

PÉCSI TUDOMÁNY EGYETEM
TERMÉSZETTUDOMÁNYI KAR
MATEMATIKAI ÉS INFORMATIKAI INTÉZET

R projekt (Studio, Shiny) telepítése
és alkalmazása a Raspberry Pi
tenyészszámítógépen



Témavezető: Prof. RAJKÓ ÁRPÁD RÓBERT
Egyetemi Tanár

Készítette: VO TRUNG HIEU (NEIABP)
Programtervező informatikus BSc

PÉCS, 2021

TARTALOMJEGYZÉK

I. Összefoglalás/absztrakt	3
II. Bevezetés és célkitűzés	4
1. Bevezetés	4
2. Célkitűzés	4
III. Eszközök/anyagok és módszertana bemutatása	5
1. Eszközök	5
2. A Raspberry Pi-ről.....	5
2.1. Általános információ	5
2.2. Részletek a RPi modellemről.....	6
2.3. RPi ház építése ventilátorral	10
3. Az Enzimkinetika vizsgálatáról	12
3.1. Enzim kinetika	12
3.2. Egyszubsztrát reakciók	13
3.3. Michaelis–Menten kinetika.....	13
IV. A projekt folyamata.....	14
1. OS Telepítés.....	14
2. Virtual Network Computing - ha nem Headless használat	15
3. Headless SSH-vel	16
4. Az RStudio és Shiny Server telepítése	17
4.1. RPi előkészítése:	17
4.2. Az Ansible telepítése	17
4.3. A „Playbooks” letöltése és konfigurálása	18
4.4. A „Playbooks” futása	19
4.5. 502-es hiba	20
5. Port Forwarding.....	20
V. Eddigi talált problémák és megoldásuk	22
1. Megismertem egy új módot – „Headless”	22
2. Headless konfigurációk – Wifi és SSH kulcspár	22
2.1. Wifi konfiguráció	22
2.2. SSH kulcspár	24
3. Újabb wifi incidens	24

VI. Biztonsági intézkedések.....	26
1. 5 ajánlott eljárás.....	26
1.1. Az alapértelmezett felhasználónév és jelszó módosítása	26
1.2. Állítsa be a sudo-t, hogy jelszót kérjen.....	27
1.3. Az SSH biztonság javítása	27
1.4. Telepítse az Uncomplicated Firewall-t	27
1.5. A Fail2ban engedélyezése	27
2. Részletesebben az eljárások.....	28
2.1. Superuser (SU).....	28
2.2. Felhasználók és SU konfigurációjának módosítások	29
2.3. SSH jelszó-nélküli bejelentkezés.....	29
2.4. UFW konfigurációk	30
2.5. Fail2ban konfigurációk	31
VII. Részletesen az RStudióról és a Shinyről	33
1. Az RStudio Server	33
1.1. Telepítése - Ansible Playbooks	33
1.2. A munkaterülete.....	34
2. A Shiny App – Pi Michaelis-Menten	35
VIII. Eredmény fejezet:	37
1. Észrevételek	37
2. Értekezés / Megbeszélés	39
IX. Következtetések.....	40
X. Hivatkozási lista	41
XI. Rövidítések lista	43
XII. Nyilatkozat az írásmű eredetiségéről.....	44

I. Összefoglalás/absztrakt

Ebben a dolgozatban, azt vizsgálom, hogy (1) alkalmas-e egy Raspberry Pi (RPi), egylapkás számítógép használatával, az RStudio fejlesztői környezetben egy webalkalmazást (WebApp)-ot működtetni, amely képes valós időben kiszámítani összetett matematikai formulákat, ez konkrét esetünkben: az enzimkinetikában használatos Michaelis-Menten modellt; valamint, (2) az elkészült WebApp, amely innentől Pi Michaelis-Menten (PMM)-nek nevezem, a Shiny Server keretrendszer segítségével grafikus felületét biztosítani helyi (LAN) és nagyterjedésű (WAN) hálózaton.

A PMM már egy-két hónap fejlesztés után probléma ill. megállás nélkül futott. A folyamat során a Raspberry Pi következetesen a maximális kapacitásának csak körülbelül 30%-án működött, amely felülmúlva várakozásaimat, így lehetőséget biztosít arra, hogy az RPi-t más, sokkal összetettebb feladatokra is alkalmazzassuk. A jövőben szeretném bővíteni más webalkalmazásokkal a jelenlegi repertoárt az R projekt közössége által kínált bonyolultabb felhasználásokra is.

II. Bevezetés és célkitűzés

1. Bevezetés

A világ rohamosan halad a digitális tér felé, ezzel egyidőben szükségessé vált a szerverek és az adatbázisok fejlesztése; a Raspberry Pi nagyon jól kezeli ezt a két feladatot.

Azt javasoltam, hogy a RPi az egylapú számítógép egy megfizethető, kényelmesen használható megoldás webes alkalmazások tárolására. A RPi képességeinek bemutatására kiválasztottam és megépítettem a PMM-nevű, R nyelven írt webalkalmazást, amely grafikus képeken keresztül, Michaelis-Menten modellt segítségével mutatja be az enzim kinetika értékváltozásait.

2. Célkitűzés

A dolgozat fő célja, hogy megfizethető és kényelmes megoldást nyújtson web hostolás és megosztási projektekhez.

Az első kutatási cél az volt, hogy a Raspberry Pi elégséges-e az összetett matematikai képletek valós idejű számításához, esetünkben a Michaelis-Menten modellhez.

A második kutatási cél az, hogy építettem egy olyan felhasználóbarát felület, amely lehetővé teszi a felhasználók számára, hogy módosítsák az említett Michaelis-Menten modellt bemeneti értékeit, ami megváltoztatja az egyenlet kiszámított eredményeit, majd ezeket az eredményeket grafikonokon mutatják be, így megkönnyítené számunkra a különbségek észrevételét.

III. Eszközök/anyagok és módszertana bemutatása

1. Eszközök

Helyi számítógép	Laptop Asus ROG GL552VX - Intel i7-6. generációs chip, Nvidia 950M grafikus kártya - OS: dual-booting - Windows 10 + Linux Manjaro (LTS 5.13)
Távoli számítógép	Raspberry Pi Model 4B - OS: Raspberry Pi OS (korábban Raspbian nevű) - Memória: 4 GB - Tárolás: SanDisk Extreme microSDHC UHS-I 32 GB 160- és 100 MB/s az olvasási és írási sebességhez. - Kis 5V hűtési ventilátorral
Hálózat	Kábel nélküli - Wifi modem: Jelenleg használt: véletlenszerűen ingyenes, Digi ISP-ről - Huawei EchoLife HG8143A5 – optikai A múltban (a szakdolgozat első fele): - Mercusys AC1200 Dual GIGAbit router – rézkábel
	Kábeles: Ethernet CAT5, 2 méteres
IDE	RStudio Server
Keretrendszer	Shiny
DDNS szolgáltatás	NoIP

2. A Raspberry Pi-ről

2.1. Általános információ

A Raspberry Pi néven ismert minimálszámítógép az angliai Cambridge Egyetem négy professzorának találmánya. A négy professzor a következő volt: Eben Upton, Rob Mullins, Jack Lang és Alan Mycroft. Azt vették észre, hogy egyre csökken az A-szintű hallgatók készsége és száma, akik CS (Computer

Science / Informatika) akartak tanulni. A négy férfi úgy gondolta, hogy ennek azaz oka, hogy az otthoni számítógépek megdrágultak, és ezért a szülők nem engedték, hogy gyermekeik kísérletezzenek velük. Így olyan megoldást kerestek, amely lehetővé teszi a fiatalabb generáció számára, hogy otthon is kísérletezzon a számítógépekkel. [1] [2]

2006-ban ez a négy férfi elkezdte tervezni azt, amely később a legelső Raspberry Pi lett. Pete Lomasszal, a "Norcott Technologies" hardvertervezési és -gyártási igazgatójával és David Brabennel, a "BBC Micro Game Elite" társszerzőjével együtt létrehozták a „The Raspberry Pi Foundation” nevű oktatási alapítványt[1]. 2011-re a Raspberry Pi első verziója elérhető volt a nyilvánosság számára, és az első két évben több mint 2 millió darabot adtak el. Az alacsony költségű termék létrehozása érdekében úgy döntöttek, hogy ARM processzorokat használnak a Raspberry Pi-hez az x86-os változat helyett[3].

2.2. Részletek a RPi modellemről

Megjegyzés: Tartsa az RPi-t úgy, hogy az USB- és Ethernet-portok jobbra nézzenek, a HDMI-port lefelé nézzen, és az elülső oldala (azaz oldal, ahol az USB portok láthatók) nézzen Ön felé [felülnézet] [4].

2.2.1. A projectnek fontos alkatrészek

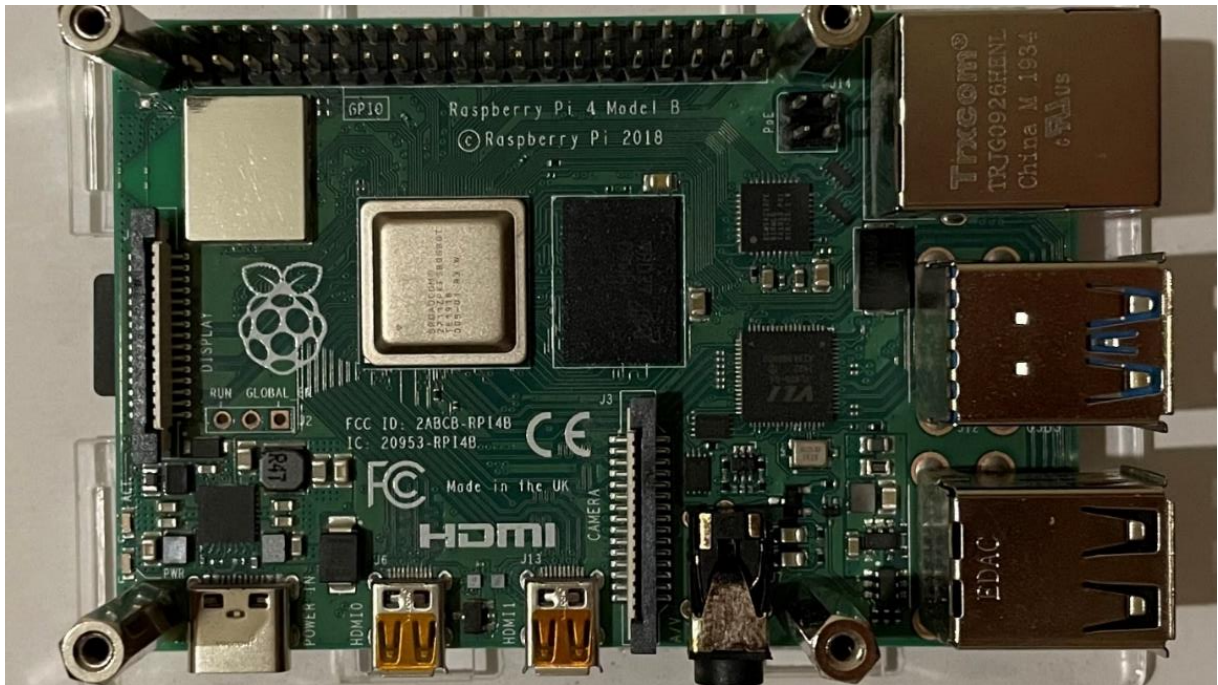
- USB-C port a tápellátáshoz [baloldalinézet], a legelső port a bal alsó sarokban helyezik, 3 és 5 Volt DC közötti tápfeszültséggel működik, járva adapterével.
„Egy jó minőségű 2,5 A-es tápegységgel használható, ha a lefelé irányuló USB-perifériák összesen kevesebb mint 500 mA-t fogyasztanak.” [4] De ez lényegtelen volt az esetben, mivel „Headless mode” használok az RPi-t, amúgy nem túl szükségesek az USB portok.
- Micro-SD kártyahely [alulnézet], az RPi hátoldalának balra található, kezeli az operációs rendszer betöltését és az adattárolást.
- Szabványos, ismert 40 GPIO (general purpose input/output) tűje[5]: a felül oldalvonalon fekszenek, ezek a tűk balról-jobbra alulról-felülre

számozva. Meddig még csak a 4-es és 6-os (a második és harmadik sor másod tűje) tűket használtam ventilátor tápellátásáért.

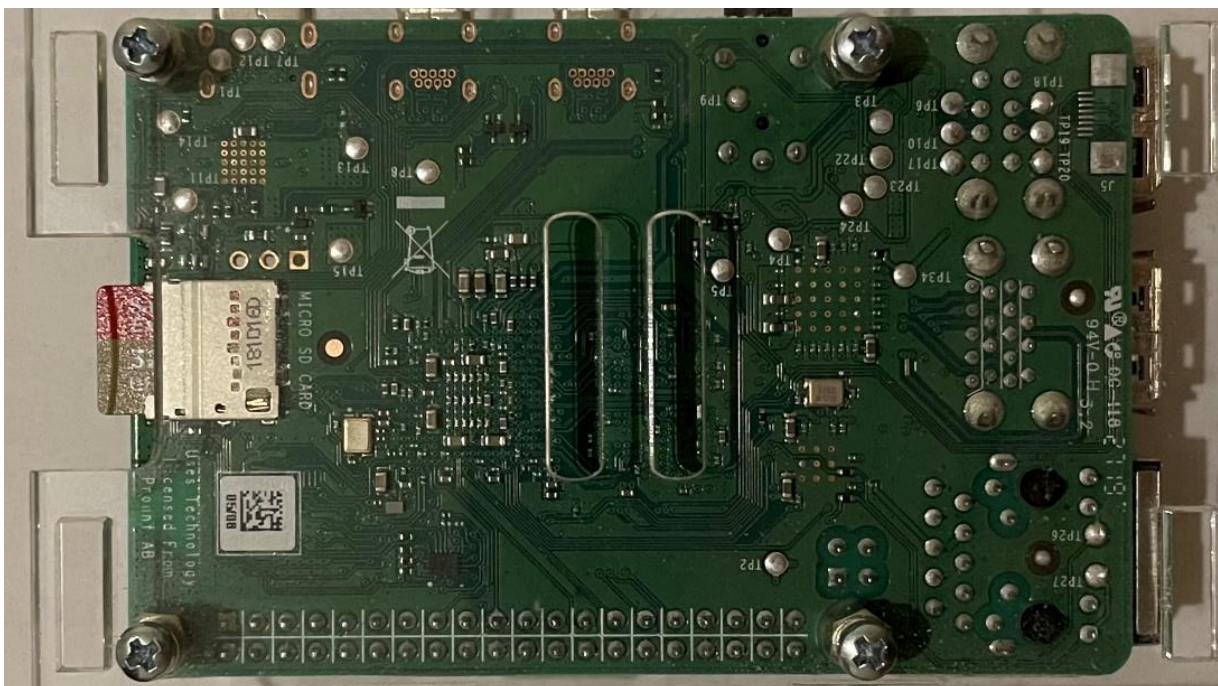
- A 4-es tű, azaz a RPi egyik +5V tűk, a 6-os tű pedig egy [GND](#) - földelt tű. A többi felhasználható kimeneti teljesítmény tűk az 1-es és 17-es, amelyek 3V-os, ha ezekkel használ egy ventilátor akkor lassabb pörgést lehetnie. Egyéb észrevehető tűk: 2-es a másik 5V tű, a 14-edik sorú tűke azaz ID EEPROM tűk haladó használatának csak. A megmaradt tűk földelt vagy általános célúak.
- Többi információ: <https://www.raspberrypi.com/documentation/computers/os.html#gpio-and-the-40-pin-header>
- Gigabit Ethernet port [[portoknézet](#)] az utolsó fontos alkatrész, 1 méteres CAT5e rézkábellet párosítom néhánykor.
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless.

2.2.2. *Az egyéb alkatrészek*

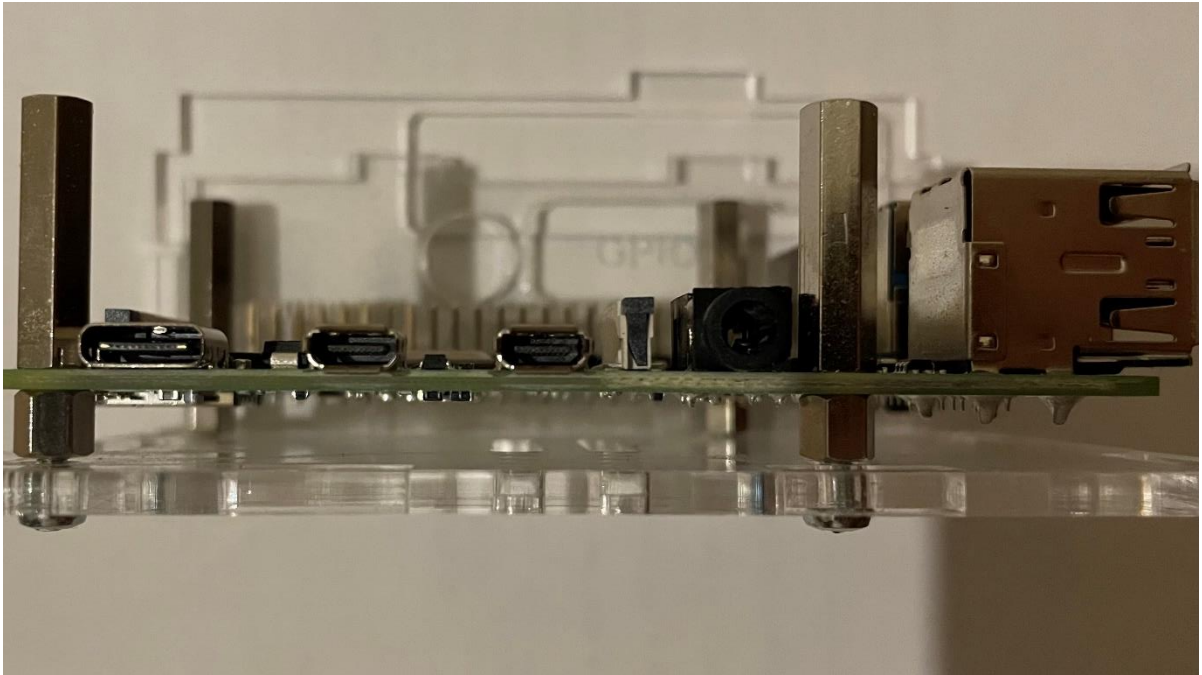
- 2 x mikro-HDMI port [[baloldalnézet](#)] a tápcsatlakozó mellette található, támogatható akár 4k60p-ig.
- 2 x USB2 port, 2 x USB3 port, az elülső arc alján, a fekete az USB2-es, a kék pedig USB3-as. [[portoknézet](#)]
- 4 pólusú sztereó audió és kompozit videó port, a kerek port a mikro-HDMI portok közelében
- Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC@ 1.5GHz chip, azaz a legnagyobb megjelenésű négyzet darab a RPi-n.
- 4GB LPDDR4-3200 SDRAM, a chip mellette fekszik.
- 2-lane MIPI DSI kijelző port, a bal teteje részén található.
- 2-lane MIPI CSI camera port, a második HDMI és AV portok között



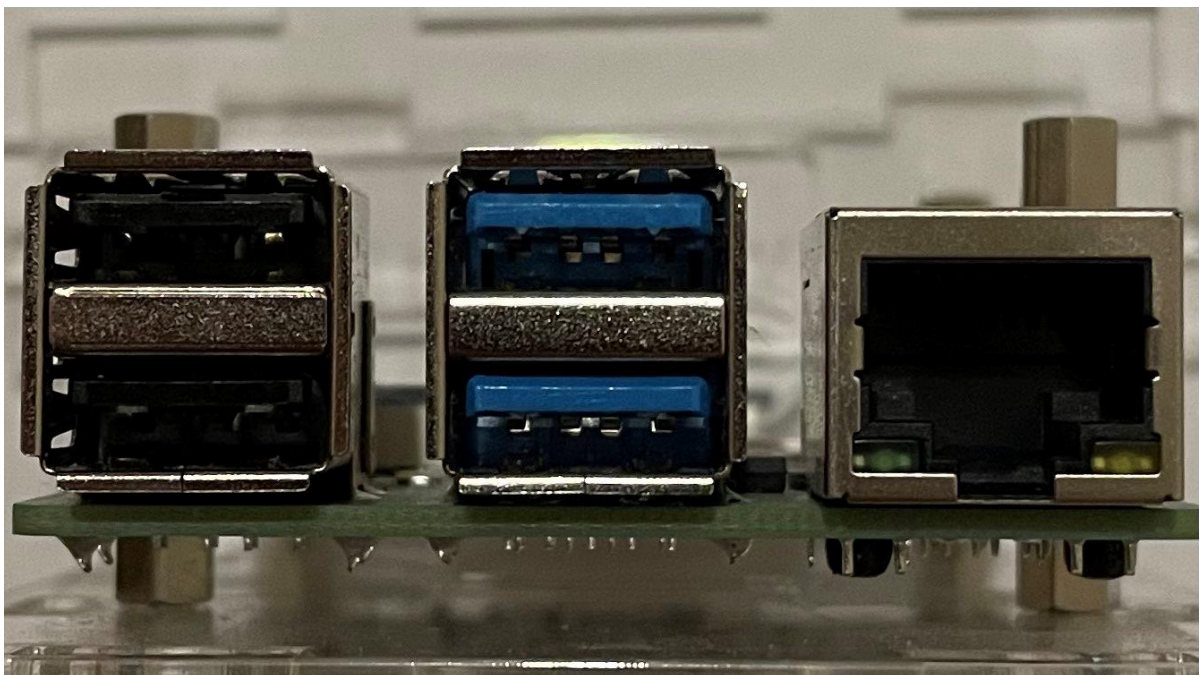
Felülnézet



Alulnézet



Baloldalinézet



Portok nézet

2.3. RPiház építése ventilátorral

Bár ez a rész nem a dolgozat egyik fő pontjai, a hűtés mégis nagyon fontos egy szerver számára. Az RPi-nek sok dedikált ház-kialakítása van, az enyém egy kis [műanyag doboz ventilátorral](#).

Az 5V-os ventilátor a Raspberry Pi GPIO tűit használja működésért, az enyém a 4-es és 6-os tűkhöz csatlakozik. Mivel a tok nagyon nyitott, nagy légáramlást (high-airflow módszer) szem előtt tartva állították össze. A kérdés csak az volt, hogy a ventilátor milyen irányba fúj, ezért futtattam néhány tesztet, hogy meghatározzam. A teszteket a következőképpen végezték:

A teszt többszörös futtatásból áll különböző változókkal, amelyek a ventilátor esetében levegő befújása vagy kiszívása; állás nélküli eltérő futási idő, azaz 4, 6 és 12 órán belül, köztük leállásokkal, hogy a gép lehűljön és minden futtatáshoz ezt a parancsot használtam:

```
stress-ng --cpu 0 --cpu-method fft -t {x} h && temp
```

abból a „--cpu 0” opció mind a 4 magon való teszteléshez, „--cpu-method fft”, amely 4096 minta gyors Fourier transzformáció, „-t {x} h”-óraban a futási idő.

Az „temp” parancs pedig, egy általam beállított álnév (alias), ami alapvetően az „vncgencmd measure_temp”[6], amely egy beépített parancs a CPU aktuális hőmérsékletének kinyomtatására. Az „&&” azt jelenti, hogy csak az előző parancs sikeres befejezése után futtassa le a mögötte lévő parancsot.

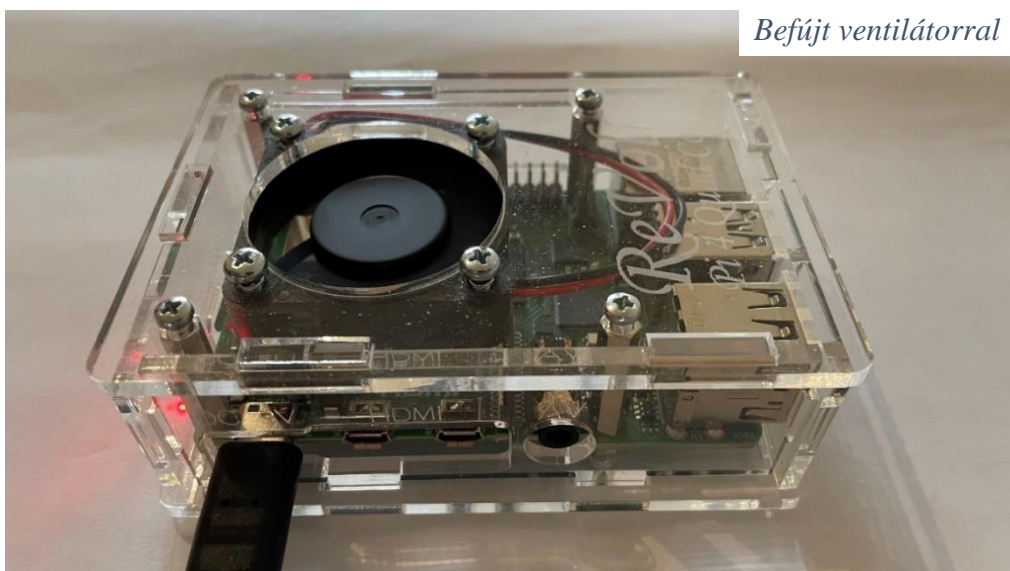
A hőmérsékletet üresjáratú állapotban méri, és minden futási idő után újra méri. Minden futtatás után kitöltöttem ezt a táblázatot a futótesztből kapott adatokkal, a hőmérsékletet Celsiusként mértük:

	4h	6h	12h
Befújása	55 – 57	57 – 58	60
Kiszívása	57 – 58	60 – 61	60 – 62

Amint látjuk, a CPU hőmérséklete jobb volt amikor a [ventilátor befűjt](#). Ezzel az eredménnyel futok, amióta megkaptam [az RPi](#)-mat. Mégis azt veszem észre, hogy por elkezdett nagyon felgyűlni, ez a levegő befűtésének egyik hátránya a levegő kiszívásához képest. Lefedhetem a ventilátort porszűrővel, de ez rosszabb teljesítményt eredményez, mert a szűrő blokkolja a ventilátorból befűjt levegő. Lehet, hogy még annál is rosszabb, mint a levegő kiszívásával futni, ami több tesztelést igényel, ezért úgy döntöttem, hogy hagyom csak.



Frissen érkezett RPi



Befűjt ventilátorral

3. Az Enzimkinetika vizsgálatáról

3.1. Enzim kinetika

Az enzimkinetika az enzimek által katalizált kémiai reakciók sebességének vizsgálata. Az enzimkinetikában a reakció sebességét mérik, és a reakció körülményeinek változtatásának hatásait vizsgálják[7]. Egy enzim kinetikájának ilyen módon történő tanulmányozása feltárhatja ennek az enzimnek a katalitikus mechanizmusát, az anyagcserében betöltött szerepét, az aktivitás szabályozásának módját, és azt, hogy egy gyógyszer vagy egy módosító (inhibitor vagy aktivátor) hogyan befolyásolhatja az enzim sebességét.

Az enzim (E) tipikusan olyan fehérjemolekula, amely elősegíti egy másik molekula, annak szubsztrátja (S) reakcióját. Ez az enzim aktív helyéhez kötődik, ES enzim-szubsztrátum komplexet hozva létre, és az ES* átmeneti állapoton keresztül enzim-termék komplex EP-vé, majd onnan P terméké alakul. A lépések sorozata mechanizmusként ismert[8]:



Ez a példa egy szubsztrátummal és egy termékkel való reakció legegyszerűbb esetét feltételezi. Az izomeráz egy általánosabb kifejezés egy olyan enzimre, amely bármely egy szubsztrátum egy termék reakcióját katalizálja, mint például a trioszfoszfát izomeráz. Az ilyen enzimek azonban nem túl gyakoriak, és jelentősen felülmúlják azokat az enzimeket, amelyek két szubsztrátum és két termék reakcióit katalizálják. Három vagy szubsztrátummal, vagy termékkel való reakciók ritkábban fordulnak elő, de léteznek.

Amikor az enzimek több szubsztrátumhoz kötődnek, az enzimkinetika azt is megmutathatja, hogy ezek a szubsztrátumok milyen sorrendben kötődnek, és milyen sorrendben szabadulnak fel a termékek. Általában van egy sebesség-meghatározó lépés, amely meghatározza az általános kinetikát. Ez a sebesség-meghatározó lépés[9] lehet kémiai reakció vagy az enzim vagy szubsztrátumok konformációs változása, például azok, amelyek részt vesznek a termék(ek)nek az enzimből való felszabadulásában.

3.2. Egyszubsztrát reakciók

Az egyszubsztrát mechanizmussal rendelkező enzimek közé tartoznak az izomerázok, például a triózfoszfát-izomeráz vagy a biszfoszfoglicerát-mutáz, az intramolekuláris liázok, például az adenilát-cikláz és a kalapácsfejű ribozim, egy RNS-liáz.[11]

Az egyetlen szubsztrátumot érintő biokémiai reakciókról gyakran feltételezik, hogy a Michaelis–Menten-kinetikát követik, tekintet nélkül a modell alapfeltevéseire.

3.3. Michaelis–Menten kinetika

A biokémiában a Michaelis–Menten-kinetika az egyik legismertebb enzimkinetikai modell.[12][13] Nevét Leonor Michaelis német biokémikusról és Maud Menten kanadai orvosról kapta.[14] A modell az enzimreakciók sebességét leíró egyenlet formáját ölti úgy, hogy a v reakciósebességet (termék képződési sebesség, $[P]$) összefüggésbe hozza az S szubsztrát koncentrációjával $[S]$ -vel. Képletét a következőképpen adja meg:

$$v = \frac{d[P]}{dt} = V_{max} \frac{[S]}{K_M + [S]}$$

Ezt az egyenletet Michaelis-Menten egyenletnek nevezik. Itt a V_{max} a rendszer által elért maximális sebességet jelenti, amely egy adott enzimkoncentrációhoz telítő szubsztrátkoncentrációnál történik. A K_M Michaelis-állandó értéke számszerűen megegyezik azzal a szubsztrátum-koncentrációval, amelynél a reakciósebesség a V_{max} fele.[15]

IV. A projekt folyamata

1. OS Telepítés

Raspbian ISO letöltés: <https://www.Raspberrypi.org/downloads/>

"Raspberry Pi Imager" használásával írva az OS-t az SD-kártyára volt egy "Advanced" mód, ezzel a kulcskombóval érheti el: Ctrl+Shift+X, ez lesz egyszerűbb módszer hogyha Headless módot szeretné használni („Headless” olyan módja amikor az elektromos vezeték kívül semmi sem kapcsolódik az RPi-hez, így lehetővé teszi az RPi vezérlését a jelenlegi gépével, pl. Laptop[16])

Felkészülés a Headless módra: az OS-t írás után: Adjon hozzá egy üres "ssh" (secure shell protocol) nevű, kiterjesztése nélküli fájlt, az SD-n belüli "bootup" mappába, így az SSH kapcsolat engedélyezéséhez.

Wifi előre konfigurálása: Ha ezt nem teszi meg, akkor az Ethernet kábel használata lehetővé teszi az SSH-t is.

Hozzáadja egy "wpa_supplicant.conf"[17] fájlt az SD belüli "bootup" mappába, és ez a fájl tartalma a következő legyen:

```
ctrl_interface=DIR=/var/run/wpa_supplicant
# A fájl teljes elérési útja

GROUP = netdev

update_config = 1

network = {
    ssid = "[your_wifi_name]"
    psk = "[wifi_password]"
    key_mgmt = WPA-PSK
    # adattitkosítás specifikáció
}
```

*** Használat előtt jó szokás frissíteni**

```
$ sudo apt update && sudo apt upgrade
```

2. Virtual Network Computing [18] - ha nem Headless használat

VNC telepítése RPi-n

```
$ sudo apt install realvnc-vnc-server realvnc-vnc-viewer
```

VNC szerver engedélyezése a commandline/parancssorban

```
$ sudo raspi-config
```

Navigáljon az "Interfacing Options"-hoz

Görögessen le, és válassza a VNC>Yes

vagy

```
$ sudo systemctl start vncserver-virtuald.service
```

Automatikus indítás / Auto-bootup

```
$ sudo systemctl enable vncserver-virtuald.service
```

Telepítse a VNC Server és Viewer a területi számítógépére

Közvetlenül:

Írja be az RPi IP-címét a VNC Viewerbe

Bejelentkezés pi és a [jelszó] nevén

A VNC ezután automatikusan elmenti a kapcsolat adatait a következő használatra

Felhős:

Regisztráljon ingyenesen itt:

<https://www.realvnc.com/en/raspberrypi/#sign-up>

Jelentkezzen be hitelesítő adataival a VNC Serverbe mindkét eszközön (területi és távoli gépek)

Kattintson a jobb felső sarokban való hamburgerre

"Licensing": Email / Password

3. Headless SSH-vel

SSH: \$ ssh pi@[RPi_IP]

Az [RPi_IP] az router irányítópultján található

Vagy ha egy Unix OS-on dolgozik akkor

```
$ ping raspberrypi
# "raspberrypi" azaz alapértelmezett neve, módosítsa a
raspberrypi-t, ha módosította az 'etc/hostname' fájlt
```

vagy nmap használva szkennelne a hálózatot

```
$ nmap [subnet_IP]/24
# "$ hostname -i" ha nem tudja a subnet IP-t
```

Jelszó nélküli SSH: A területi gépen

Meglévő SSH-kulcsok keresése

```
$ ls ~/.ssh
```

Az id_rsa fájl az privát kulcsa, tartsa ezt a számítógépen.
Az id_rsa.pub pedig, amit oszt meg azokkal a gépekkel,
amelyekhez csatlakozni szeretne.

Ha léteztettek "id_rsa.pub" és "id_rsa" fájlok, akkor nem kell a
következő lépés.

Új SSH-kulcspár generálása

```
$ ssh-keygen
```

A parancsfolyamat után, látnia kell az id_rsa kulcspárt az
.ssh mappában

Másolja a kulcsát a Raspberry Pi-be

```
$ ssh-copy-id -i ~/.ssh/id_rsa.pub [user_on_pi]@[RPi_IP]
```

Másik módszer: másolja fájlokat, mappákat SSH-on Pi-ra

SCP - Secure copy használata

```
$ scp [path_to_file] [user]@[RPi_IP]
```

Ha több fájlt másolás, akkor szóközzel elválasztva a fájlok

Ha mappát szerint másolni, használja meg a -r kapcsolót

4. Az RStudio és Shiny Server telepítése

Óriási köszönet Andres-nak: Andres's guide

https://andresrcs.rbind.io/2021/01/13/raspberry_pi_server/

4.1. RPi előkészítése:

```
$ sudo raspi-config
```

SSH engedélyezése

Csökkentse a GPU-jegyzetet 16MB-ra

Bővítse ki a teljes SD-kapacitás kihasználásához

A "Predictable Network Interface" letiltása

```
$ sudo reboot now
```

Statikus IP konfigurálása: nagyon fontos, Ethernet-kábel használata javasolt

```
$ sudo nano /etc/dhcpd.conf
```

```
# keress a "static ip_address" kifejezéssel sort
```

4.2. Az Ansible telepítése

***Következő lépések a területi/helyi gépen lesznek**

```
$ sudo pacman -S python3 python3-pip
```

```
$ sudo pip install ansible
```

Az **SSH inaktivitásának megelőzése**: mert az RStudio telepítése sokáig tart (első alkalommal az egész éjszakába telt, későbbi próbálkozásokkal pedig 4-6 órát)

```
$ sudo nano /etc/ssh/ssh_config
```

A "# Host *" alá add hozzá:

```
ServerAliveInterval 300
```

```
ServerAliveCountMax 2
```

4.3. A „Playbooks” letöltése és konfigurálása

4.3.1. *Letöltése*

```
$ sudo pacman -S git
```

```
$ git clone https://github.com/andresrcs/raspberry_pi_server.git --depth 1
```

A parancsfolyamat után, látnia meg egy `./raspberr_pi_server` nevű mappát

4.3.2. *Konfigurálása*

Adja hozzá a Pi [RPi_statikus_IP] # amit írta az `'/etc/dhcpd.conf'`-ban, így nézik a `'./raspberry_pi_server/inventory.ini'` fájl tartalmát:

```
[raspberrries]
```

```
raspberry_01 ansible_host=[RPi_statikus_IP]
```

Állítsa be a [raspberrries] csoport általános változóit, amelyek az `'./raspberry_pi_server/group_vars/raspberries.yml'` fájlban vannak:

```
ansible_user: 'pi'
```

```
# az RPi-nak alapértelmezett felhasználója
```

```
ansiber_become_method: su
```

```
ansible_python_interpreter: /usr/bin/python3
```

```
ansible_ssh_private_key_file: [full_path_to_RPi_private_key]
```

Módosítsa az alapértelmezett beállításokat a

`'./raspberry_pi_server/vars/config_vars.yml'` fájlban vannak:

```
fail2ban_email: [youremail]
```

```
postgres_password: '[very_random_password]'
```

```
# és ha nem Ethernetet használnia
```

```
disable_wifi: false
```

```
### volt már gondom ezzel korábban, és sokáig tartott, amíg
```

```
rájöttem, hogy mi történt, "miért nincs internete az RPi??" # és a
```

```
wifi előkonfigurálása a 'wpa_supplicant.conf' fájlal is nagyon
```

```
problémás volt.
```

4.4. A „Playbooks” futása

```
$ cd [path_to_this]/raspberry_pi_server/
```

Innentől kezdve futtathatja a teljes csomagot

```
$ ansible-playbook main.yml
```

***vagy inkább 3 lépésben**, lépésről lépésre célszerű, mert ha valami elromlik, akkor könnyebben azonosítható a probléma

```
$ ansible-playbook install_basic_services.yml
```

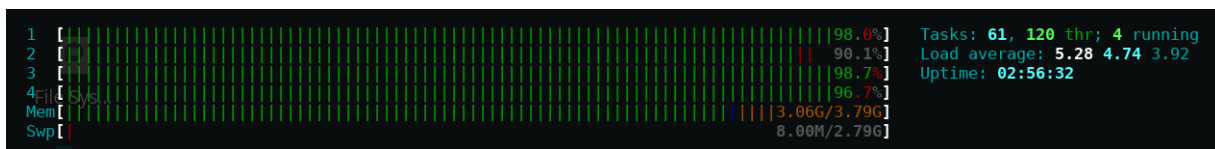
FIGYELEM: ha nem változtatta meg a „disable_wifi” pár lépések ezelőtt, ezaz parancsfolyamat után a Wifi letiltva lesz, itt még akadályozhatja meg a Wifi letiltását, a „--skip-tags” kapcsoló/opció használatával:

```
$ ansible-playbook install_basic_services.yml --skip-tags "disable_wifi"
```

```
$ ansible-playbook install_shiny_server.yml
```

```
$ ansible-playbook install_rstudio_server.yml
```

```
# ez önmagában 3 órát végre hajtott
```



```
1 [|||||] 98.0% Tasks: 61, 120 thr; 4 running
2 [|||||] 90.1% Load average: 5.28 4.74 3.92
3 [|||||] 98.7% Uptime: 02:56:32
4 [|||||] 96.7%
FileSys[|||||] 3.06G/3.79G
Mem[|||||] 8.00M/2.79G
Swp[|||||]
```

Az összes “Playbooks” sikeres futtatása után egy teljesen működőképes telepítés készen áll a használatra, így egyszerűen megnyithat egy RStudio munkamenetet a [http://\[RPi_IP\]/rstudio/](http://[RPi_IP]/rstudio/) és/vagy tegye közzé Shiny alkalmazásait az [/srv/shiny-server/](http://[RPi_IP]/shiny-server/) mappába, és érje el őket ezen a címen:

```
http://[RPi_IP]/shiny/[app_name]
```

4.5. 502-es hiba

Ha 502-es hibát talál, próbálja meg:

```
$ rstudio-server status # hogy fut-e a kiszolgáltatás
```

Ha inaktív, akkor:

```
$ rstudio-server stop
```

```
$ rstudio-server start
```

```
$ rstudio-server status  
# ellenőrizze újra, hogy most fut-e
```

További alkalommal, elég csak ezt a parancsot használni

```
$ rstudio-server start
```

Vagy ennek használata ugyanazt az eredményt kap

```
$ sudo systemctl status rstudio-server.service
```

```
$ sudo systemctl stop rstudio-server.service
```

```
$ sudo systemctl start rstudio-server.service
```

A következőket automatikus indítás érdekében

```
$ sudo systemctl enable rstudio-server.service
```

Megjegyzés: Ha a szolgáltatás inaktív, az „enable” futtatása egyedüli nem fog indítani el, vagy a „start” parancsot kell használnia, vagy újra kell indítani a Raspberry Pi-t.

5. Port Forwarding

A **Port Forwarding** gyártóként, router modellként változik, de az általános beállítások nem lesznek túl nagy különbségek.

A **DDNS** szolgáltatáshoz a NoIP-t választottam az ingyenes csomagjához, 30 naponként kell újítani meg: <https://www.noip.com/>

Az enyém beállítások a következő képeken látható:

Port Forwarding beállítások

	WAN Name	Internal Host	External Host	Enable	
<input type="checkbox"/>	ShinyServ	2_INTERNET_R_VID_921	RPi_privát_IP	--	Enable

Type: ☒ User-defined ☐ Application

Application:

Enable Port Mapping: ☒

Mapping Name:

WAN Name:

Internal Host: *

External Source IP Address:

Protocol: Internal port number: -- *

External port number: nyílt portok -- External source port number: --

A TCP/80 általánosan a HTTP port.

Az „External port number” vagy „nyílt portok” fontos lesz a NoIP beállításában, de itt véletlenszerűen dönthető.

Hostname ⓘ

Domain ⓘ

NoIP beállítások

Record Type

- ☐ DNS Host (A) ⓘ
- ☐ AAAA (IPv6) ⓘ
- ☐ DNS Alias (CNAME) ⓘ
- ☒ Web Redirect ⓘ

[Manage](#) your Round Robin, TXT, SRV and DKIM records.

Wildcard ⓘ

[Upgrade to Enhanced](#)

to enable wildcard hostnames.

Protocol

URL / IP ⓘ

☒ Mask URL ⓘ

Page Title ⓘ

Az URL/IP mezőben IP-t nyílt porttal használok, mivel a forgalmat az RPi-hez és onnan kell irányítanom, úgy kellene a router nyilvános IPv4 címét, és az előző választott nyílt portok számát, a router IPv4 a <https://whatismyipaddress.com/> weboldalon található.

Egy példa az URL/IP-re: 8.8.8.8:1234

V. Eddigi talált problémák és megoldásuk

1. Megismertem egy új módot – „Headless”

Kezdetben nem a nem tudtam a „Headless” módról, mivel nekem csak egy laptopom van és csak ezzel dolgozhatok, úgy tűnődtem, hogy közvetlenül csatlakoztathatom-e a RPi-t HDMI-kábellel a laptopom monitorára.

Utólag visszagondolva, vagy nagyon naiv voltam, mert a laptopmodellem ezt biztosan nem tudja elérni, vagy nem voltam elég képzett ahhoz, hogy ezt megvalósíthassam. Így aztán kb. egy óra kutakodás után jött az ötlet, találtam egy cikket a "Headless mode"-ról, amely alapvetően lehetővé teszi az RPi használatát anélkül, hogy bármilyen típusú bemeneti vagy kimeneti eszközt csatlakoztatna hozzá, nincs szükség külön monitorra – head-re, hogy Headless, ami szerintem elég kreatív a név.

2. Headless konfigurációk – Wifi és SSH kulcspár

2.1. Wifi konfiguráció

Először a "wpa_supplicant.conf" fájl baja volt. Megpróbáltam újról újra, de mindig sikertelenül csatlakozni a wifihez. Sok sikertelen próbálkozás után, az Ethernet-kábel használatához fordultam, hogy „legalább bejelentkezhessenek”, aztán a wifi-t a „raspi-config” paranccsal konfiguráltam.

A raspi-config csak a Raspberry Pi operációs rendszerhez használható, itt egy univerzálisabb módon (amit én is használtam egyszer, mert jó néhánykor eltörtem a RPi OS összeállítását...):

Azonosítsa a wifi bejegyzést:

```
# ip link
```

wpa_supplicant létrehozása, esetemben "wlan0" a [wifi_card_name]:

```
# touch /etc/wpa_supplicant/wpa_supplicant-[wifi_card_name].conf
```

Az előtte-létrehozott fájl szerkesztése:

```
# vi /etc/wpa_supplicant/wpa_supplicant-wlan0.conf
```

A fájl tartamaza legyen:

```
ctrl_interface=/run/wpa_supplicant  
update_config=1
```

```
# wpa_passphrase [SSID] [PSK] >>
```

```
/etc/wpa_supplicant/wpa_supplicant-wlan0.conf
```

Végül, a „wpa_supplicant-wlan0.conf” tartalma a következő lett:

```
ctrl_interface=/run/wpa_supplicant  
update_config=1  
network={  
    ssid="[SSID]"  
    #psk="[PSK]" # ez olvasható, a valódi jelszava  
    psk=[randomly_long_string_of_characters]  
}
```

Biztonsági okokból törölje a #psk sort.

Inicializálja a fájlt, és lehetnie wifi kapcsolattal:

```
# wpa_supplicant -B -i wlan0 -c /etc/wpa_supplicant-wlan0.conf
```

A korábbi wpa_supplicant.conf fájl módszer is működhet, tudom, mert egy alkalommal, amikor újra-konfiguráltam a wifit, megint átnéztem a módszert, pontosan úgy csináltam, mint korábban, de meglepetésemre ezúttal működött. Felkuncogtam, és varázslatnak tartottam...

2.2. SSH kulcspár

Utána, az SSH kulcspár küldési baja volt. Összességében nem volt olyan zavaró, mint a wifi, csak picit „trial-and-error” volt és a helyes parancsopció kitalálása során:

A területi (laptopom) gépről küldenie kell a távoli géphez (Raspberry Pi)

```
$ scp [path_to_id_rsa.pub] pi@[RPi_IP]:/home/pi/.ssh/
```

Aztán a távoli gépen, az elküldött kulcs hozzáfűzése:

```
$ echo id_rsa.pub >> ~/.ssh/authorized_keys
```

Egyéb problémák, amelyek felmerülhetnek: „Connection refused (Port 22)” amelynek jelent vagy az SSH szolgáltatás inaktív, ez könnyen futtatható így '\$ sudo systemctl start openssh' vagy az SSH-ható port nem a 22; "Public keys" hiba, azaz a fájl- és mappa engedélyek nincsenek megfelelően konfigurálva, kellett

```
$ sudo chmod 700 ~/.ssh
```

```
$ sudo chmod 600 authorized_keys id_rsa id_rsa.pub known_hosts
```

„Connection timeout” lehet, hogy a tűzfal blokkolja, és stb.

Azon problémák mindegyikével találkoztam és megoldottam.

3. Újabb wifi incidens

Ez a probléma az Ansible Playbooks konfigurálásának lépésénél jelentkezett, az előkészítés során nem figyeltem teljesen az egyes tevékenységek mögött meghúzódó okokra. A probléma az volt, hogy a wifi interfész „elveszett”, ezért követtem ezt az útmutatót: <https://raspberrypi.stackexchange.com/questions/40344/mysterious-rpi-wifi-problem-no-wireless-interfaces-found>, de javítás helyett letiltottam az Ethernet interfészt '/etc/network/interfaces', teremtettem egy teljesen új megoldandó problémát.

Mivel most ki voltam zárva a saját hardveremből, az SD kártya elérése volt az egyetlen lehetőség, és ez sem volt egyszerű, mert a Linux fájlrendszer típusai eltérnek a Windowsétól, ext4 vs NTFS, nem tudtam közvetlenül hozzáférni az SD-hez Windows 10-zel, így vagy VM-et vagy külön Linuxot kellett használni.

Ehhez a következőkre volt szükség: lehetővé tenni a Virtuális gép számára, hogy olvasni tudjon a gazdagép I/O-ról (SD-kártya); visszavonja a változtatásokat az `/etc/network/interfaces` fájlban, végül pedig megoldást kellett keresni és kijavítani bármit, ami a wifi modullal történt.

A folyamat a következő volt:

A VM olvashatósága gazdagép való SD-ról:

<https://scribles.net/accessing-sd-card-from-linux-virtualbox-guest-on-windows-host/>

Az Ethernet interfész rögzítéséről, szerencsére mentettem egy biztonsági másolatot, hogy másoltam és hozzáadtam „bak”-et (nem egy igazi fájl-bővítménye a Linuxban, egy olyan szintaxis volt, amelyet az Androidtól szoktam meg) az alapértelmezett fájlhoz, úgy csak kellett törölni a „bővítmény”-et, és a jelenleg meghibásodott fájl cseréje, elég ez a parancs használat:

```
$ cd /etc/network/
```

```
$ sudo mv interfaces.bak interfaces
```

A Wifi modul aktiválása a `/boot/config.txt` fájlban található, kommentáltam a letiltott wifi sort:

```
dtoverlay=disable-wifi
```

```
dtoverlay=disable-bt # bónusz a Bluetooth letiltás
```

VI. Biztonsági intézkedések

1. 5 ajánlott eljárás

Miután a „Port Forwarding”-ról és a VNC-ről, alapvetően nyitottunk meg a Raspberry Pi-t az internetre, a Raspbian, egy Linux Debian alapú és általában a Linux is, kínál néhány biztonsági funkciót, ezeket végig nézzük, és megbeszéljük az aktiválást, hogyan működnek. Mindezt nagyon jól dokumentálták és irányították az RPi hivatalos weboldalán[19], többnyire idéztem őket csak, hozzáfűzve néhány beírással.

1.1. Az alapértelmezett felhasználónév és jelszó módosítása

Hogy **módosítottam a bejelentkezett felhasználó jelszavát:**

```
$ passwd
```

Új felhasználó létrehozása és hozzáadása a su csoporthoz, valamint egyéb fontos engedélyek megadása:

```
$ sudo adduser [name]
```

```
$ sudo usermod -a -G adm, dialout, cdrom, sudo, audio, video,  
plugdev, games, users, input, netdev ,gpio, i2c, spi [name]
```

Miután új SU-felhasználóval rendelkezik, **törölnie kell az alapértelmezett pi-felhasználót**, a második parancs használásával a /home/pi-t is törli

```
$ sudo deluser pi
```

```
$ sudo deluser -remove-home pi
```

1.2. Állítsa be a sudo-t, hogy jelszót kérjen

```
$ sudo visudo /etc/sudoers.d/010_pi-nopasswd
```

Módosítsa a pi-t és a superuser jogosultsággal rendelkező felhasználókat

```
pi ALL=(ALL) PASSWD: ALL
```

1.3. Az SSH biztonság javítása

Vagy nagyon erős felhasználónévvel és jelszóval, vagy jelszó nélküli kapcsolattal kulcspár használatával. Ez igazán már [beállítottam](#) a jelszó-nélküli bejelentkezést.

1.4. Telepítse az Uncomplicated Firewall-t

```
$ sudo apt install ufw
```

Apache engedélyezése:

```
$ sudo ufw allow 80
```

```
$ sudo ufw allow 443
```

SSH engedélyezése egy különleges IP-ről:

```
$ sudo ufw allow from [IP_adr] port 22
```

1.5. A Fail2ban engedélyezése

Alapértelmezett konfiguráció használatával

```
$ sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

```
$ sudo nano /etc/fail2ban/jail.local
```

A „local” fájlt módosítsa csak, arra az esetre, ha sérült a fájl.

2. Részletesebben az eljárások.

2.1. Superuser (SU)

A Windows-felhasználók számára ismeretlen, a „Superuser” vagy „root user” alapvetően egy Unix rendszer „Admin” felhasználó, amelyre Unixban azt mondjuk, hogy „root” jogosultságokkal rendelkeznek.

És ami megkülönbözteti ezeket az SU-felhasználókat a normál felhasználóktól, az az a képesség, hogy hozzáférjenek az adminisztrációs műveletekhez, mint például a rendszer beállításainak módosítása, szoftver telepítése, és a legszembetűnőbb az, hogy hozzáférnek a „su” és „sudo” előtagokhoz egy terminálon.

Szóval amikor terminálban dolgozik, ha root jogosultságokkal rendelkezik, használhatja a parancsokat normál vagy root felhasználóként, például meg akarja változtatni az RPi hostname-ját, szerkesztheti azt a /etc/hostname fájlban, normálisan nano vagy vi (szövegszerkesztő) lehetővé teszi a szöveges fájl szerkesztését ezzel a paranccsal:

```
$ nano /etc/hostname
```

De nem itt, mivel az /etc/hostname kulcsfontosságú az eszköz számára, ezért a szerkesztéséhez su jogosultság szükséges, ez könnyen elérhető a terminálban, a su hozzáadásával az előző parancshoz

```
$ sudo nano /etc/hostname
```

Írja be a jelszavát a su jogosultságokhoz, és a szokásos módon szerkesztheti még a rendszer legfontosabb fájljait is. Így a „sudo” lehetővé teszi, hogy egyszer futtatni parancsot superuser-ként.

A másik előtag, a sima - nincs mögötte “su” (switching user), alapértelmezés szerint a "felhasználó váltása" root-ra. A „su” parancs után végrehajtott minden akciót root felhasználóként lesz végrehajtva, **ezért nagyon óvatossá kell lennie ennek használatakor**. „# exit” parancs használása visszatérjen az előző felhasználóra.

2.2. Felhasználók és SU konfigurációjának módosítások

Mivel népszerű az RPi, amivel több ember megtudja az alapértelmezett felhasználóját. Ezek az ajánlott akciók nagyon egyszerűek és bár alapvetőek, valamiért, ami könnyen megvalósítható, még hatásosak ellen a gyakorlatlan emberek, valamiért, ami könnyen megvalósítható.

2.3. SSH jelszó-nélküli bejelentkezés

2.3.1. *Definíció*

Az SSH, más néven „Secure Shell” vagy „Secure Socket Shell”, egy hálózati protokoll, amely biztonságos módot ad a felhasználóknak, különösen a rendszergazdáknak, hogy nem biztonságos hálózaton keresztül hozzáférjenek a számítógéphez. A SSH erős jelszavas hitelesítést és nyilvános kulcsú hitelesítést, valamint titkosított adatkommunikációt biztosít két nyílt hálózaton, például az interneten keresztül csatlakozó számítógép között.[\[20\]](#)[\[21\]](#)

A jelenlegi verzióban az SSH2 továbbra is erős, és bizonyítja a biztonságát, amely immár 14 éves.

2.3.2. *Hogyan működik*

Az SSH protokoll a kliens-szerver modellen alapul. Ezért az SSH-kliens SSH-munkamenetet kell kezdeményeznie egy SSH-szerver. A kapcsolat beállításának nagy részét az SSH-kliens végzi. Nyilvános kulcsú titkosítást használnak az SSH-szerver azonosságának ellenőrzésére, majd szimmetrikus kulcsú titkosítási és hash-algoritmusokat használnak az adatátvitel titkosított szövegben történő fenntartására. Így biztosítva van a magánélet és az adatátvitel integritása mindkét irányban a kliens és a szerver között, és mérséklődnek a köztes támadások. [\[22\]](#)[\[23\]](#)

Az SSH-munkamenet létrehozásának lépései a következők:

- Az ügyfél kapcsolatba lép a szerverrel a kapcsolat kezdeményezéséhez.

- A szerver egy nyilvános titkosítási kulcs elküldésével válaszol.
- A szerver egyezteti a paramétereket, és biztonságos csatornát nyit az ügyfél számára.
- A felhasználó a kliensén keresztül bejelentkezik a szerverre.

Ennek köszönhetően, ha az RPi-hez való bejelentkezési adatait ellopták, és tudtommal a távoli géphez való előzetes hozzáférés nélkül nem küldhetsz neki nyilvános kulcsot, akkor az RPi nem fogadja el a kapcsolatot egy ismeretlen, kulcspárosítás nélküli gépről.

2.4. UFW konfigurációk

2.4.1. *Definíció*

Az Uncomplicated Firewall (UFW) az iptables frontendje, és különösen jól illeszkedik a gazdagép alapú tűzfalakhoz. Az UFW keretet biztosít a netfilter kezeléséhez, valamint egy parancssori felületet a tűzfal kezeléséhez. Az UFW célja, hogy könnyen használható felületet biztosítson azoknak, akik nem ismerik a tűzfal fogalmakat, ugyanakkor leegyszerűsíti a bonyolult iptables parancsokat, hogy segítsen egy rendszergazdának, aki tudja, mit csinál.[\[24\]](#)[\[25\]](#)

2.4.2. *Konfiguráció*

Abból, amit csináltam, ami alapvetően az, hogy bizonyos portokon keresztül engedélyezze a kapcsolatot

```
$ sudo ufw allow 80
```

```
$ sudo ufw allow from [IP_adr] port 22
```

***Győződjön meg róla, hogy az ssh engedélyezett, különben ki lesz zárva a Headless RPi-ből**

```
$ sudo ufw status
```

```
$ sudo ufw enable
```

Itt található egy hivatkozás az Ubuntu hivatalos webhelyére további részletekért: <https://wiki.ubuntu.com/UncomplicatedFirewall>

2.5. Fail2ban konfigurációk

2.5.1. *Definíció*

A Fail2ban, amely egy behatolásgátló szoftver keretrendszer, nemcsak a lehetséges betörési kísérletek, különösen a „brute-force” azonosításának automatizált módját kínálja, hanem gyorsan és egyszerűen, a felhasználó által definiálható módon történő fellépésre is.[26]

2.5.2. *Hogyan működik*

A Fail2Ban ellenőrzi a naplófájlokat, mint például a [file:///var/log/pwdfail pwdfail], és letiltja azokat az IP-címeket, amelyek túl sok jelszóhibát okoznak. Frissíti a tűzfalszabályokat, hogy elutasítsa az IP-címet. Ezeket a szabályokat a felhasználó határozhatja meg. A Fail2Ban több naplófájlt is képes olvasni, például az sshd-t vagy az Apache webservereket. [27]

2.5.3. *Konfiguráció*

A konfigurációm nagyon egyszerű, itt az 'jail.local' fájl tartalma:

```
[DEFAULT] # Vegyük az alapértelmezett beállításokat, de  
néhányat az alábbiakban módosítunk.
```

```
bantime = 1h # 5 sikertelen próbálkozás után 30 méteren belül  
tiltsa le a kapcsolatot az adott IP-ről 1 órára
```

```
banaction = ufw # Az alapértelmezett „banaction”  
meghívása a firewall-cmd-bő, de a Raspberry Pi-ben nincs ilyen,  
ezért az ufw-t használjuk helyette
```

```
[sshd]
```

```
enabled = true # engedélyezze ezt a fail2ban szabályt az SSH  
szolgáltatáshoz
```

A további beállításokhoz a hivatalos wikioldal mindent felsorol,
https://www.fail2ban.org/wiki/index.php/MANUAL_0_8#Usage.

Vagy használja a man oldalt a következővel:

```
$ man fail2ban
```


Ha végezt a szerkesztéssel, mentse el a fájlt, és futtassa ezeket a parancsokat, hogy a változtatások érvénybe lépjenek:

```
$ sudo systemctl enable --now fail2ban
```

```
$ sudo systemctl restart sshd
```

VII. Részletesen az RStudióról és a Shinyről

1. Az RStudio Server

1.1. Telepítése - Ansible Playbooks

1.1.1. *Definíció*

Az „Ansible Playbook” az automatizálási feladatok vázlata – összetett informatikai műveletek, amelyeket korlátozott vagy emberi beavatkozás nélkül hajtanak végre. Az Ansible playbookokat a gazdagépek halmazán, csoportján vagy osztályozásán hajtják végre, amelyek együtt alkotják az Ansible leltárt.

A „Ansible Playbooks” alapvetően keretrendszerek, előre megírt kódok, amelyeket a fejlesztők ad-hoc módon vagy kezdő sablonként használhatnak. Az Ansible playbookokat rendszeresen használják az IT-infrastruktúra (például operációs rendszerek és Kubernetes platformok), hálózatok, biztonsági rendszerek és fejlesztői személyiségek (például Git és Red Hat CodeReady Studio) automatizálására[28]. Esetünkben pedig, RStudio és Shiny szerverek telepítésére használták.

1.1.2. *Hogyan működik*

A Playbooks YAML (.yaml) formátumban vannak kifejezve, minimális szintaxissal. Egy playbook egy vagy több „play”-ból áll egy rendezett listában. A „playbook” és a „play” kifejezések sport analógiák. Minden play végrehajtja a playbook általános céljának egy részét, egy vagy több feladatot végrehajtva. Minden feladat egy Ansible modul-nak szólítanak.

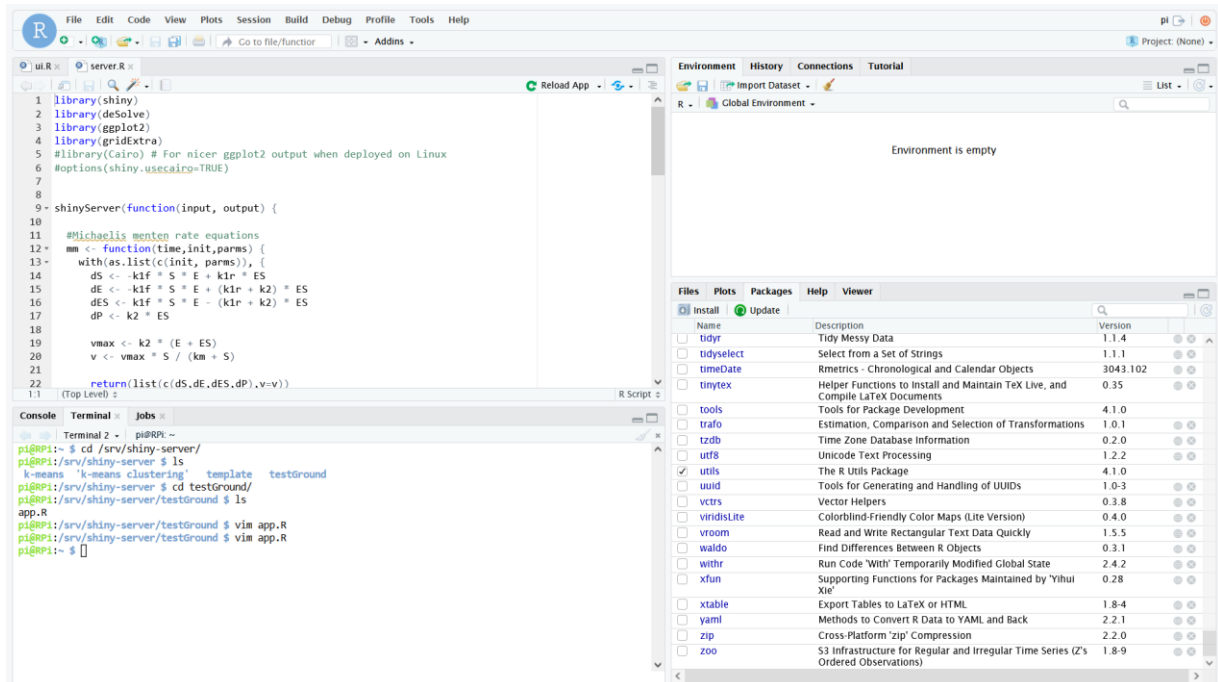
1.1.3. *Hogy miért használták esetünkben*

Az Ansible Playbooks megismételhető, újrafelhasználható, egyszerű konfigurációkezelési és többgépes üzembe helyezési rendszert kínál, amely kiválóan alkalmas összetett alkalmazások telepítésére.

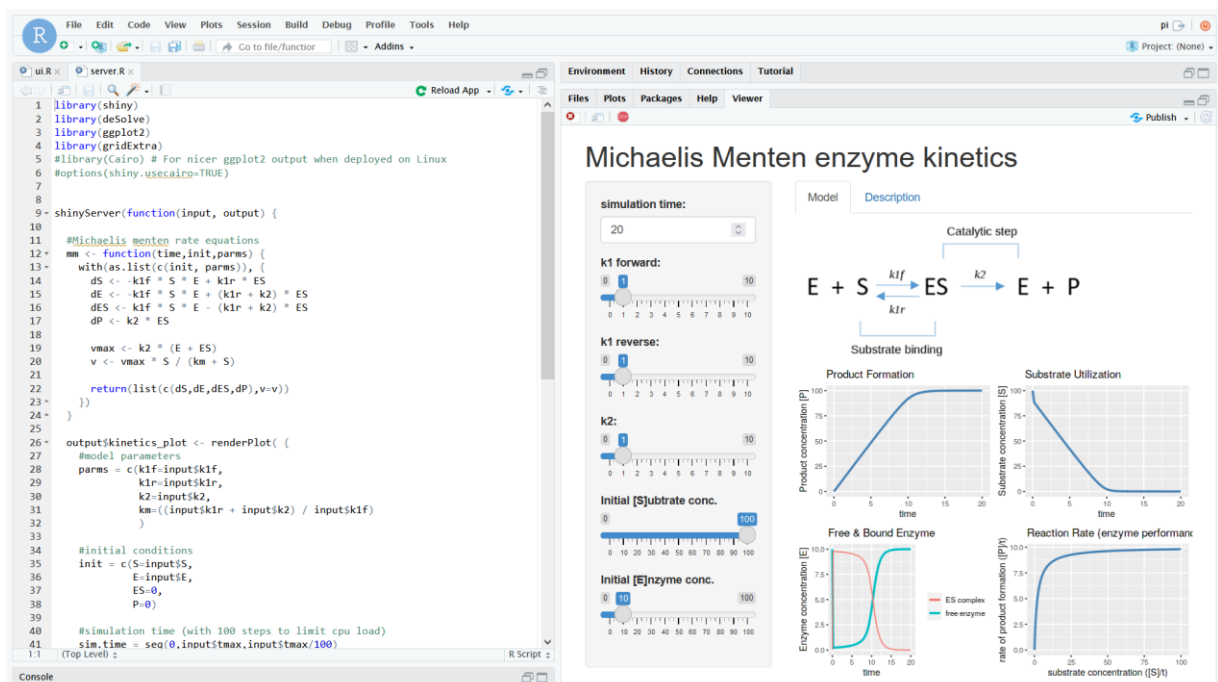
Tökéletesen alkalmas szerverek RPi-re történő telepítésére számára.

1.2. A munkaterületre

1. ábra



2. ábra



Az 1. ábrán, ez az RStudio munkaterület, amely 4 fő panelből áll.

- A bal felső sarokban, a kódolási helyet így hívtam, találhatók a projektünk fájljai, nevezetesen a server.R és az ui.R
- Az alatta az R konzol, a Unix terminál.
- A jobb oldalon a felsőt használjuk, ha adatokat importáltunk, ezzel nem sokat dolgoztam, csak lássam egyszerre, hogyan működik.
- Az alábbi panelen két fontos lapja található, ezek a "Packages" és a "Viewer".

Szóval a packages/csomagok alapvetően "libraries/könyvtárak" más programozási nyelvekhez, mint például C, Python. A "Viewer" hely arra való, amikor a kódokat a kódolási helyen futtatja.

Amikor kódolok, a weboldal úgy fog kinézni, mint a 2. ábrán.

Ezaz projektnek szükséges csomagok a következők: shiny, deSolve, ggplot2, gridExtra és Cairo (szebb kimenet Linux rendszeren)

2. A Shiny App – Pi Michaelis-Menten

A Shiny App, R nyelvű írott, 2 forráskód-fájllal rendelkezik, az egyik a server.R, a másik az ui.R, ahogy a nevük is sugallja, a szerver többnyire backendre szolgál, míg az ui a frontendre. Az /srv/shiny-server/[az_alkalmazás_neve]/ mappában található.



```
pi@RPi:/srv/shiny-server/demo $ pwd
/srv/shiny-server/demo
pi@RPi:/srv/shiny-server/demo $ ls
credit.txt  README.md  server.R    ui.R        www
LICENSE    rsconnect  server.R.bak ui.R.bak
```

A kezelőfelületem 3 fő részből áll, a bal oldalon a felhasználói beviteli tábla, a jobb oldalon 2 fül található, az első a grafikonokhoz, a második fül az alkalmazásról és annak utasításairól tartalmaz további információkat.

A háttérben 2 fő rész a Michaelis-Menten egyenlet és a "rajzoló" grafikonok.

```
#Michaelis Menten rate equations
mm <- function(time,init,parms) {
  with(as.list(c(init, parms)), {
    dS <- -k1f * S * E + k1r * ES
    dE <- -k1f * S * E + (k1r + k2) * ES
    dES <- k1f * S * E - (k1r + k2) * ES
    dP <- k2 * ES

    vmax <- k2 * (E + ES)
    v <- vmax * S / (km + S)

    return(list(c(dS,dE,dES,dP),v=v))
  })
}

#output plots
p1 <- ggplot(data=out, aes(x=S, y=v)) +
  geom_line(colour="steelblue",size=1) +
  xlab("Szubsztrát koncentráció ([S]/t)") +
  ylab("Termékképződés sebessége ([P]/t)") +
  ggtitle("Reakciósebesség (enzim teljesítmény)")

p2 <- ggplot(data=out, aes(x=time, y=S)) +
  geom_line(colour="steelblue",size=1) +
  xlab("Idő") +
  ylab("Szubsztrát koncentráció [S]") +
  ggtitle("Szubsztrát felhasználás")

p3 <- ggplot(data=out, aes(x=time, y=P)) +
  geom_line(colour="steelblue",size=1) +
  xlab("Idő") +
  ylab("Termékkoncentráció [P]") +
  ggtitle("Termék kialakulása")

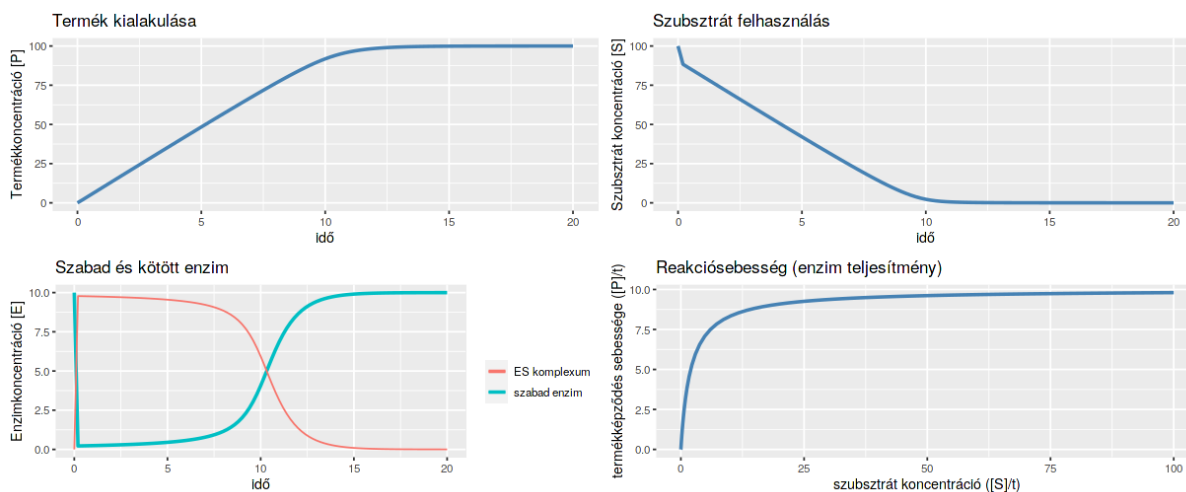
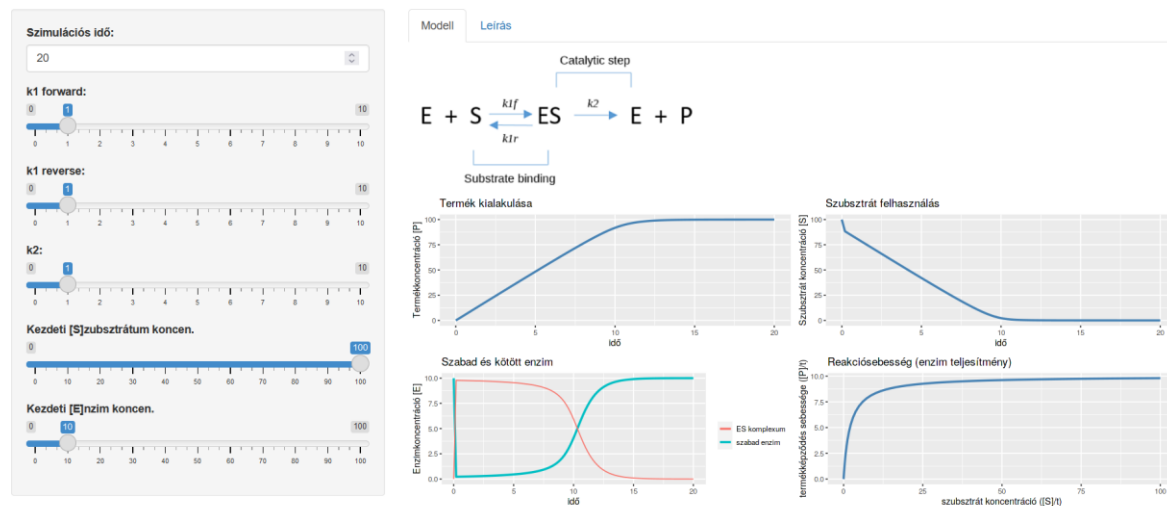
p4 <- ggplot(data=out, aes(x=time, y=E, col="Szabad enzim")) +
  geom_line(size=1) +
  geom_line(aes(x=time, y=ES, col="ES komplexum")) +
  guides(col = guide_legend(title = "")) +
  xlab("Idő") +
  ylab("Enzimkoncentráció [E]") +
  ggtitle("Szabad és kötött enzim")

#arrange the plots in a grid layout
grid.arrange(p3,p2,p4,p1, ncol=2)
```

VIII. Eredmény fejezet:

1. Észrevételek

Michaelis Menten enzim kinetika

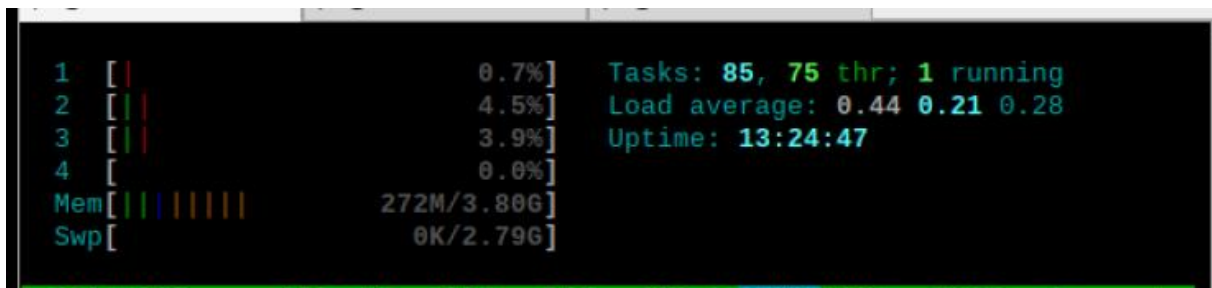


Láthatjuk, hogy a 4. grafikon a reakciósebesség változásait mutatja parabolával ábrázolva, és hogy a sebességnek el kell érnie egy maximális értéket.

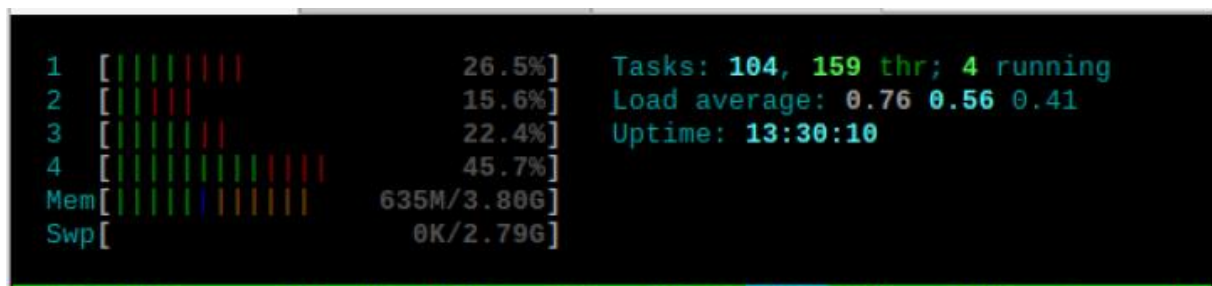
Az 1. és 2. grafikonon azt láthatjuk, hogy az idő előrehaladtával minél több terméket kapunk, annál alacsonyabb szubsztrátum marad.

Itt látható képek pedig a Raspberry Pi állapotairól, amikor üresjáratban, normál munkafolyamattal és webalkalmazás üzemeltetése közben:

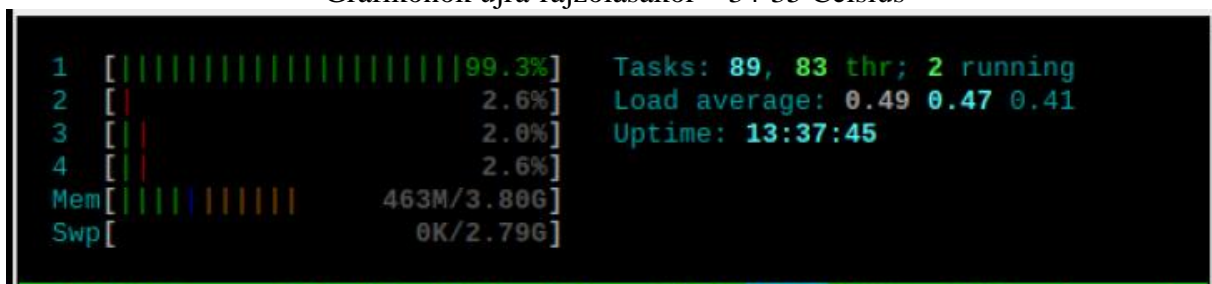
Üresjáratban állapotú – 30-32 Celsius



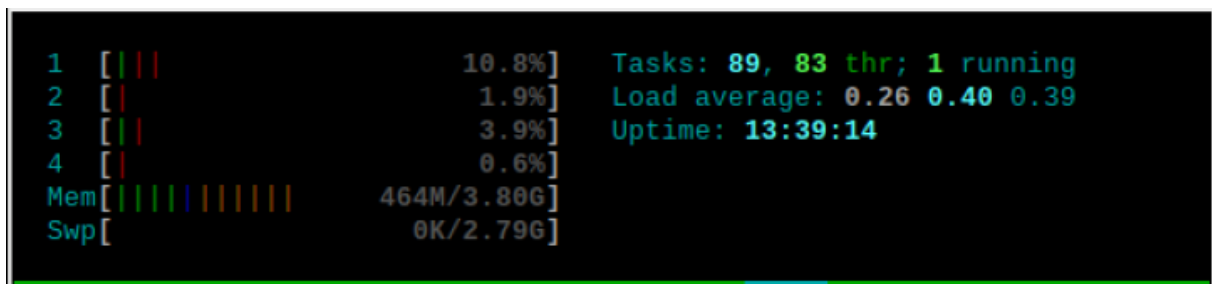
Munkafolyamatban állapotú - ~35 Celsius



Grafikonok újra-rajzolásakor – 34-35 Celsius



Amikor befejezi a számítást/rajzolást, és visszatér normálhostolásra – 30-32 Celsius

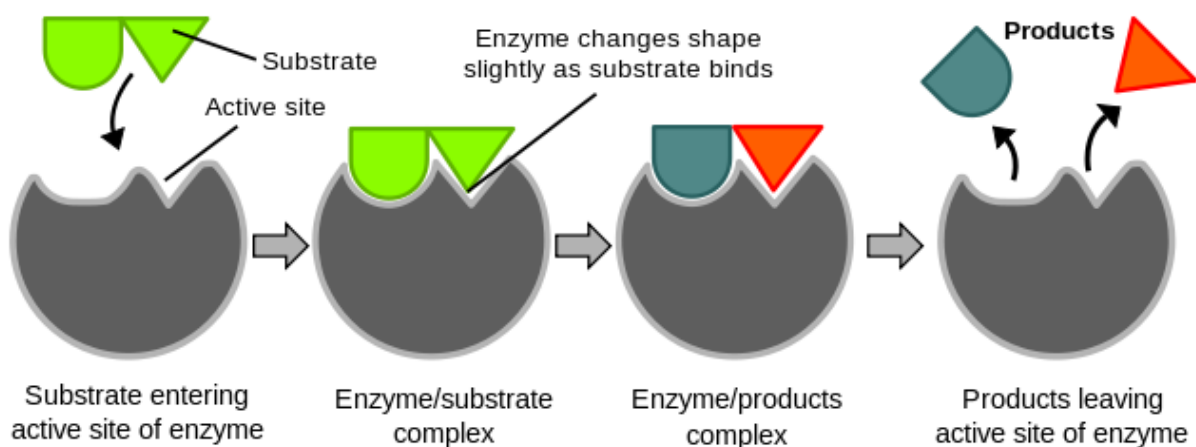


2. Értékezés / Megbeszélés

Ezek az eredmények a spekuláción belül vannak, mivel mint tudjuk, a Michaelis-Menten modellel a reakciósebességet ezzel a képlettel ábrázoljuk, amely tört formában van:

$$\text{Michaelis-Menten egyenlet: } v = \frac{d[P]}{dt} = V_{max} \frac{[S]}{K_M + [S]}$$

Az 1. és 2. gráfokon pedig, hiszen az enzim [E] és a szubsztrátum [S] közötti reakció kikényszerítésével a szubsztrátumot termékekké [P] alakítottuk, az alábbi kép leegyszerűsíti ezt a folyamatot:



Az alapbemenetek eredményein kívül a bemenetek megváltoztatásával (a bal oldali tábláról) a Raspberry Pi 3-4 másodpercen belül képes ábrázolni a grafikonokat, átlagosan grafikononként 1-re. De ennek az átlagnak azonban nagy a hibahatára, mivel minden grafikon más-más számítást igényel, így a számítandó képletek összetettségétől függően a betöltési idő változni fog.

Létezik egy lehetséges megkerülő megoldás, azaz a váltógombok használásával, amelyek segítségével kiválasztható, hogy melyik grafikonokat kell frissíteni, csak ha valaki úgy dönt, de következtetésem szerint erre a projektre nem lesz szükség, mivel az a cél, hogy mindegyiket bemutassuk egyszer. Talán egy bonyolultabb projektben ez hasznos lenne.

Egy másik probléma az, hogy néhány bemenettel, a grafikonok szekciókra lettek osztva, ahol látható kapcsolódási pontok vannak. Erre sajnos nem találtam megoldást vagy okot, hogy miért viselkedik így.

IX. Következtetések

Összegezve, elégedett vagyok a demóm eredményével, a WebApp több hónapja sikeresen fut probléma nélkül, a Raspberry Pi még a felét sem használta ki a maximális kapacitásának, ami azt jelenti, hogy több sok-sok kapacitás marad még pl. hány webalkalmazást képes tárolhat-e és futható egyszerre.

De most, az eredeti kérdésekkel visszatérve arra a következtetésre jutottam, hogy: igen, a Raspberry Pi remekül tud és fog is teljesíteni a PMM mellett webalkalmazások tárolásakor, ezzel az eredménnyel sok különböző háttérű ember számára hasznosnak bizonyul majd.

X. Hivatkozási lista

- [1] Raspberry Pi Foundation (2015-05-09). *About us*.
<https://www.raspberrypi.org/about/>
- [2] C. Severance (2013). “*Eben Upton: Raspberry Pi*”
Computer, Vol. 46, Issue 10, p14-16.
- [3] Raspberry Pi Foundation (2015-05-09). *What is a raspberry pi?*
<https://www.raspberrypi.org/help/what-is-a-raspberry-pi/>
- [4] Raspberry Pi Official Webpage: *Specifications*
<https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/>
- [5] Raspberry Pi Official Webpage: *GPIO*
<https://www.raspberrypi.com/documentation/computers/os.html#gpio-and-the-40-pin-header>
- [6] Raspberry Pi Documentation: *Useful Utilities - vcgencmd*
<https://www.raspberrypi.com/documentation/computers/os.html#vcgencmd>
- [7] Rodarte, Beatriz. (2013). “*A sejt molekuláris biológiájának gyakorlati kézikönyve I*”.
Természettudományi Kar, UNAM.
- [8] Michaelis, L., Menten, M.L. (1913). “*Die Kinetik der Invertinwirkung*”.
Biochem Z. Issue 49, p333–369.
- [9] Steinfeld J. I., Francisco J. S., Hase W. L. (1999). “*Chemical Kinetics and Dynamics*”
2nd ed., Prentice-Hall
- [10] H. Stephen Stoker (2015-01-01). *Organic and Biological Chemistry*.
Cengage Learning. p. 371. ISBN 978-1-305-68645-8.
- [11] Murray JB, Dunham CM, Scott WG. (2002-01). “*A pH-dependent conformational change, rather than the chemical step, appears to be rate-limiting in the hammerhead ribozyme cleavage reaction*”. *Journal of Molecular Biology*. 315 (2): 121–30.
- [12] Srinivasan, Bharath (2021-07-16). “*A Guide to the Michaelis-Menten equation: Steady state and beyond*”. *The FEBS Journal*: febs.16124.
- [13] Srinivasan, Bharath. (2021-03-18). “*Explicit Treatment of Non-Michaelis-Menten and Atypical Kinetics in Early Drug Discovery*”.
- [14] Srinivasan, Bharath. (2020-09-27). “*Words of advice: teaching enzyme kinetics*”.
The FEBS Journal. 288 (7): 2068–2083.
- [15] Worthington Biochemical Corp. “*Substrate Concentration (Introduction to Enzymes)*”.
<https://www.worthington-biochem.com/introBiochem/substrateConc.html>

- [16] Matt Richardson & Shawn Wallace (2013-04). “*Getting started with Raspberry Pi*”. 2. verzió, 2. fejezet 29-30.
- [17] Johannes Martin Berg (2009-02-29). “*Wifi overview*”
- [18] T. Richardson, Q. Stafford-Fraser, K. R. Wood and A. Hopper, "Virtual network computing" in IEEE Internet Computing, vol. 2, no. 1, pp. 33-38, Jan.-Feb. 1998, DOI: 10.1109/4236.656066.
- [19] Raspberry Pi Documentation: *Securing your Raspberry Pi*
<https://www.raspberrypi.com/documentation/computers/configuration.html#securing-your-raspberry-pi>
- [20] Peter Loshin & Michael Cobb, *Secure Shell (SSH)*
<https://searchsecurity.techtarget.com/definition/Secure-Shell>
- [21] Daniel J. Barrett & Richard E. Silverman (2001-02).
“SSH, the Secure Shell: The Definitive Guide” 1. verzió 1.1 fejezet “What is SSH?”
- [22] Kim Crawley (2021-05-21). *How SSH works: Secure Shell explained*
<https://cybersecurity.att.com/blogs/security-essentials/explain-how-ssh-works-to-me>
- [23] Daniel J. Barrett & Richard E. Silverman (2001-02). “SSH, the Secure Shell: The Definitive Guide” 1. verzió 1.3 fejezet.
- [24] Sbeattie (2021-10-17). *UncomplicatedFirewall*.
<https://wiki.ubuntu.com/UncomplicatedFirewall>
- [25] George Kyambadde & John Ngubiri (2018-09). “A Tool to Mitigate Denial of Service Attacks on Wired Networks”. IJCAT – International Journal of Computing and Technology., Vol 5, Issue 9, Szeptember 2018. ISSN:2348-6090
- [26] Ellingwood, Justin (2014). "How Fail2ban Works to Protect Services on Linux Server." Digitalocean. com 7.
- [27] Rama Bansode (2018-04 2018-09) “Mitigation of The Risk factor on Apache web server from DDoS Attack”.
Cyber Times International Journal of Technology & Management. Vol. 11 Issue 2.
- [28] RED HAT (2019-01-08). *What is an Ansible playbook?*
<https://www.redhat.com/en/topics/automation/what-is-an-ansible-playbook>

XI. Rövidítések lista

<u>Rövidítés</u>	<u>Jelentése</u>
RPi	Raspberry Pi
PMM	Pi Michaelis-Menten
OS	Operation System / Operációs Rendszer
LTS	Long Term Support kernel
ISP	Internet Service Provider
CS	Computer Science
GPIO	General Purpose Input/Output pins
GND	Ground
DIY	Do It Yourself
SSH	Secure (Socket) Shell protokoll
VNC	Virtual Network Computing
SU	Super User
UFW	Uncomplicated Firewall

XII. Nyilatkozat az írásmű eredetiségéről

NYILATKOZAT az írásmű eredetiségéről

(PTE SZMSZ 5. sz. mellékletének 14/1. számú melléklete alapján)

Alulírott

..... VO TRUNG HIEU(név)
.....NEIABP(NEPTUN kód), büntetőjogi felelősségem tudatában
kijelentem, hogy **R projekt (Studio, Shiny) telepítése és alkalmazása a
Raspberry Pi tenyérszámítógépen**.....

.....
című írásomban foglaltak saját, önálló munkám eredményei, ennek elkészítéséhez kizárólag a
hivatkozott forrásokat (szakirodalom, eszközök stb.) használtam fel, írásomat a Pécsi
Tudományegyetem vonatkozó szabályzatainak betartásával készítettem. Tudomásul veszem,
hogy a szerzői jogi szabályok betartását a Pécsi Tudományegyetem plágiumkereső rendszeren
keresztül ellenőrizheti.

Pécs, 20.....21.... év 12.... hó04 ... nap



.....
hallgató aláírása