

CC510 – Final Project Writeup

Abstract

In this writeup, I describe a potential solution to the lack of digitalized mental health care management in rural areas that lack internet connection by exploring a free open-source, locally hosted medical software called OpenEMR to be implemented in such areas. I outline what it means to be locally hosted compared to cloud-based, and I explain why this solution is needed in certain communities. Following that is a description of my own prototype of this software, including the building process, testing of multiple functions and data exportation features, and configuring specific settings, as well as why OpenEMR is an optimal choice among its competitors. A full review of this solution follows, including a SWOT analysis that outlines the strengths, weaknesses, opportunities, and threats to an OpenEMR implementation. In the conclusion, I reiterate why readers should care about this solution, and I suggest a potential future direction that developers could head with this software, which is creating new installable diagnostic testing modules, based on a previous coding project I completed in an Advanced Programming course with Python as its primary language. Supplemental files will be included for a more in-depth look into some of the concepts described in this writeup.

Introduction

It is no secret that individuals and organizations alike have been migrating towards using cloud-based services and data storage, a trend that seems to have its roots extending back to the 1960s. However, it wasn't until the late 1990s, when notable advances in virtualization, networking, and general computer technology began to emerge, that this switch from locally hosted services to cloud computing became the norm. According to Cloud Academy, "With more than 60% of the world's corporate data stored in the cloud, this service area has exploded, with cloud infrastructure services generating \$178 billion in revenue annually" (Singh, 2023). And it is not surprising why consumers are leaning toward the cloud.

SHARE OF CORPORATE DATA STORED IN THE CLOUD OVER TIME

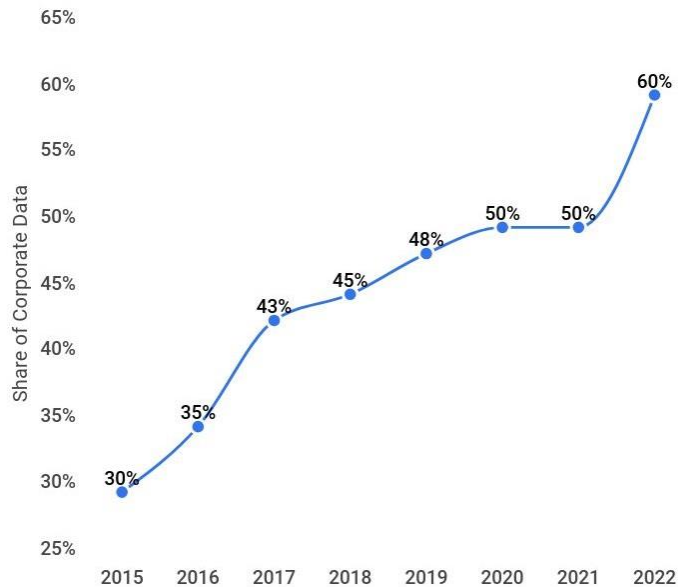


Figure 1 - This graph from zippia.com shows the increasing percent of corporate data being stored in the cloud over 7 years.

Cloud storage is the utilization of a third-party provider, such as Microsoft, Google, and Amazon, that hosts others' data on their own infrastructure, and this includes the maintenance of hardware, software, and all security components that are necessary for data access and control. To access your data through such a provider, one sets up an account with them, oftentimes with a monthly charge, from which they can login, view, and manipulate their data with added security features that the server provides. This separation of duties between the consumer and the provider is one of the main reasons cloud-based services have skyrocketed, because not every organization has the capabilities to provide an advanced level of security and maintenance expertise, or they can save on costs by not delegating a person or team to the effort. Other notable benefits to working in the cloud include no large upfront costs spent on locally based hardware, the potential for multiple types of security authentication provided by the cloud service, and the duty of updating software placed on the shoulders of the provider. Despite there being obvious benefits to cloud computing, there remain a few organizations that are hesitant to make the switch from self-hosting, and these primarily include companies that wish to keep their data completely internal and nearby for the fastest, and potentially most secure, service possible,

such as hospitals and other medical providers. There is also one major drawback to cloud computing for a surprising large number of people: the cloud needs the internet.

This paper serves as a proposal for the permanent and stand-by implementations of self-hosted counseling software in areas where internet access is either unstable or completely unavailable due to economic, environmental, or disastrous conditions. In most American communities, the internet is so integrated into nearly every aspect of our lives, school, work, healthcare, socializing, etc., that it is difficult for us to imagine a world where the internet is unavailable. However, according to Statista, a leading online research statistics portal, “As of October 2023, India was the country with the largest offline population worldwide. The South Asian country had over 672 million people without internet connection. China ranked second, with around 347 million people not connected to the internet” (Petrosyan, 2023). These countries are only two of many that have large populations that remain disconnected from the internet. This means that any services needing a stable connection, such as cloud data and other software, may not be utilized by organizations within these communities, and this lack of service can put these populations at a disadvantage in areas such as access to mental health care. According to Mental Health America, 65% of rural counties do not have a psychiatrist or other mental health professionals, and, what’s worse and possibly resulting from this lack of care, these same areas have a significantly higher rate of negative mental health outcomes, such as suicide (<https://mhanational.org/rural-mental-health-crisis>). In internet-lacking communities with so many people, it seems reasonable to assume that health and counseling professionals would benefit from software that allows them to keep track of patient records, such as scheduling, billing, test results, etc. A study conducted by Chilean researchers lends support to this claim by outlining the recent growth of digitally based mental health technologies in Chile and how continuous expansion of this field could benefit their country, stating that more resources should be allocated toward the effort (Rojas et al., 2019). So, if there exists a product designed to implement such services quickly and easily until the internet access deficiency is ever addressed in these areas, then a local government should look into providing such applications.

There does happen to exist open-source medical software with self-hosting capabilities, including all of the medical record services described above. The software that I have chosen to analyze, test, and suggest additional features for is called OpenEMR and is “the most popular

open-source electronic health records and medical practice management solution”, according to the product’s website (open-emr.org). Rather than storing all of a clinic’s data in the cloud, a practitioner would have their own hardware on-premises that would store SQL data used by OpenEMR. That data would include appointment info, prescriptions, billing statements, report documents, lab results, clinical decision rules, user account details, and multi-language customizations. OpenEMR also offers two different access portals, for patients and administration, hosted on the same port, so each type of user is only presented with the services they require.

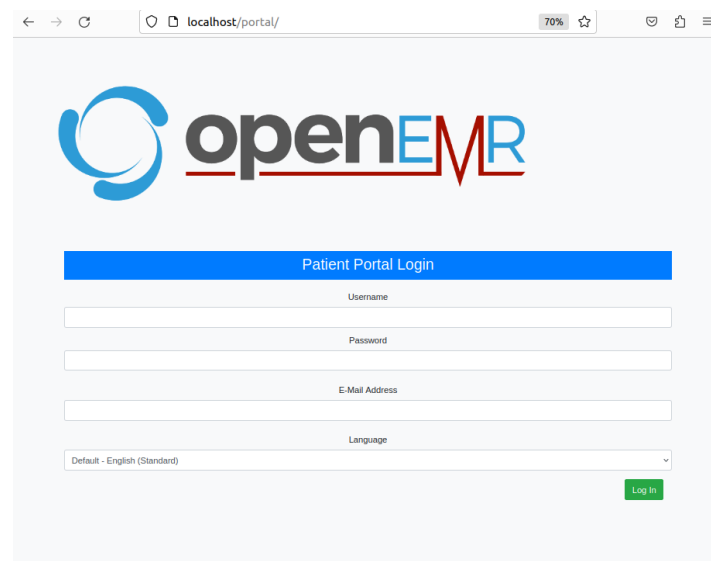


Figure 2 - An example of the separate login features for patients.

Unfortunately, I was unable to find self-hosted software dedicated to psychological services specifically, but self-hosting medical software also provides the advantage of customization for your particular clinic type. Most of the available open-source medical software I found in my search have a very generic setup and services that can potentially pertain to any more specific type of medical care organization, so they are essentially like blank slates waiting for user customization. For example, when an administrator adds a new patient to the system, directly after submission, a popup window appears notifying the creator of the medical records and screenings that this new patient is in need of, such as a pap smear if the new patient is female. If one were to customize OpenEMR to be utilized by a clinic focused on mental health/counseling, then patients’ required materials and procedures could be changed to more

relevant items, such as mental disorder screenings if the patient shows preliminary signs of a disorder. In fact, I plan to describe a potential addition to the software, complete with a graphical user interface (GUI) that would allow the psychologists of the clinic to administer psychological tests to the patients based on current official diagnostic testing, and the results would be stored within the local database.

To further the point that a locally hosted psychological care program would be easy to implement and necessary for communities without access to internet, the rest of this proposal will be structured to include an argument against the alternatives to OpenEMR, an overview of my own implementation of this software and the testing of key features, a critical evaluation and SWOT analysis of the solution, and some commentary on where future directions of this software could go and what additional features would be beneficial to add.

Related Work

Other solutions to the lack of medical cloud software that offers multiple digitalized features and data storage in areas without reliable internet include other attempts at self-host medical software, use of isolated notetaking and data storage features, and traditional physical recording methods.

Some of the other available locally hosted medical software with features similar to OpenEMR include FreeMED and GaiaEHR. When comparing these options against OpenEMR, it is important to consider installation method and ease, the variety of complex features included, and the user friendliness of the features, including the GUI. When comparing OpenEMR to these alternatives, they seem to have many of the same features like patient record keeping, prescription writing, billing methods, and management tools, but OpenEMR has the most modern and easy to navigate GUI, as well as additional features like lab integration (FreeMED has this) and clinical decision rules. Lab integration allows a practitioner to “have lab orders automatically sent to a lab and integrate the results into a patient’s chart automatically”, and the clinical decision rules let one “navigate complex patient algorithms to ensure the highest quality care for patients”, so a patient’s symptoms, severity of ailment, and rate of recovery are all possible variables that can be inserted into these clinical rules to provide a caretaker with an

optimal recovery plan; and these rules can be customized per practice needs. FreeMED lacks a separate patient portal like OpenEMR's, as well as patient demographic services and reporting/analytics.

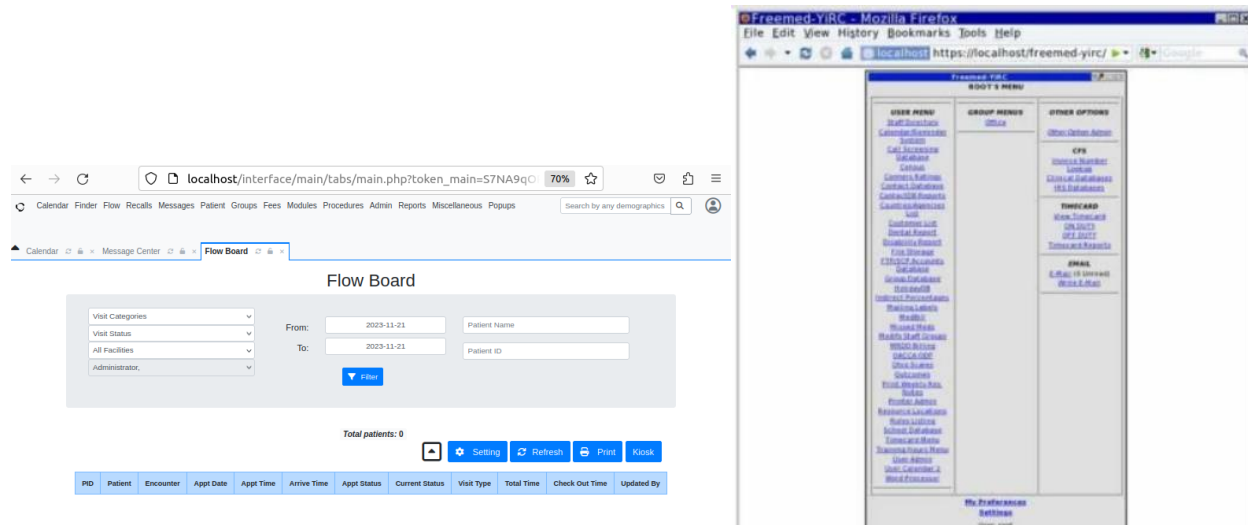


Figure 3 - OpenEMR (left) GUI vs. FreeMED (right) GUI. OpenEMR's is much more modern.

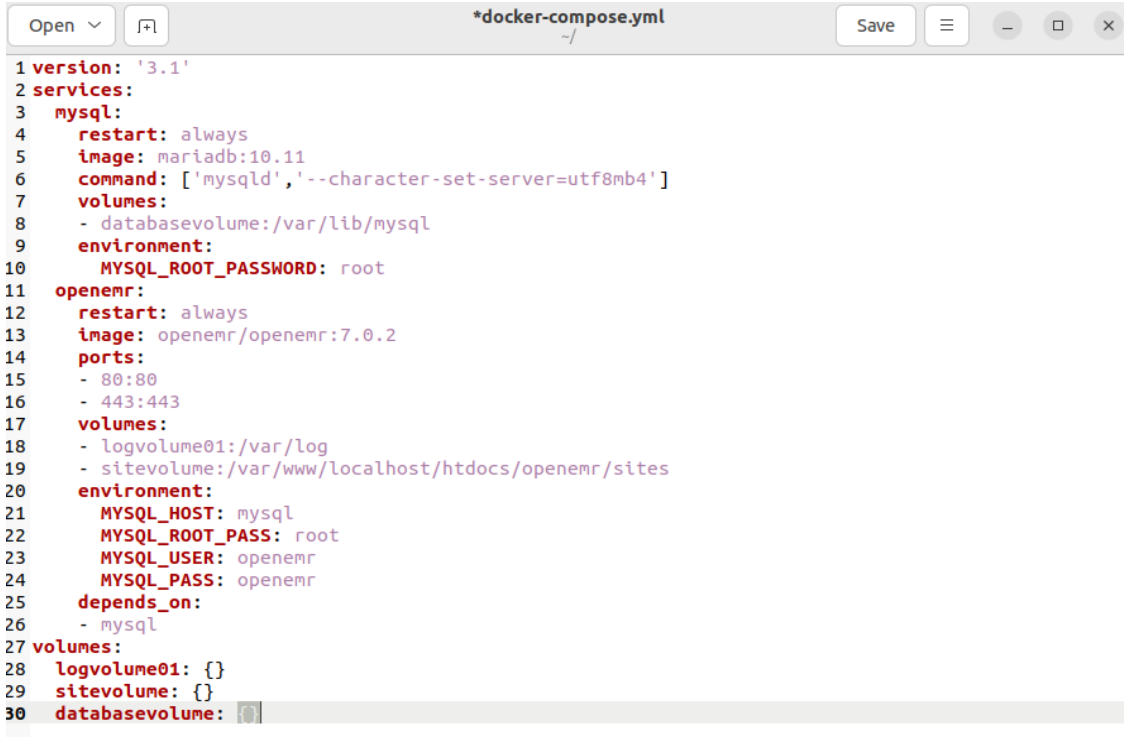
More important distinctions between the software options include the ease of installation and how consistently they are being maintained and updated. Both OpenEMR and FreeMED offer Docker images for running the software. A Docker image is essentially a prewritten file containing all the necessary application code, tools, libraries, and dependencies for a user to be able to build a container that can run the entire program (Gillis, 2021). This container can then be ran and accessed by a simple terminal command. Docker Compose files are sections of code that can help define and connect applications that are needed in a single program. Upon testing the Docker image for FreeMED and using a Docker Compose file to connect the application to an SQL database, the application was not working despite multiple attempts at troubleshooting. It turns out that FreeMED is no longer being consistently updated by the creators, and multiple of the required installations that the Docker image needs are outdated. It may be possible to successfully install this application through the traditional, more manual method, but compared to the simplicity of using a Docker image successfully with OpenEMR, the rigorous installation of FreeMED is not worth it, especially since its features fall short of OpenEMR's. To install GaiaEHR, you must also do it manually, including the installation of MYSQL database server, manual creation of users and databases, the downloading and unzipping of the program files, and

then a separate set of web instructions to follow. On top of this, the latest version of GaiaEHR was released in 2014. Perhaps with the push for cloud-based software, creators of open-source, locally hosted programs such as these have begun to abandon efforts to maintain their projects. This is another advantage of choosing OpenEMR: it is still being maintained. The latest version was released one week ago, and there exists an active forum for users to receive answers about the software when needed. Overall, when comparing the three software choices, OpenEMR seems to be the most well-rounded and reliable. It is no wonder that it is ranked the #1 PHP open-source medical software by medevel.com in 2019 and #2 by goodfirms.co in 2023.

The other forms of patient management and keeping counseling service data, utilizing individual programs and physical notetaking, are most likely the methods currently used by any counseling services in communities without internet service. But it can be very reasonably assumed that a program that combines patient record keeping, prescription assignment, employee management, and scheduling, among many other features, would be much more beneficial and organized for those in charge of a clinic. That is the entire reason programs like OpenEMR were created. The fact that these locally hosted software are scalable makes them usable by a practitioner of any size clinic without having to worry about having too few or too many clients.

Implementation

In order to implement OpenEMR into a local counseling clinic's service using Docker Compose, one needs to have a Linux operating system. I utilized a virtual machine to isolate the program on my computer so there would be no interference from any other previous installations or MYSQL instances and used the Ubuntu 20.04 desktop 64-bit installation file. To do this, openemr.org supplies its prospective users with a prewritten Docker Compose file which includes all the necessary services the OpenEMR needs to run in an isolated environment. The file also helps guide the services to the preferred ports so that you are aware of where to allow your firewall through as well as where to access them. This application in particular can be accessed on ports 80 and 443, which is the default network port used to access HTTP and HTTPS sources. You can copy the file from github and add it to the appropriate computer's file system and name it "docker-compose.yml".



```
1 version: '3.1'
2 services:
3   mysql:
4     restart: always
5     image: mariadb:10.11
6     command: ['mysqld', '--character-set-server=utf8mb4']
7     volumes:
8       - databasevolume:/var/lib/mysql
9     environment:
10      MYSQL_ROOT_PASSWORD: root
11   openemr:
12     restart: always
13     image: openemr/openemr:7.0.2
14     ports:
15       - 80:80
16       - 443:443
17     volumes:
18       - logvolume01:/var/log
19       - sitevolume:/var/www/localhost/htdocs/openemr/sites
20     environment:
21       MYSQL_HOST: mysql
22       MYSQL_ROOT_PASS: root
23       MYSQL_USER: openemr
24       MYSQL_PASS: openemr
25     depends_on:
26       - mysql
27 volumes:
28   logvolume01: {}
29   sitevolume: {}
30   databasevolume: {}
```

Figure 4 - Default OpenEMR Docker Compose file.

To access these Docker images, you first need to download Docker Engine and Docker Compose onto your system. Once these applications are installed, you simply need to run the command “sudo docker compose up”, and you will see the image layers of each included application building. For OpenEMR, it typically takes 5-10 minutes on the first run to build the dependencies and connect to the corresponding MYSQL database.

To access OpenEMR once the Docker image is ready, you can visit your preferred web browser at “localhost:80”. There, you will see the administrator login page where you can insert in the default admin credentials of “admin” for the username and “pass” for the password in order to begin configuring the application to fit your needs. To access the patient portal, you first need to indicate in settings to allow the usage of the portal, and then you can access it at localhost/portal. To make this process easier for both admin users and patients, a simple startup webpage can be created that links to the two login pages; this addition is also beneficial if clinic administrators wish to provide users of OpenEMR to more information about their service or provide contact information in a single and organized place. I have coded my own OpenEMR

webpage to demonstrate this potential capability, and the code is included in the supplemental documents.

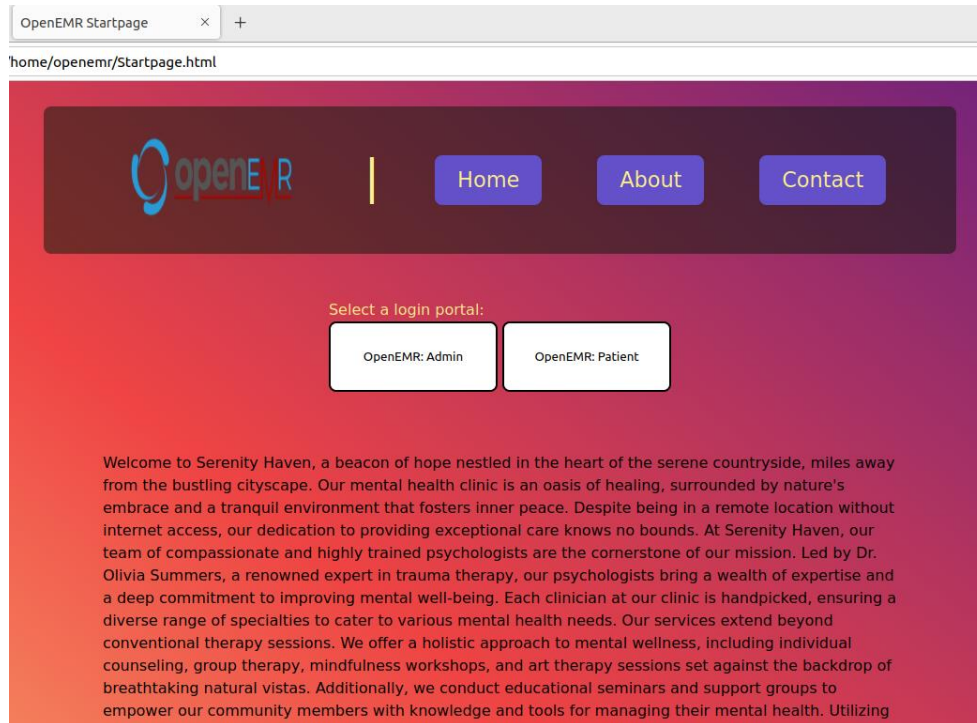


Figure 5 - Example clinic website to access OpenEMR logins.

In order to have this webpage start up automatically upon logging in, similar to how most organizations have an easily accessible startup application, I added the file path as the default Firefox page and added Firefox as a startup application using the “gnome-session-properties” command in the terminal and specified the rule.

Once you are in OpenEMR, you see a toolbar across the top with all your necessary features for keeping your data organized, including Patient Finder, Messages, Calendar, Groups, Reports, Admin, and many more tabs. The first item on an admin’s list will be to change their password to something more secure, which is easily done under the Admin>Users tab. This is also where an admin can add additional admin users with various roles and access types to select from. Once a user or a patient is created, multiple users can create appointments, send messages and documents, and create office reports for anyone in the system, and reminders or notifications will appear on the appropriate accounts, and all of the created information is stored on the clinic’s local system and hard drives. If the clinic truly has no access to internet, then a single

computer will have to be the sole means of accessing clinic information, which will most likely lead to a patient portal being obsolete and disabled. However, if the clinic is able to set up a storage area network (SAN) locally, then multiple computers within a clinic can access shared storage devices, such as hard drives; this way, patients can have their own device(s) within the building and can use them to utilize all the application's functions, like messaging, billing, and scheduling, while the administrators have theirs separate. Also located under the Admin navigation is a tab called Config whose page contains multiple customization features, including full rebranding of the site, module activation, stylization, security configuration, and many more that would make it extremely easy for a psychology clinic to make their application more specialized for their particular practice and needs.

Another important factor to explore when considering a self-hosted program that involves large amounts of data is the ability to export data between two systems running the same software locally and how easy this process is for the average user. Luckily, the developers of OpenEMR included a module called Carecoordination to handle the exporting and importing of medical patient data, but the process to do so requires a few steps of configuration. First, you must navigate to Admin>Config>Connectors and enable the CCDA Service ability. Then, activate the Carecoordination module under the Modules>Manage Modules tab, and make sure that the module is both installed and registered. Once this is done, access the module's settings and ensure that the appropriate users can "Send to HIE", and select a user for both the Author and Primary Care Provider dropdowns. The Carecoordination module should now be correctly configured. To actually send and receive data, an admin simply needs to open the module and select their desired task. For exporting, a patient needs to have had at least one "Encounter" with the clinic; an encounter can be added on the patient's account page. The admin then selects their patient, clicks "SEND TO", and then chooses download, since in this scenario the clinic is without internet service. This action will download a zipped file containing a few data files. The one administrators will be most interested in to transfer data to another system will be the file with the .xml extension, since this is the file that another clinic's administrator will use to view the patient's data. While testing this feature, I created my own test patient, downloaded their data, and attempted to import them back into the system. To import, you simply select the Carecoordination's "Import" tab and upload your CCDA.xml file under the CCDA tab. Once the file is uploaded, the software should grant you the ability to automatically create a new patient

from the data or add the data to an existing patient's account. If these features do not work correctly for the admin attempting them, they are still able to upload the file and view the patient's information and enter it manually.

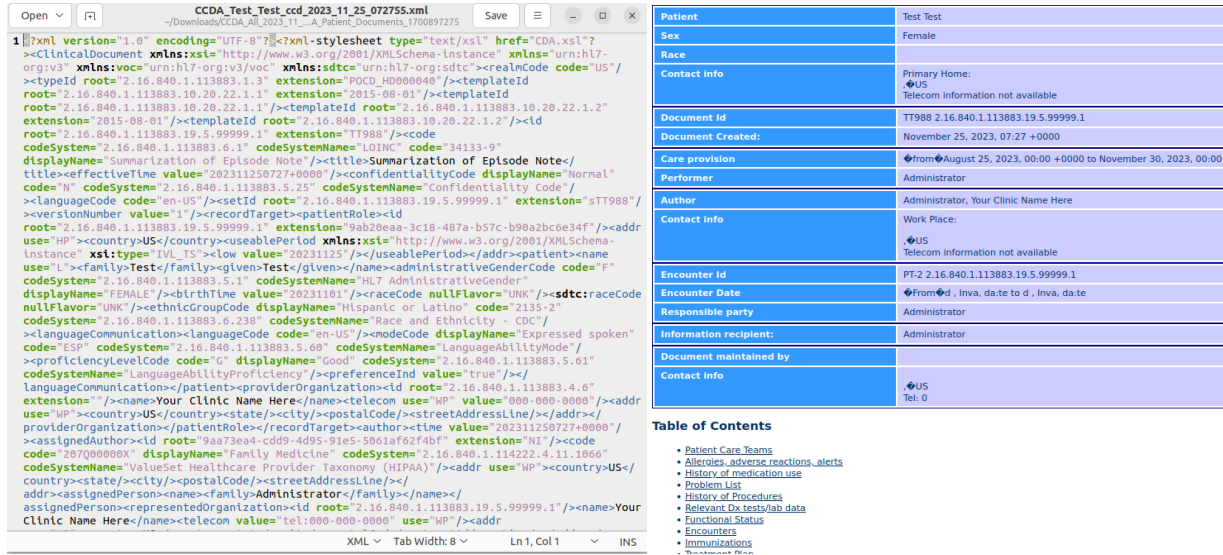


Figure 6 - Example patient .xml file (left) and how it appears in the Carecoordination module (right).

Transferring the downloaded data file to another clinic's database can be done in multiple ways, and I will be going over two. One does not require an internet connection but is more time-consuming, and one does require a connection but which is very useful for data sharing between medical providers of the same clinic in different locations. The first method which must be done without internet is simply using an external hard drive to download then transport the file from one location to the next, and this obviously must be done with legal precautionary steps, since the file contains sensitive information. The second method which must be done with internet service is by utilizing server and client devices. The computer acting as the server must have a form of directory access server software, such as LDAP and its Account Manager software, which will allow for the creation of users, groups, and file permissions for others accessing data on the host server, as well as SSSD, which can be configured for the client to use LDAP for authentication. The client can then use the server IP address to log into an existing user profile created on the server.

```
[sssd]
config_file_version = 4
domains = ldap.openemr.india.gov

[domain/ldap.openemr.india.gov]
id_provider = ldap
auth_provider = ldap
ldap_uri = ldap://ldap.openemr.india.gov
cache_credentials = True
ldap_search_base = dc=ldap,dc=openemr,dc=india,dc=gov
```

Figure 7 - Example of an SSSD config file for a client connecting to the clinic server.

In order for the server to share its own files with the client user, a file server must be configured, and this can be done using Samba, a tool which allows file sharing between machines accessing a single network. Samba must be installed on the server machine, and a Samba configuration file should be edited to add new shares to the network, and the users on the server who will be accessing the shares should be added to the Samba server. From the client, you should now be able to access the shared data and can proceed to mount any shared folders to your device.

This proposal does not suggest an OpenEMR implementation as a permanent solution for a lack of internet-based counseling software or for large-scale services, but for the usage by communities who are in the process of implementing internet services in the future and small rural communities who need mental health care. A small clinic can very feasibly use this software to manage a group of users, as one current user of OpenEMR had 3,000 patients in their system with 1TB of hard drive space with 80% still remaining, using HTPC motherboards, quad core AMD A4 5050 processors at 2.4Ghz, and 8G RAM. This setup seemed to last about 5 years before the user had to upgrade the hardware. As long you are able to find an IT specialist to install and occasionally upgrade or repair the technology, a clinic can easily self-host more data storage if needed if a similar setup is used. If the clinic's services begin to expand rapidly, OpenEMR has been able to handle much larger number of patient data. Another user writes about a hospital in Kenya which utilized OpenEMR and had 120,000 patient visits in a single year, and that was using the outdated versions from 2011, so the capabilities are likely much more impressive today.

Evaluation and SWOT Analysis

The first requirement to evaluate your own solution is testing to see if it is feasible and if all the components work for its intended use. As stated above, the initial usage of the Docker Compose file and the login process was very simple and worked perfectly. Unlike other self-hosted solutions, like FreeMED, OpenEMR is constantly being updated to fix bugs and add additional features, and there exists an active user help forum if you come across any trouble. On top of that, there is already a plethora of previous question threads that have been answered thoroughly and have been helpful in my usage of the software. The messaging, patient setup, data exportation, appointment scheduling, and module configuration features have all been tested and work seamlessly. The ability for multiple instances/runs of the OpenEMR container to maintain the same data has been tested as well with no issue. So far, the only feature I have found to not work is the Carecoordination CCDA document import function that allows you to automatically add an imported user's data directly into your system or add them to an existing user.

SWOT: STRENGTHS

The first strength of the OpenEMR solution to the lack of mental health care in areas without internet access or with an unstable connection is that it does not require internet beyond the initial software installations and the simple Docker Compose file build. This can easily be completed in an area with connection, and then the devices can be transported to clinics that need the software. If the clinic decides it needs extra storage space and processing power, it would need to have the support of an IT specialist to set up processors and other hardware to do so. An additional strength of OpenEMR when compared to its competitors is that it includes many additional features that would be beneficial for any health clinic that the others lack, such as lab support, a patient portal, clinical decision rules, multilanguage support, and a patient data exportation module. On top of that, OpenEMR is completely free and ONC certified, meaning the product meets the "technology capability, functionality, and security requirements adopted by the U.S. Department of Health and Human Services" (HealthIT.gov). OpenEMR also has broad user and developer support in the form of forums, chats, social media pages, and recorded demos to demonstrate the software's capabilities and uses. This strength is particularly important when open-sourced, locally hosted software is involved, since with the rise of cloud computing such

software is becoming obsolete and developers cease maintenance, as seen with FreeMED. Since this program would be locally hosted, it comes with the general strengths from that as well, such as quick processing times and not relying on or paying a third-party service to store and protect your data, which adds to security. The final strength I will touch on is that of customization; since OpenEMR is a generic health software, it was not designed with solely psychological services in mind. However, the creators included multiple customization features within the original code, including web redesign and renaming capabilities, multiple admin user types to choose from, module activation and deactivation options, customizable clinic rules, and many more.

SWOT: WEAKNESSES

Locally hosting your own software and data comes with a few general drawbacks, some of which have promoted the rise of cloud computing. One is the need for expertise when dealing with the installation, maintenance, and security features that normally your third-party service like Google would take care of for you. If your clinic chooses to install the entire database server setup such as the user mentioned above, it will need both the expert and the hardware to do so, which may result in larger up-front costs. A clinic would also need to ensure that such an expert would be on hand to fix any issues that may arise with the hardware or handle any updates. People skilled in IT services are becoming more and more prevalent, however, so this may not be a particularly strong weakness. The lack of stable internet in the areas that need such services like OpenEMR do face another challenge, which is being able to view web help forums for the software in order to answer questions. One possible solution would be to simply call a representative that can search the solutions for you, or the initial installer of the program could print out or download any necessary/helpful pages before travelling to the area without proper service. One final weakness with self-hosting this software is apparent in a real-life story told by a local IT specialist supporting the Kansas State University's Lafene Health Center. According to them, the health center used to house all their data in the university's library datacenter, so locally hosted. In 2018, the library caught on fire, and hundreds of thousands of water poured into the building during the efforts to put the fire out, and this flooded the servers in question. After this unforeseen disaster, the health center was "without servers for almost a month which meant we had no medical software. Which means no access to charts, medications etc.", and they

promptly switched to becoming hosted by a medical software vendor. All they need is a stable internet connection to get their software. This story highlights one of the major concerns with self-hosted software, that your programs and data are all subject to loss by unseen circumstances, especially if backup processes are not in place. However, this weakness does not counter the fact that rural clinics or clinics in areas that have suffered a disaster often do not have the internet capabilities to *not* be self-hosted.

SWOT: OPPORTUNITIES

I believe that it is apparent that mental health and the importance of intervention have been topics that have been more recently emphasized in popular media, with tv shows and movies having themes of mental health, popular celebrities speaking out about their own experiences with detrimental mental health, and the public outwardly beginning to not ostracize those who are not mentally well or encouraging them to not speak about such topics. This rise in awareness could be very helpful in this cause of distributing free digital health software to psychologists in rural areas, both for the acceptance of supporters and donators and for the encouragement of the local communities that will be attending such clinics. With the simple implementation of OpenEMR in mental health clinics, there usage could raise awareness of the existence and ease of such locally hosted software, and more health organizations, like doctors' offices, dental service providers, etc. could also adopt the program for their own needs. This is an opportunity to aid more citizens through the benefit of the software being customizable. Another benefit of the changing times is the continuous improvement of technology. Hardware is constantly becoming more user-friendly, smaller, and accessible, which can make the task of self-hosting a larger application less daunting. One blog about the comeback of self-hosting states that technology like AI "makes it even easier for those who lack the knowledge to get started" (<https://noted.lol/the-future-of-self-hosting/>). It is believed that AI can aid in application deployment and setup, making these tasks simpler for beginners, so setting up a locally-hosted data server in a remote area may be simpler than before if one studies on how to do it before venturing to the specific locations.

SWOT: THREATS

One potential threat using free open-source software is the possibility that the creators and contributors will cease to continue supporting the application, so stop producing upgrades,

recording demos and help videos, posting on the help forums, etc. In this case, the end users would most likely have to rely on their own learned expertise, read old material, or reach out to other experts for help. Or they may have to switch to an entirely new software, which may be difficult to import patient data into automatically. Although the implementation of internet into the targeted communities can be seen as a threat to this solution, broad internet connection and getting people access to online resources is the ultimate goal; the usage of a locally hosted service may only need to be a temporary solution. Another potential threat would be the reluctance of mental health providers in the area who already have their own methods of conducting their practices to adopt this new software. It would be important to demonstrate to such clinics previous use cases and how OpenEMR has benefited other health organizations in the past and why it may be more reliable or convenient than current practices.

Conclusions and Future Work

The problem that I am aiming to address and suggest a solution for with this writeup is that of a lack of mental health care and digitalized patient management in areas without a stable internet connection. The proposed solution seems feasible and can easily be undertaken by a local government or organization. I propose that, if necessary, hardware running OpenEMR be provided to communities that do not have access to cloud health resources, especially within the realm of mental health. OpenEMR provides a simple solution to managing patient data, recording lab results, organizing admin resources, scheduling, and sharing data across clinics. Poor mental health is associated with many negative life experiences and outcomes, such as poor educational success, substance use disorders, suicidal thoughts, lower economic contribution, among dozens of others (whitehouse.gov). These consequences are not short-term or isolated, either; studies show that children of parents with mental health disorders are twice as likely to develop disorders of their own (Landstedt & Almquist, 2019). Poor mental health can affect all aspects of one's life, and if there are is a lack of organized treatment available, the entire community is more likely to suffer as a whole if fewer people are receiving treatment. This is a reasonable call for those in power to provide any citizens lacking organized counseling resources and clinics that need digitalized health tools to provide them with a resource like OpenEMR. It is

easily configurable, free of charge, and full of impressive features which providers could find invaluable.

If counseling or psychiatric clinics in areas without internet service do decide to implement an OpenEMR solution, there could be potential for future additional features that would improve the flow of record taking and could make testing procedures more uniform. This potential add-on is that of integrated psychiatric tests based on standardized assessment tools. Such a feature could be added into OpenEMR as an additional optional module that can be installed and enabled if needed. Such a module would simply take diagnostic criteria tests and their physical paper forms and turn them digital so that the psychologist administering the test can automatically receive a calculated result of that patient's likelihood for the tested disorder and then insert the patient's results into their account for future reference. An example of such a program can be seen in a project I completed for my degree at Kansas State University approximately one year ago in the course named Advanced Programming, whose code will be included in the supplementary documents. I employed advanced python skills and knowledge to write the source code, tests, and GUI elements for a program that would allow a child psychologist to diagnose a child with a violent personality and likelihood of becoming disruptive in the future. To do so, I employed severity enums for different types of violence, used three layers of inheritance with parent and child classes to differentiate between varieties of childhood signs, victims of their violence, and abusive situations, and various useful python structures like singletons. For example, the Abuse class within this program holds the functions for recording the presence and severity of any abuse suffered by the child patient, inheriting from the Experience parent class. It includes functions such as `__init__` for instantiating the class, `name` for returning the factor name, `score` for totaling the abuse score that will contribute to the child's overall diagnosis depending on the severity of the abuse, `__str__` for returning a string description of the experience in future records, and `__eq__` for determining if the object created by the Abuse class is a qualified abuse object.

```

13 class Abuse(Experience):
42     @property
43     def score(self) -> int:
44         """Score getter.
45
46         This method determines what the score of the experience will be
47         based upon its severity enum and returns that int value.
48
49         Returns:
50             The experience's total score.
51
52         Raises:
53             ValueError: The severity must be a valid Severity enum.
54         """
55         if self._severity == Severity.LIGHT:
56             return 5
57         elif self._severity == Severity.MEDIUM:
58             return 10
59         elif self._severity == Severity.SEVERE:
60             return 15
61         else:
62             raise ValueError("Incorrect severity")
63
64     def __str__(self) -> str:
65         """String override.
66
67         This method returns a string description of the experience
68         based on which severity was chosen and the experience's name.
69
70         Returns:

```

Figure 8 - A portion of the Abuse class code as described.

Once all of the criteria classes were coded and tested with more code, I created the GUI using the built in Python GUI tool called tkinter. The GUI is the portion that the user will interact with in order to manipulate the code in meaningful ways and get results. Creating this portion consisted of coding and arranging multiple different panels' columns, rows, titles, and buttons associated with certain button clicks and choices made by the user. Depending on which selections were made, the program would total up a score that would indicate how violent the child patient may be in the future and print the results out on a receipt-like page, detailing the child's history and signs. Instead of only printing out these results, a Carecoordination diagnostic module could also add these results to a new SQL data column for the patient for the practitioner to access later.



Figure 9 - Example of my GUI code (left) and the main user panel (right)

A psychiatric clinic could have multiple copies of a similar diagnostic testing program, but for many different disorders and conditions; you would simply need to change the scoring criteria to fit the appropriate case. In order to store these criteria for the different conditions, one could simply make a SQL table for each diagnosis and add the criteria measures as row entries and the attributes of these measures as the column types, such as severity, type of victim, age of victim when the measure was present, score, etc.. Then, a user could access the diagnosis table through Python, query for all rows to set up the diagnostic test's questions, and then the column types would represent which choice types popped up in the GUI for each question. For example, if the user selected a depression test, the criteria options/buttons that would show up on the GUI main panel. Then, when you click on the criteria's button, you see another button with the selectors based on the columns for that criteria that have input. This could then create a copy of the table but for the answers to all of these questions, and a user could find a specific value by entering a command like: "SELECT * FROM john_doe_depression WHERE severity = 'high';". This command would search the patient John Doe's depression test results and return any criteria that he indicated having high severity.

This proposed additional module to an implementation of OpenEMR is only one possible direction a developer could go to improving the application and customizing it to fit the specific

clinic's expertise. Because of OpenEMRs customization abilities, the possibilities are endless for someone with the skills and knowledge to explore this software further. Even without these capabilities, learning how to run the application and use it is extremely simple, and its usage has the potential to improve many people's lives if the initiative is taken.

References

- Benner, T. (2022, May 31). Reducing the Economic Burden of Unmet Mental Health Needs. The White House Council of Economic Advisers. <https://www.whitehouse.gov/cea/written-materials/2022/05/31/reducing-the-economic-burden-of-unmet-mental-health-needs/>
- Choi, J. (2019). The Role of Telepsychiatry in Meeting the Mental Health Needs of Underserved Populations: A Review and Future Directions. *BMC Psychiatry*, 19(1). <https://bmcp psychiatry.biomedcentral.com/articles/10.1186/s12888-019-2278-1#:~:text=Background,present%20with%20similar%20problems%20themselves.>
- Cloud Academy. 10 Benefits of Using Cloud Storage. <https://cloudacademy.com/blog/10-benefits-of-using-cloud-storage/>
- GitHub. OpenEMR Docker Compose File. <https://github.com/openemr/openemr/blob/master/docker/production/docker-compose.yml>
- GoodFirms. Best Free Open Source Electronic Medical Records Software Solutions. <https://www.goodfirms.co/electronic-medical-records-software/blog/best-free-open-source-Electronic-Medical-Records-software-solutions>
- Institute for Economics & Peace. (2023, March 7). Global Number of Natural Disasters Increases Ten Times. Vision of Humanity. [https://www.visionofhumanity.org/global-number-of-natural-disasters-increases-ten-times/#:~:text=A%20look%20at%20data%20over,Threat%20Register%20\(ETR\)%20shows.](https://www.visionofhumanity.org/global-number-of-natural-disasters-increases-ten-times/#:~:text=A%20look%20at%20data%20over,Threat%20Register%20(ETR)%20shows.)
- Mendenhall, A. N., & Stuart, E. A. (2019). Trauma-Informed Care and Mental Health. *Frontiers in Public Health*, 7. <https://www.frontiersin.org/articles/10.3389/fpubh.2019.00391/full>
- Medevel. Open Source Cloud PHP EMR EHR HIS. <https://medevel.com/opensource-cloud-php-emr-ehr-his/>
- Microsoft. Cloud Storage vs. On-Premises Servers. <https://www.microsoft.com/en-us/microsoft-365/business-insights-ideas/resources/cloud-storage-vs-on-premises-servers>

Mental Health America. Rural Mental Health Crisis. <https://mhanational.org/rural-mental-health-crisis>

OpenEMR. Carecoordination Module Configuration. https://www.open-emr.org/wiki/index.php/Carecoordination_Module_Configuration

OpenEMR. OpenEMR. <https://www.open-emr.org/>

OpenEMR. Use the Carecoordination Module. https://www.open-emr.org/wiki/index.php/Use_the_Carecoordination_module

OpenEMR Community. OpenEMR Limits. <https://community.open-emr.org/t/openemr-limits/5577/2>

Secure Storage Services. Pros and Cons of Cloud Storage.
<https://www.securestorageservices.co.uk/article/11/pros-and-cons-of-cloud-storage#:~:text=If%20your%20Internet%20connection%20fails,access%20your%20remotely%20stored%20data.>

SelectHub. OpenEMR. <https://www.selecthub.com/p/emr-software/openemr/>

Software Finder. FreeMED. <https://softwarefinder.com/emr-software/freemed>

Statista. (2021). Countries with the highest number lacking internet access 2021.
<https://www.statista.com/statistics/1155552/countries-highest-number-lacking-internet/>

TechTarget. Docker Image. [https://www.techtarget.com/searchitoperations/definition/Docker-image#:~:text=A%20Docker%20image%20is%20a,virtual%20machine%20\(VM\)%20environments.](https://www.techtarget.com/searchitoperations/definition/Docker-image#:~:text=A%20Docker%20image%20is%20a,virtual%20machine%20(VM)%20environments.)

Zippia. Cloud Adoption Statistics. https://www.zippia.com/advice/cloud-adoption-statistics/#Cloud_Adoption_Trends