POST	/login	Authorize User To Access The App
POST	/signup	Create new user account
GET	/scoreboard	List of users and their scores
POST	/models	Admin publishes a new model
GET	/models	List of available models
GET	/models/<id>	Get model description and info
DELETE	/models/<id>	Remove a challenge
POST	/models/<id>/model	Upload trained model
POST	/models/<id>/dataset	Upload dataset
POST	/models/<id>/attack	Upload images to attack a model
GET	/models/<id>/model	Download the model
GET	/models/<id>/dataset	Download the dataset

HEADERS:

With the exception of /login and /signup all endpoints require a “Cybnetics-Token” header with an auth token that you can get from calling /login or /signup

POST **/login** Authorize User To Access The App

Request Params (body):

```
{
    "username": "user1234",
    "password": "password"
}
```

Returns:

Situation	Value	HTTP Code	Type
success	JSON	200	Application/JSON
invalid input data	Bad Request	400	text/plain
Bad username or password	Bad Request	401	text/plain

```
{
    "token": "generated jwt”,
}
```

POST **/signup** Create new user account

Request Params (body):

```
{
    "username": "user1234",
    "password": "password"
}
```

Returns:

Situation	Value	HTTP Code	Type
success	JSON	201	Application/JSON
invalid input data	Bad Request	400	text/plain
User already exists	Forbidden	401	text/plain

```
{
   "token": "generated jwt”
}
```

GET **/scoreboard** List of users and their scores

Request Params (**QUERY**):

```
username=<specific username (optional)>
```

Returns:

Situation	Value	HTTP Code	Type
success	JSON	200	Application/JSON
invalid token OR token expired	Invalid Auth Token	401	text/plain
invalid request	Invalid Request	400	text/plain

```
{
    "total_successes": 328,
    "total_attempts": 615,
    "gold_medals": 23,
    "silver_medals": 42,
    "bronze_medals": 61,
    "users": [
       {
          "_id": "user1234",
          "total_points": 1500,
          "total_attempts": 1,
          "total_sucesses": 1,
          "attacked_models": [
             {
                "_id": "898879ae (model_id)"
                "name": "cat picture recognition",
                "successes": 1,
                "attempts": 1,
                "points_earned": 500,
             },
             ...
          ],
       }
    ]
}
```

POST **/models** Admin publishes a new model

Request Params (body):

```
{
    "name": "challenge name”,
    "description": "wordy explanation”,
    "model_type": "cifar|mnist”,
```

```
    "attack_mode": "white|gray|black”,
    "dropouts": [{(optional) see dropout spec}], //pass empty array if there are none
    "pools": [{(optional)see pools spec}], //pass empty array if there are none
    "layers": [{see layers spec}],
    "color": true, // false if input images are black and white
}
```

Dropouts is an array of items with the following json schema, used for the specification of a model

```
{
       "type": "object",
       "properties": {
       "type": {"type": "string"}, // a dropout class in torch.nn. eg Dropout2d
       "args": {"type": "array", "items": {"type": "number"}}, // args to dropout constructor
       },
       "required": ["type", "args"]
}
```

Pools is an array of items with the following json schema, used for the specification of a model

```
{
       "type": "object",
       "properties": {
       "type": {"type": "string"}, // a pool class in torch.nn. eg MaxPool2d
       "args": {"type": "array", "items": {"type": "number"}}, // args to pool constructor
       },
       "required": ["type", "args"]
}
```

Layers is an array of items with the following json schema, used for the specification of a model

```
{
       "type": "object",
       "properties": {
       "type": {"type": "string"}, // a layer class in torch.nn. eg Conv2d
       "args": {"type": "array", "items": {"type": "number"}}, // args to the layer constructor
       "activation": {"type": "string"}, // activation function in torch.nn.functional. eg relu
       "pool": {"type": "number"}, // the index of the pool that this layer uses
        "dropout": {"type": "number"}, // the index of the dropout that this layer uses
       "name": {"type": "string"}, // the name of the layer for training purposes
       },
       "required": ["type", "args", "name"] //name is not required when type=="view"or“flatten”
}
```

Returns:

Situation	Value	HTTP Code	Type
success	json	200	application/json
Invalid request	Invalid Request	400	text/plain
Invalid or expired token	Invaid Auth Token	401	text/plain

```
{
    "_id": "id of challenge created”,
    "name": "challenge name”,
    "description": "wordy explanation”,
    "attack_mode": "white|gray|black”,
    "ready": false, // set to true once all the required resources are uploaded for this attack mode
    "owner": "owners username”
}
```

GET /models — List of available models

Request Params (**QUERY**):

```
query=<(optional) text query>
user=<(optional) only return results for this user>
attack_mode=<(optional) only return results with this attack mode>
```

NOTE: if the user is the requester's own username it will show all models regardless if they are read or not, but if it is not, it will show all public and ready models

Example query: "Foo" will return results with foo in the name or description, case insensitive. You can also search multiple words "foo bar" (ok!)

Returns:

Situation	Value	HTTP Code	Type
success	json	200	application/json
No matching model		404	
Invalid request	Invalid Request	400	text/plain
Invalid or expired token	Invalid Auth Token	401	text/plain

```
[
    {
        "_id": "id of challenge created”,
        "name": "challenge name”,
        "description": "wordy explanation”,
```

```
        "attack_mode": "white|gray|black”,
        "owner": "owners username”
    },
    {
        "_id": "id of challenge created”,
        "name": "challenge name”,
        "description": "wordy explanation”,
        "attack_mode": "white|gray|black”,
        "owner": "owners username”
    },
]
```

GET	/models/<id>	Get model description and info

Returns:

Situation	Value	HTTP Code	Type
Success	json	200	application/json
Invalid request	Invalid Request	400	text/plain
Invalid or expired auth token	Invalid Auth Token	401	text/plain
Model not found		404	

```
{
    "_id": "id of challenge created”,
    "name": "challenge name”,
    "description": "wordy explanation”,
    "attack_mode": "white|gray|black”,
    "owner": "owners username”
}
```

DELETE	/models/<id>	Remove a challenge

Returns:

Situation	Value	HTTP Code	Type
Success (no content)		204	
Invalid request	Invalid Request	400	text/plain
Invalid or expired auth token	Invalid Auth Token	401	text/plain

Model not found		404	

POST /models/<id>/model — Upload trained model

Request Params (**FILE**):

```
model=<Model image file>
See js form file upload
```

Returns:

Situation	Value	HTTP Code	Type
Success (no content)	Lol :)	204	
Invalid request	Invalid Request	400	text/plain
Wrong file type	Wrong File Type	415	text/plain
Invalid or expired token	Invalid Auth Token	401	text/plain

POST /models/<id>/dataset — Upload dataset

Request Params (**FILE**):

```
dataset=<Dataset zip>
See js form file upload
```

Returns:

Situation	Value	HTTP Code	Type
Success (no content)		204	
Invalid request	Invalid Request	400	text/plain
Wrong file type	Wrong File Type	415	text/plain
Invalid or expired token	Invalid Auth Token	401	text/plain

POST /models/<id>/attack — Upload images to attack a model

Request Params (**QUERY**):

```
label=0 //Whatever number the original images label was
```

```

```

Request Params (**FILE**):

```
image=<Attack image>
See js form file upload
```

Returns:

Situation	Value	HTTP Code	Type
Success	json	200	application/json
Invalid request	Invalid Request	400	text/plain
Wrong file type	Wrong File Type	406	text/plain
Invalid or expired token	Invalid Auth Token	401	text/plain

```
{
    "_id": "id of attack created”,
    'label': label,
    'predicted_label': predicted_label,
    'user': user,
    "success": true|false,
    "points": 1,
    "place": "gold|silver|bronze”,
}
```

GET **/models/<id>/model** Download the model

Returns:

Situation	Value	HTTP Code	Type
Success	<Model Image>	200	TODO
Invalid request	Invalid Request	400	text/plain
Invalid or expired auth token	Invalid Auth Token	401	text/plain
Model not found		404	
You do not have permission to download the model	Not Authorized For This Attack Mode	403	text/plain

```
The model image
```

GET	/models/<id>/dataset	Download the dataset

Returns:

Situation	Value	HTTP Code	Type
Success	<Dataset zip>	200	TODO
Invalid request	Invalid Request	400	text/plain
Invalid or expired auth token	Invalid Auth Token	401	text/plain
Model not found		404	
You do not have permission to download the model	Not Authorized For This Attack Mode	403	text/plain

The dataset zip