**RESTful API** for Smart India Hackathon Radionomics

Base domain: [http://brain.southindia.cloudapp.azure.com](http://brain.southindia.cloudapp.azure.com/)

Dedicated IP: 52.172.5.20

(Turn on the VM on when required. Keep it off when not required)

The updated version of the API always available in the gitlab repository https://gitlab.com/RitwikBhar/Radionomics

(First Prototype)

Note: The API can be used both from mobile app and web. To use the API from mobile app the key needs to be provided during api call using GET or POST according to the need of the API. But for the web the *key* parameter is not required. Instead a session containing key *username* should be present containing the *username of the logged in user*.

**Auth**

1. **Authentication**: URL: /api/auth/get-key

Request parameters: POST

username: string, required

password: string, required

Response: (JSON)

error: true/false

key: string, present only if error is F

The key is valid for 12hours. A new key is expected to be generated for each login

Sample response:

{error:true}

{error:false,key:”tidkdxk$%kjdik544sd5@@3”}

1. **New User:** URL: /api/auth/new

Request parameters: POST

name: string, required

username: string, required

email: string(email), required

phone: string(digits-only), optional

password: string, required

doctor\_reg\_id: string, required

Response: (JSON)

success: true/false

error\_msg: present only if success is F

Sample Response:

{success:true}

{success:false,error\_message:”Username Already Exists”}

1. **Check if username is usable:** URL:/api/auth/check-username

Request Parameters: GET

username: string, required

Sample request:

/auth/check-username?username=drtbiswas

Response: JSON

valid: true/false, true if username is valid and it does not exist

Sample Response:

{valid:false}

1. **Check if doctor id is usable:** URL: /api/auth/check-doctor-id

Request Parameters: GET

did: string

Sample Response:

{valid:false}

1. **Check if email id is usable:** URL: /api/auth/check-email

Request Parameters: GET

email: string

Sample Response:

{valid:false}

1. **Change Password:** URL: /api/auth/change-password

Request Parameters: POST  
key: string, required

old-password: string, required

new-password: string, required

Response: JSON

valid: true/false

**Patient-cases**

1. **Upload Patient MRI Scans:** URL: /api/patient-cases/upload

Request Params: POST  
key: string, required

case\_name: string, optional

files: file, multiple, required

Sample html form:

<form method=”POST” action=”baseurl/api/patient-cases/upload” enctype=”multipart/form-data”>

<input name=’case\_name’>

<input type=’hidden’ value=”your key” name=”key”>

<input type=’file’ name=’files[]’ multiple>

</form>

Response: JSON

error: object{

error: true/false

error\_message: string, present only error = T,

error\_type: string, present only if error = T,

files: array(Object{

reason: string,

file\_name: string,

})

}

case\_id: string,

patient\_name: string,

patient\_age: int,

case\_name: string(sent only of it was sent in the request),

images: array(Object{

series\_description: string,

series\_time: string,

images: array(Object{

filename: string,

url: string, without the key

})

})

1. **Get all the previous cases uploaded by a user:**

/api/patient-cases

Request parameters: GET

key: string, mandetory

page: int, optional

Note: if page is not provided all the cases in the uer's history will be sent. If the page is provided with the parameter 10 cases pre page will be sent.  
Response: JSON

error: Object{

error: true/false,

error\_message: string

},

cases : array(Object{

case\_id: string,

patient\_name: string,

patient\_age: int,

datetime: datetime,

files: array(Object{

series\_time: string,

series\_description: string,

images: array(Object{

filename: string,

url: string

})

})

})

1. **Get Details of a case:**

/api/patient-cases/<case\_id>

Request parameretes: GET

key: string, required

ignore\_images: optional

Response: JSON

error: Object{

error: true/false,

error\_message: string

},

case\_id: string,

patient\_name: string,

patient\_age: int,

case\_name: string(sent only of it was sent in the request),

date\_time: datetime,

images(sent if *ignore\_images* is not set): array(Object{

series\_description: string,

series\_time: string,

images: array(Object{

filename: string,

url: string, without the key

})

})

1. **Image URL**:

/api/patient-cases/<case\_id>/images/<index\_1>/<index\_2>?key=<key>

Returns the image, or 403 Access Denied, or 404 not found

1. **RI Mapped Image URL**:

/api/patient-cases/<case\_id>/ri-map/<index\_1>/<index\_2>?key=<key>

Returns the image, or 403 Access Denied, or 404 not found

1. **Count no of series in a case:**

/api/patient-cases/<case\_id>/images/count?key =<key>

Response: JSON

error: Object{

error: true/false,

error\_message: string

}

count: int, present only if error = F

1. **Count no of images in a series:**

/api/patient-cases/<case\_id>/images/<index\_1>/count?key=<key>

Response: JSON

error: Object{

error: true/false,

error\_message: string

}

count: int, present only if error = F

1. **Get all images in a series:**

/api/patient-cases/<case\_id>/images/<index\_1>?key=<key>

Response: JSON

error: Object{

error: true/false,

error\_message: string

}

images: array(Object{

filename: string,

url: string, url of the image without the key

})

1. **Get all images in a case:**

/api/patient-cases/<case\_id>/images?key=<key>

Response: JSON

error: Object{

error: true/false,

error\_message: string

}

images: array(Object{

series\_description: string,

series\_time: string,

images: array(Object{

filename: string,

url: string, without the key

})

})

1. **For API no. 11 and 12 replace *images* with *ri-map* to get the ri map solution.**
2. **Analysis of a single image:**

/api/patient-cases/<case\_id>/images/<index\_1>/<index\_2>/analysis

Request parameters: GET

key: string, mandatory

re\_analysis: true/false, optional

if false or not provided the image will be not be analysed again if it was analysed before.

If set then the image will be analysed again, previous analysis will be erased

response: JSON

error: Object{

error: true/false,

error\_message: string

}

infected\_areas: array(Object{

coordinate : array(int),//length 2 0:x, 1:y

area: float, //mm2

tumor\_width: float, // mm

tumor\_height: float, // mm

average\_chemical\_composition: array(float)//length 20,

update\_actual\_result\_url: string, //url where the actual result is to be posted

detected\_result: string, //the disease detected using ri

actual\_result: string //present only if analysis was previously uploaded, the string should be as string obtained from get Disease

}),

analysis\_image: url //string without the key

**Analysis Upload**

1. Upload the analysis data:

/api/patient-cases/<case\_id>/<int:index\_1>/<int:index\_2>/analysis/<analysis\_id>/<int:analysis\_index>/upload-analysis

Request Method: POST

key: mandatory

actual\_result: string, mandatory

response:

error: Object{

error: true/false,

error\_message: string

}

**Utility**

1. **Get Spectroscopy Data:**

/api/utility/get-spectroscopy

Request parameters: GET

key: string, required

pixel: int, required ( 255>pixel>0, however no data is also available for pixel < 85)

Response: JSON

error: Object{

error: true/false,

error\_message: string

},

x : array(float),

y:array(float),

labels: array(string)

Note: length of array is 20

1. **Get Disease Names:**

/api/utility/get-disease-names

Response:

array(string)