# Smart - UI

# Hi! I am Manan Goyal, a 3rd year CSE Undergrad student of IIT Bombay. Smart UI was a great opportunity and I got to learn a lot of things. I took this project entirely by myself due to which I had to put in more efforts probably, but seeing the results, I believe it was worth it! I would like to thank team Tech-fest and Morning Star for sharing this opportunity with me.

Looking forward to more such opportunities …

Table of Index :

[Installation on Ubuntu 20](#_t459j9icw5xw) **1**

[SYSTEM REQUIREMENTS](#_62f2p7gw0yks) 1

[REQUIREMENTS INSTALLATION](#_mtzr9f6yf7l2) 1

[Debian+Ubuntu](#_i9wfc8lhwzy5) 1

[SMART\_UI Installation](#_yacp5gcsf5d7) 2

[D. APPLICATION STRUCURE](#_jplmmy4o3hwe) 2

[E. USER GUIDE](#_motu2l5594i) 3

[**Stages**](#_wou0u9i9jwnq) **3**

[Parse with tesseract](#_lshhdc92mf8j) 3

[Reparse output to “zones db”](#_wphpvxdywpd8) 3

[Generate json output](#_tip8o6pslxlt) 3

[TLDR :](#_46q2ferwmdd3) **3**

# Installation on Ubuntu 20

## SYSTEM REQUIREMENTS

* PHP / cli
* Tesseract - ocr library

1. For extended version

* ImageMagic libraries - convert source .png files

## REQUIREMENTS INSTALLATION

### Debian+Ubuntu

sudo apt install php7.4-cli

sudo apt-get install php-mbstring

sudo apt install tesseract-ocr

sudo apt install imagemagick

MacOS

**PHP**

PHP is MAMP part

or use Homebreww

brew search php

brew install php / or / php@7.3

**Tesseract**

brew install tesseract --all-languages

**Imagemagick**

brew install imagemagick

## SMART\_UI Installation

1. **SmartUI.tgz ungzip/untar to directory**

tar -xzvf smartui\_xxx.tgz

1. **set permissions for directories**

chmod 755 smart\_ui -R

chmod 777 smart\_ui/\*.php

and / or

chown [myaccount] smart\_ui -R

## D. APPLICATION STRUCURE

conf/ - configuration file / system paths

vendor/ - PHP libraries / see composer.json

**System folders**

*source\_imgs /*

images for parsing

*encoded\_sources /*

helper encoded images for better results - encoded with imagemagick library convert

images created by 1.tesseract\_parser\_extended.php

*process\_data /*

output tesseract ocr process in HTML

created by

1.tesseract\_parser.php

1.tesseract\_parser\_extended.php

parse\_image-cli.php

*grid\_recognized /*

file database with overlapping output from different parsing methods

filenames and serialized php arrays in files in understandable structure for extending

and finetuning json output

created by

3.grid2josn-extended.php

3.grid2josn.php

*output\_json /*

json file per image

created by

2.elements2grid.php

## E. USER GUIDE

SmartUI is CLI application, run from command line

Put images in .png format are parsed from **source\_imgs** folder

Parsing only one specific image for testing, debuging could be done with

php ./parse\_image-cli.php path-to-image.png

Parsing all images with helper files

php

# Stages

Batch rendering more images:

(blue - command)

## Parse with tesseract

add \*.png images to ./source\_imgs

php ./1.tesseract\_parser.php

results in ./process\_data/[image filename]/[methodname.html]

## Reparse output to “zones db”

php ./2.elements2grid.php

## Generate json output

php ./3.grid2json.php

# **TLDR :**

copy images to source\_imgs/

just run the **./smart\_ui.sh**

or

just run the following scripts as **./[filename].php :**

1.tesseract\_parser.php

2.elements2grid.php

3.grid2json.php

# Next steps, what can be improved

smart\_UI structure is prepeared for next extending

zone db is open for properties from different parsing, recognizing, learning tools

Use teseract to "ocr" recognizing html5 form elements,

train library for this “specific font”.

Use an imagemagick histogram for extending elements with color and box properties.

Use autotracer + tesselart for html5 boxes and elements.

# Other good solutions for your information

other ways:

( but I don't recommend switch to it, my solution is extendable

also with some internal logic - "AI" )

1.

- GoogleCloudVision API

2.

- really impressive scientific project:

<https://arxiv.org/abs/1705.07962>

( problematic is fast instalation with python libraries

- and solution in short time will be only interface

for already "learned" parsing)

3. interesting sources

doc about Mockups and Deep Learning

https://blog.floydhub.com/turning-design-mockups-into-code-with-deep-learning/

project with learning interface

https://gamera.informatik.hsnr.de/