks.EarlyStopping(patience=3,
                                                     restore_best_weights=True)

#specifies the exponential function with the initial learning rate(lr0) and the dec

def exponential_decay(lr0, s):
    def exponential_decay_fn(epoch):
        return lr0 * 0.1 **(epoch / s)
    return exponential_decay_fn

exponential_decay_fn = exponential_decay(0.01, 20)

lr_scheduler = tf.keras.callbacks.LearningRateScheduler(exponential_decay_fn)
```

Training

In [61]:
```
malaria_train_model = cnn_model.fit(
    X_train, y_train,
    epochs=no_epochs,
    verbose=1,
    validation_data=(X_val, y_val),
    callbacks = [checkpoint_cb, early_stopping_cb, lr_scheduler]
)
```

```
Epoch 1/5
467/467 [==============================] - 1843s 4s/step - loss: 0.6802 - accurac
y: 0.5998 - val_loss: 9.4372 - val_accuracy: 0.4920 - lr: 0.0100
Epoch 2/5
467/467 [==============================] - 1648s 4s/step - loss: 0.6447 - accurac
y: 0.6432 - val_loss: 1.0229 - val_accuracy: 0.5224 - lr: 0.0089
Epoch 3/5
467/467 [==============================] - 1620s 3s/step - loss: 0.6399 - accurac
y: 0.6468 - val_loss: 0.6956 - val_accuracy: 0.5303 - lr: 0.0079
Epoch 4/5
467/467 [==============================] - 1644s 4s/step - loss: 0.6330 - accurac
y: 0.6520 - val_loss: 0.6922 - val_accuracy: 0.5080 - lr: 0.0071
Epoch 5/5
467/467 [==============================] - 1650s 4s/step - loss: 0.4404 - accurac
y: 0.7941 - val_loss: 1.1794 - val_accuracy: 0.4962 - lr: 0.0063
```

In [62]:
```
cnn_model.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_27 (Conv2D)          (None, 150, 150, 16)      448

 conv2d_28 (Conv2D)          (None, 150, 150, 16)      2320

 max_pooling2d_27 (MaxPoolin  (None, 75, 75, 16)       0
 g2D)

 sequential_1 (Sequential)   (None, 37, 37, 32)        2160

 sequential_2 (Sequential)   (None, 18, 18, 64)        7392

 sequential_3 (Sequential)   (None, 9, 9, 128)         27072

 dropout (Dropout)           (None, 9, 9, 128)         0

 sequential_4 (Sequential)   (None, 4, 4, 256)         103296

 dropout_1 (Dropout)         (None, 4, 4, 256)         0

 flatten_9 (Flatten)         (None, 4096)              0

 sequential_5 (Sequential)   (None, 512)               2099712

 sequential_6 (Sequential)   (None, 128)               66176

 sequential_7 (Sequential)   (None, 64)                8512

 dense_21 (Dense)            (None, 1)                 65

=================================================================
Total params: 2,317,153
Trainable params: 2,314,785
Non-trainable params: 2,368
_________________________________________________________________
```

In [63]:
```
cnn_model.evaluate(X_test, y_test)
```

```
97/97 [==============================] - 131s 1s/step - loss: 1.1925 - accuracy:
0.4921
```

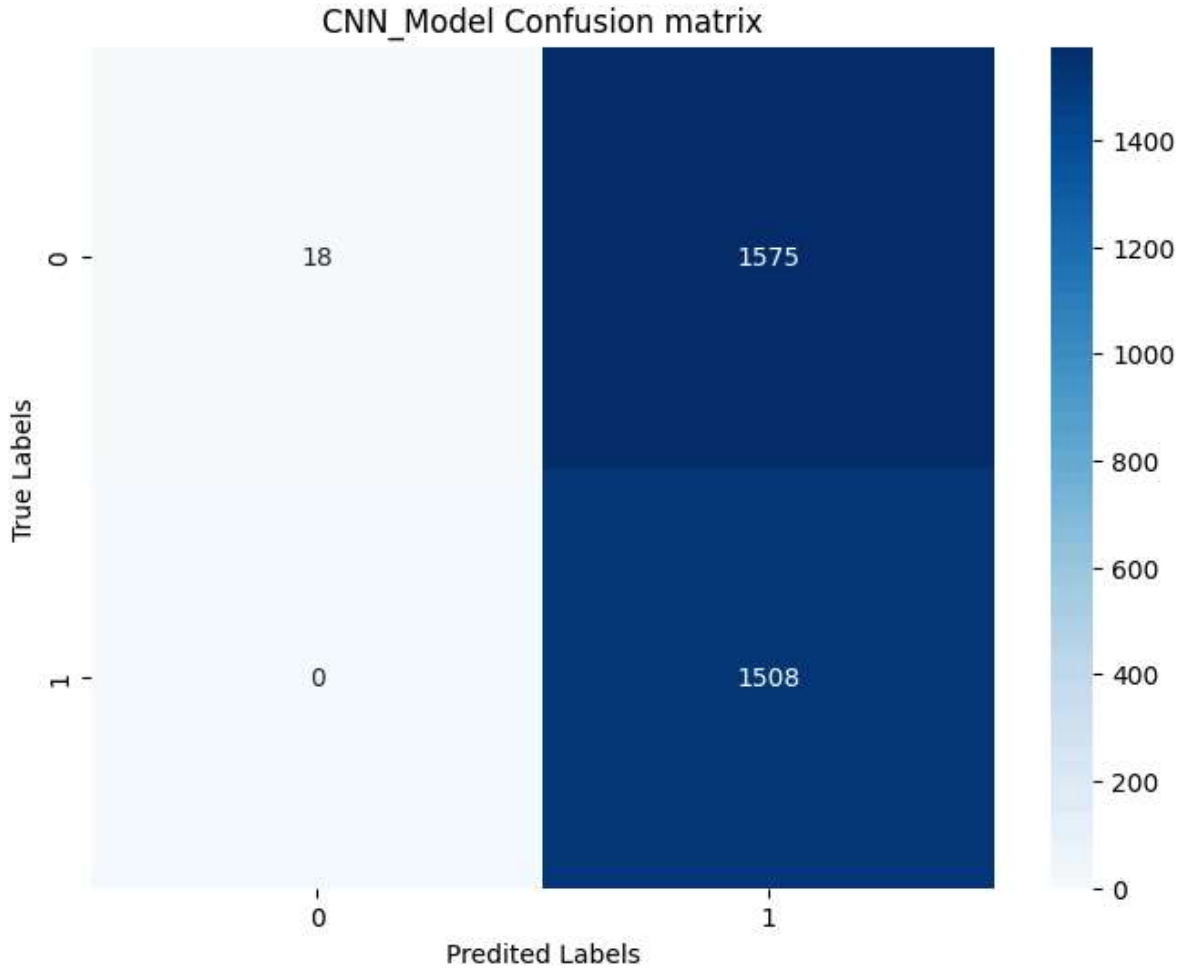
Out[63]:
```
[1.1924573183059692, 0.4920993149280548]
```

In [65]:
```
Y_test_pred = cnn_model.predict(X_test)
Y_test_pred_labels = Y_test_pred > 0.5
Y_test_true_labels = y_test
```

```
97/97 [==============================] - 131s 1s/step
```

In [67]:
```
cnn_confusion_matrix = confusion_matrix(Y_test_true_labels, Y_test_pred_labels)

plt.figure(figsize=(8, 6))
sns.heatmap(cnn_confusion_matrix, annot=True, fmt="d", cmap="Blues")
plt.ylabel('True Labels')
```

```
plt.xlabel('Predited Labels')
plt.title('CNN_Model Confusion matrix')

# save the confusion matrix
plt.savefig('confusion_matrixV2.png')

plt.show()
```



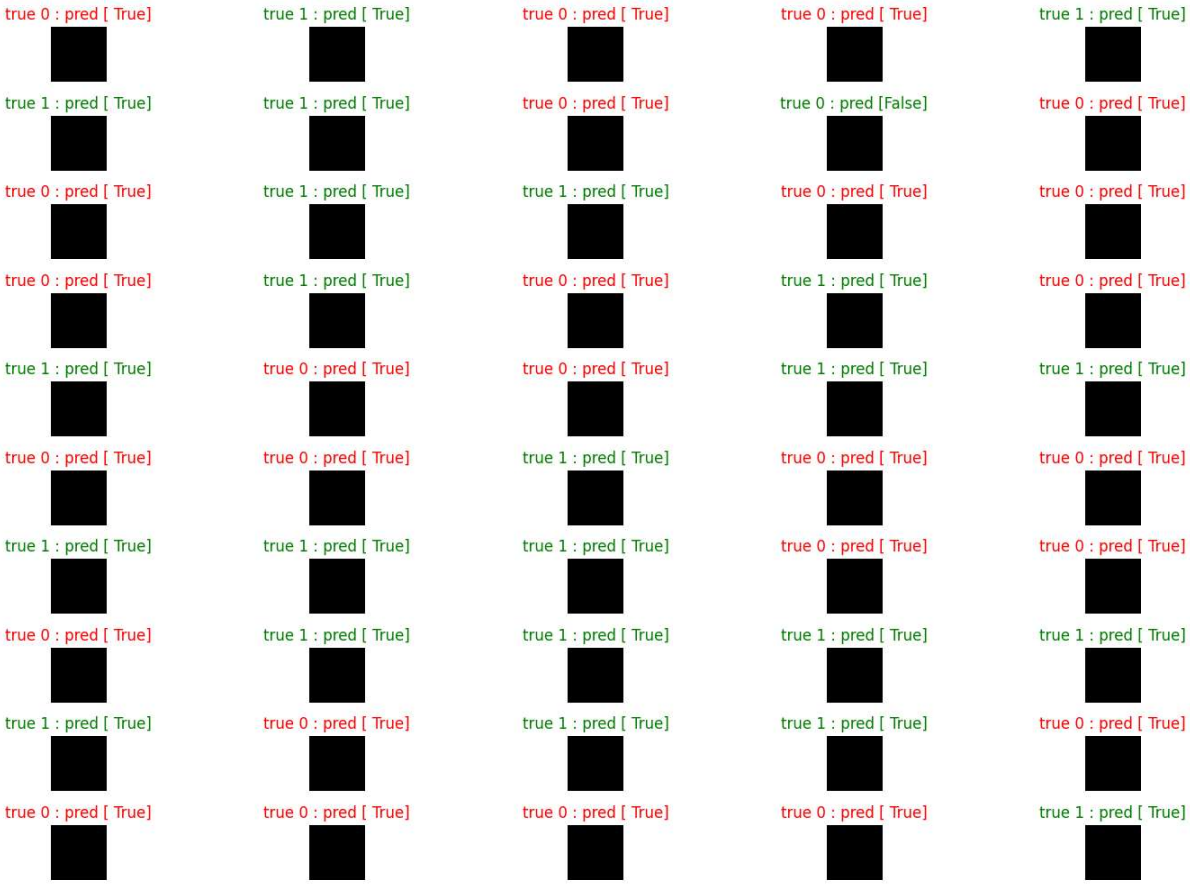
Classification report

In [68]:
```
cnn_report = classification_report(Y_test_true_labels, Y_test_pred_labels, digits=4
print(cnn_report)
```

```
              precision    recall  f1-score   support

           0     1.0000    0.0113    0.0223      1593
           1     0.4891    1.0000    0.6569      1508

    accuracy                         0.4921      3101
   macro avg     0.7446    0.5056    0.3396      3101
weighted avg     0.7516    0.4921    0.3309      3101
```

Graphically represent the predicted images

In [69]:
```
fig, axs = plt.subplots(10, 5, figsize=(16, 10))
axs = axs.flatten()

for ax in axs:
    r = np.random.randint(0, X_test.shape[0])
    true_label = Y_test_true_labels[r]
    pred_label = Y_test_pred_labels[r]
    title_color = 'green' if true_label == pred_label else 'red'
    ax.imshow(X_test[r], cmap="gray")
    ax.set_title('true {} : pred {}'.format(true_label, pred_label), color=title_co
    ax.axis('off')

plt.tight_layout()

plt.savefig('predicted_outputV1.pdf')

plt.show()
```



From the above the accuracy is very low, I will retrain the model with epoch = 20 and anlyse if there is an increase in accuracy

I am using the ensembe approach, here hoping to achieve a higher accuracy.

In [24]:
```
# Model
v2_model = Sequential()
# Add convolution 2D
v2_model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 kernel_initializer='he_normal',
                 input_shape=(150, 150, 3)))
v2_model.add(MaxPooling2D((2, 2)))
# Add dropouts to the model
v2_model.add(Dropout(0.25))
v2_model.add(Conv2D(64,
                 kernel_size=(3, 3),
                 activation='relu'))
v2_model.add(MaxPooling2D(pool_size=(2, 2)))
# Add dropouts to the model
v2_model.add(Dropout(0.25))
v2_model.add(Conv2D(128, (3, 3), activation='relu'))
# Add dropouts to the model
v2_model.add(Dropout(0.4))
v2_model.add(Flatten())
v2_model.add(Dense(128, activation='relu'))
# Add dropouts to the model
v2_model.add(Dropout(0.3))
v2_model.add(Dense(categories, activation='softmax'))
```

In [29]:
```
def conv_block(filters):
    block = tf.keras.Sequential([
        tf.keras.layers.SeparableConv2D(filters, 3, activation='relu', padding='sam
        tf.keras.layers.SeparableConv2D(filters, 3, activation='relu', padding='sam
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.MaxPool2D()
    ]
    )

    return block

def dense_block(units, dropout_rate):
    block = tf.keras.Sequential([
        tf.keras.layers.Dense(units, activation='relu'),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Dropout(dropout_rate)
    ])

    return block
```

In [30]:
```
def build_model():
    cnn_model = tf.keras.Sequential([
        tf.keras.Input(shape=(150, 150, 3)),

        tf.keras.layers.Conv2D(16, 3, activation='relu', padding='same'),
        tf.keras.layers.Conv2D(16, 3, activation='relu', padding='same'),
        tf.keras.layers.MaxPool2D(),

        conv_block(32),
        conv_block(64),
```

```
        conv_block(128),
        tf.keras.layers.Dropout(0.2),

        conv_block(256),
        tf.keras.layers.Dropout(0.2),

        tf.keras.layers.Flatten(),
        dense_block(512, 0.7),
        dense_block(128, 0.5),
        dense_block(64, 0.3),

        tf.keras.layers.Dense(1, activation='sigmoid')
    ])

    return cnn_model
```

In [31]:
```
v2_model = build_model()

v2_model.compile(
    optimizer='adam',
    loss='binary_crossentropy',
    metrics=['accuracy']
)
```

In [32]:
```
checkpoint_cb2 = tf.keras.callbacks.ModelCheckpoint("malaria_classification_model_V
                                                    save_best_only=True)

early_stopping_cb2 = tf.keras.callbacks.EarlyStopping(patience=3,
                                                      restore_best_weights=True)
```

In [33]:
```
malaria_train_model =v2_model.fit(
    X_train, y_train,
    epochs=6,
    verbose=1,
    validation_data=(X_val, y_val),
    callbacks = [checkpoint_cb2, early_stopping_cb2]
)
```

```
Epoch 1/6
467/467 [==============================] - 1971s 4s/step - loss: 0.7454 - accurac
y: 0.5831 - val_loss: 1.1728 - val_accuracy: 0.5076
Epoch 2/6
467/467 [==============================] - 1903s 4s/step - loss: 0.3081 - accurac
y: 0.8803 - val_loss: 0.4826 - val_accuracy: 0.8168
Epoch 3/6
467/467 [==============================] - 1905s 4s/step - loss: 0.1867 - accurac
y: 0.9453 - val_loss: 0.3570 - val_accuracy: 0.8703
Epoch 4/6
467/467 [==============================] - 1809s 4s/step - loss: 0.1744 - accurac
y: 0.9464 - val_loss: 1.1690 - val_accuracy: 0.7060
Epoch 5/6
467/467 [==============================] - 1757s 4s/step - loss: 0.1710 - accurac
y: 0.9492 - val_loss: 2.8626 - val_accuracy: 0.4924
Epoch 6/6
467/467 [==============================] - 1702s 4s/step - loss: 0.1622 - accurac
y: 0.9507 - val_loss: 1.2072 - val_accuracy: 0.7181
```

In [34]:
```
v2_model.summary()
```

```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)           (None, 150, 150, 16)      448

 conv2d_3 (Conv2D)           (None, 150, 150, 16)      2320

 max_pooling2d_1 (MaxPooling  (None, 75, 75, 16)       0
 2D)

 sequential (Sequential)     (None, 37, 37, 32)        2160

 sequential_1 (Sequential)   (None, 18, 18, 64)        7392

 sequential_2 (Sequential)   (None, 9, 9, 128)         27072

 dropout (Dropout)           (None, 9, 9, 128)         0

 sequential_3 (Sequential)   (None, 4, 4, 256)         103296

 dropout_1 (Dropout)         (None, 4, 4, 256)         0

 flatten (Flatten)           (None, 4096)              0

 sequential_4 (Sequential)   (None, 512)               2099712

 sequential_5 (Sequential)   (None, 128)               66176

 sequential_6 (Sequential)   (None, 64)                8512

 dense_3 (Dense)             (None, 1)                 65

=================================================================
Total params: 2,317,153
Trainable params: 2,314,785
Non-trainable params: 2,368
_________________________________________________________________
```

Predict the values

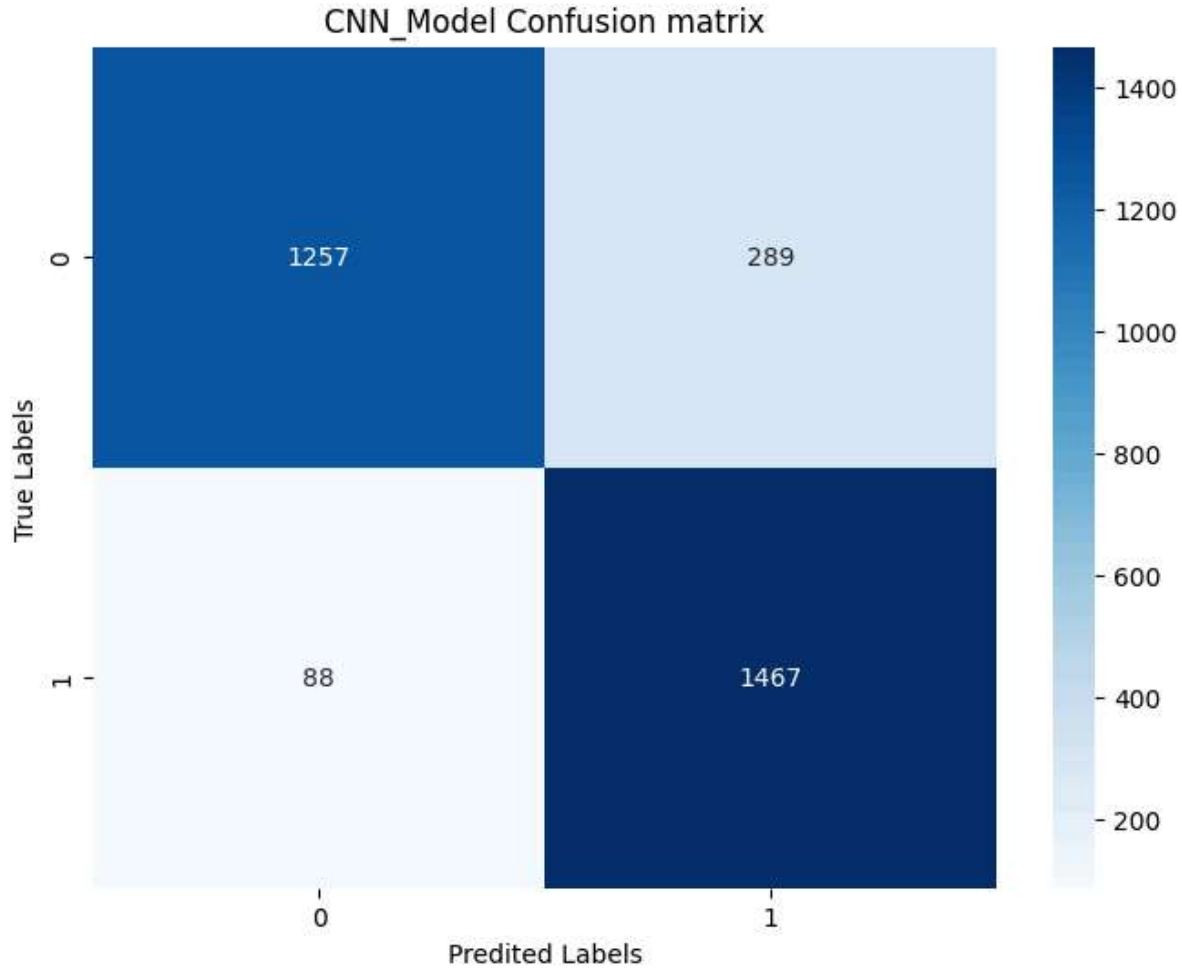
In [36]:
```
Y_test_pred = v2_model.predict(X_test)
Y_test_pred_labels = Y_test_pred > 0.5
Y_test_true_labels = y_test
```

```
97/97 [==============================] - 131s 1s/step
```

Plot the confusion matrix

In [37]:
```
cnnV2_confusion_matrix = confusion_matrix(Y_test_true_labels, Y_test_pred_labels)

plt.figure(figsize=(8, 6))
sns.heatmap(cnnV2_confusion_matrix, annot=True, fmt="d", cmap="Blues")
plt.ylabel('True Labels')
plt.xlabel('Predited Labels')
```

```
plt.title('CNN_Model Confusion matrix')

# save the confusion matrix
plt.savefig('confusion_matrixV2.png')

# plot it
plt.show()
```

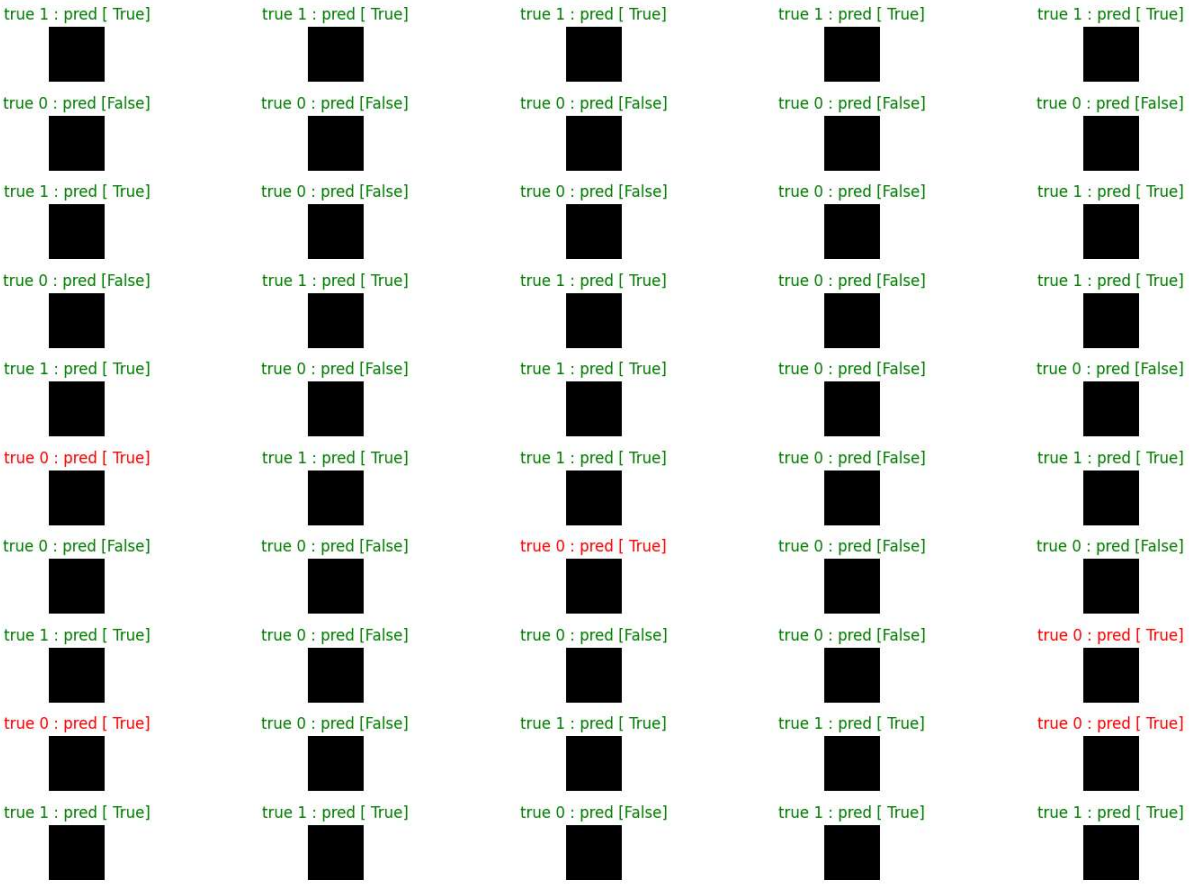


In [38]:
```
cnnV2_report = classification_report(Y_test_true_labels, Y_test_pred_labels, digits
print(cnnV2_report)
```

```
              precision    recall  f1-score   support

           0     0.9346    0.8131    0.8696      1546
           1     0.8354    0.9434    0.8861      1555

    accuracy                         0.8784      3101
   macro avg     0.8850    0.8782    0.8779      3101
weighted avg     0.8849    0.8784    0.8779      3101
```

Visualize the predicted classes and save the output as a pdf. file

In [39]:
```
fig, axs = plt.subplots(10, 5, figsize=(16, 10))
axs = axs.flatten()

for ax in axs:
    r = np.random.randint(0, X_test.shape[0])
    true_label = Y_test_true_labels[r]
    pred_label = Y_test_pred_labels[r]
    title_color = 'green' if true_label == pred_label else 'red'
    ax.imshow(X_test[r])
    ax.set_title('true {} : pred {}'.format(true_label, pred_label), color=title_co
    ax.axis('off')

plt.tight_layout()

plt.savefig('predicted_outputV2.pdf')

plt.show()
```



Load the trained model

In [41]:
```
malaria_model = tf.keras.models.load_model('malaria_classification_model_V2.h5')
```

The accuracy of the ensembled model improve from 42% to 87% giving us 45 coreect predictions out of 50 samples.

In [ ]: