|  |
| --- |
| C:\Users\CHAYAPATHI-CPN\Desktop\download.png  **Department of Information Science and Engineering** |
| **Acharya Institute of Technology** |
| Acharya Dr. Sarvepalli Radhakrishnan Road,Bangalore-560107 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **WEEKLY PROGRESS REPORT** | | | | | |
| **Batch No** | | **05** | | | |
| **Guide** | |  | | | |
| **Project Title** | | **IoT based Wireless Smart Board** | | | |
| **Progress Report No** | | **4** | | | |
| **Date of Submission** | | **03 April 2019** | | | |
| **Date** | | **From: 13 March 2019** | | | **To:01 April 2019** |
|  | | | | | |
| **Sl. No.** | **Student Name** | | **USN** | **Signature with date** | |
| **1** | Pravesh Kasaundhan | | **1AY15IS072** |  | |
| **2** |  | |  |  | |
| **3** |  | |  |  | |
| **4** |  | |  |  | |
|  | | | | | |
|  | | | | | |

Progress Report:

List of Features Implemented as per the Gantt chart

1. Create a bootable SD card installed with Raspbian OS.

|  |
| --- |
|  |

1. Implementation of code to register device and Raspberry Pi on to the online Dataplicity Portal.
2. Unit testing to check for correct installation of Raspbian OS and to check if the device is recognised by the Dataplicity

|  |
| --- |
|  |

**Create a bootable SD card installed with Raspbian OS.**

Official images for recommended operating systems are available to download from the Raspberry Pi website https://www.raspberrypi.org/downloads/

We will need to use an image writing tool to install the image you have downloaded on your SD card for that purpose we will be using Etcher

**Etcher** is a graphical SD card writing tool that works on Mac OS, Linux and Windows, and is the easiest option for most users. Etcher also supports writing images directly from the zip file, without any unzipping required. To write your image with Etcher:

* Download [Etcher](https://etcher.io/) and install it.
* Connect an SD card reader with the SD card inside.
* Open Etcher and select from your hard drive the Raspberry Pi .img or .zip file you wish to write to the SD card.
* Select the SD card you wish to write your image to.
* Review your selections and click 'Flash!' to begin writing data to the SD card.

**Implementation of code to register device and Raspberry Pi on to the online Dataplicity Portal**.

This script prepares your Raspberry Pi or other Linux device for use with the Dataplicity agent, an open source application that powers remote terminals and other features.

from \_\_future\_\_ import division, print\_function, unicode\_literals

import datetime

import inspect

import json

import locale

import os

import platform

import pwd

import subprocess

import sys

import time

import traceback

import urllib

import urllib2

from functools import partial

# Personalized settings for the installer

SETTINGS = """

{

"agent\_download\_url": "https://github.com/wildfoundry/dataplicity-agent/releases/download/v0.4.34/dataplicity",

"agent\_version": "v0.4.34",

"api\_url": "https://api.dataplicity.com/",

"device\_url": "https://www.dataplicity.com/devices/",

"dry\_run": false,

"interactive": true,

"m2m\_url": "wss://m2m.dataplicity.com/m2m/",

"register\_url": "https://www.dataplicity.com/install/",

"report\_url": "https://www.dataplicity.com/installer\_event/",

"token": "xum0q6h3",

"time": 1554222248.9204059

}

"""

settings = json.loads(SETTINGS)

AGENT\_BASE\_PATH = "/opt/dataplicity/agent"

AUTH\_PATH = "/opt/dataplicity/tuxtunnel/auth"

CPU\_INFO\_PATH = "/proc/cpuinfo"

DOWNLOAD\_CHUNK\_SIZE = 16384

LOG\_PATH = "/var/log/dpinstall.log"

MAX\_STEPS = 4

SERIAL\_PATH = "/opt/dataplicity/tuxtunnel/serial"

SUDOER\_LINE = "dataplicity ALL=(ALL) NOPASSWD: /sbin/reboot"

SUDOERS\_PATH = "/etc/sudoers"

SUPERVISOR\_CONF\_DIR\_PATH = "/etc/supervisor/conf.d/"

SUPERVISOR\_CONF\_PATH = "/etc/supervisor/conf.d/tuxtunnel.conf"

# A string template for the supervisor conf

SUPERVISOR\_CONF\_TEMPLATE = """

# Installed by Dataplicity Agent installer

[program:tuxtunnel]

environment={ENV}

command={AGENT\_PATH} --server-url {API\_URL} run

autorestart=true

redirect\_stderr=true

user=dataplicity

stdout\_logfile=/var/log/dataplicity.log

stderr\_logfile=/var/log/dataplicity.log

"""

# Fields from /proc/cpuinfo we want in the report

CPU\_INFO\_FIELDS = {

"cpu architecture",

"cpu implementor",

"cpu part",

"cpu revision",

"cpu variant",

"hardware" "model name",

"revision",

}

RETURN\_SUCCESS = 0

RETURN\_USER\_ABORT = -1

RETURN\_FAIL = -2

RETURN\_CRASH = -3

open\_write\_binary = lambda path: open(path, "wb")

open\_append\_binary = lambda path: open(path, "ab")

open\_append\_text = lambda path: open(path, "at")

open\_read\_text = lambda path: open(path, "rt")

open\_read\_binary = lambda path: open(path, "rb")

subprocess\_check\_output = subprocess.check\_output

symlink = os.link

rename = os.rename

chmod = os.chmod

# Store the log in memory as well as write to file

install\_log = []

# Current step in the installation process, updated by `show\_step`

current\_step = 0

start\_time = time.time() # time installer started

last\_log\_time = start\_time # time since last log\_time

# Get the start time of the installer, in ISO 8601 format

# This is sent to keen so we can detect incorrect clocks

start\_time\_iso\_8601 = datetime.datetime.utcnow().isoformat()

class AbortInstall(Exception):

"""Installation can not continue."""

def create\_log\_file():

"""Create (and wipe) log file."""

try:

with open\_write\_binary(LOG\_PATH):

pass

except IOError:

# Probably means no permission

pass

def log(text, \*args, \*\*kwargs):

"""Log technical details to LOG\_PATH."""

caller = inspect.stack()[1][3] # Get the caller function name

log\_text = text.format(\*args, \*\*kwargs)

lines = (log\_text + "\n").splitlines()

install\_log.extend(lines)

write\_lines = (

"\n".join("{}: {}".format(caller.ljust(16), line) for line in lines) + "\n"

)

try:

with open\_append\_binary(LOG\_PATH) as log\_file:

log\_file.write(write\_lines.encode("utf-8", errors="replace"))

except IOError:

# Probably a permissions error

pass

def log\_time():

"""Log seconds since last call to `log\_time`."""

global last\_log\_time

t = time.time()

log("{:.1f}s elapsed", t - last\_log\_time)

last\_log\_time = t

def user(text, \*args, \*\*kwargs):

"""Writes progress information for the user."""

log\_text = text.format(\*args, \*\*kwargs)

print(log\_text)

log(log\_text)

def log\_exception(msg, \*args, \*\*kwargs):

"""Log a traceback."""

tb = traceback.format\_exc()

log(msg, \*args, \*\*kwargs)

log(tb)

The above code snippet displays some part of the complete code.

Unit testing to check for correct installation of Raspbian OS and to check if the device is recognised by the Dataplicity.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Technique** | **Test Description** | **Input** | **Expected Output** | **Actual Output** | **Result** |
| #T1 | General | Checking power supply to the Raspberry Pi | Power Supply | Red light Indicator glows | Red light Indicator glows | Pass |
| #T2 | General | To check whether the SD card is detected by the Raspberry Pi | SD card | Booting of Pi should start | Booting was initiated | Pass |
| #T3 | General | To check whether the Raspbian OS is installed Successfully | SD card | Dashboard will appear | Dashboard appeared | Pass |
| #T4 | General | If the user registration is successful | User's credentials | An E-mail will be sent to the user | Email was received | Pass |
| #T5 | General | If the user registration is not successful | User's credentials | An error message will pop up stating to enter valid email | Error message appeared | Pass |
| #T6 | General | To check if the device is recognized by the Dataplicity | Click on the link says list the devices | Device will be listed under the connected devices list | Device was successfully listed | Pass |

Guide Project Coordinator HOD-ISE

(Signature with Date) (Signature with Date) (Signature with Date)