Smart Medication Dispenser
Project Proposal

Matthew Boyd (mcboyd@bu.edu)
Evelyn Liu (levelyn@bu.edu)
Yutong Wang(wyutong@bu.edu)

Introduction

The purpose of this proposal is to establish our EC 544 semester project vision, direction, resource needed, risks, and milestones. The main issue we wish to address is a patient's burden of sorting and remembering to take medication. 50% of patients forget to take their pills at least once a month, and 75% patients do not take their pills properly. Our solution is to program a voice-controlled smart medication dispenser that will reliably dispense pre-sorted pills to the patient at specific times of the day. Our goal is to improve patient care by setting up visual and audio reminders via the dispenser, and subsequently alerting the patient's family member(s) if the pills have not been taken. The patient's medication history and progress will also be available to their doctor(s) for monitoring confidentially.

The smart medication dispenser will function as follows: the software will read in a list of medications assigned to a patient and their relevant details. At various times of the day, visual and audio cues will signal for the patient to come take their medication. In case the patient forgets, the dispenser will call and text to remind the patient, and subsequently his/her family members. Fingerprint authentication will be used to identify the patient before dispensing medication, though in our system that authentication will be emulated with a graphical UI-based "authenticate" button. A Mycroft voice assistant will communicate with the patient to confirm that the medications are taken. It will also double-check for restrictions, such as whether the patient has eaten before ingesting a pill. Finally, authenticated doctors will be able to extract encrypted confidential information from blockchain storage to monitor patient progress and use a chatbot to communicate with the patient if they desire.

We will be learning to implement the Mycroft open-source voice assistant to provide accessibility for the patient. We will use blockchain technology to confidentially and reliably store and share the patient information (and medication use) with their authorized physicians. We will use Twilio, Telegram, and AWS Lambda APIs to provide availability and reliability to our system's SMS/calling, chatbot, and systems integration functions. Implementing a small database, NodeJS server, and Mycroft locally on our Raspberry Pi systems will allow our system to provide basic local functionality without constant internet access. Encryption will be implemented in the data stored on the blockchain, but local data stored on the Raspberry Pi will likely not be encrypted due to time constraints in the project.

Resources

Hardware

Item	Use
Wired router with switch & ethernet cables	To connect all of the devices locally
Raspberry Pi 4B w/ 4GB RAM	Required for Mycroft, owned by Evelyn
Raspberry Pi 2	Required for non-Mycroft components, owned by Evelyn
Microphone & Speaker: Google AIY Voice Kit v1	Required for Mycroft - Matt purchased

Software

Item	Use
Mycroft: Picroft Raspbian Distro	For local voice interactions
Twilio free Account	API for SMS/Calling
Telegram chatbot	API for online chatbot (for physician)
AWS Lambda on free tier	Serverless computing to connect Raspberry Pi, Telegram, Twilio, and Blockchain
MariaDB	Database to store prescription information on Raspberry Pi
NodeJS + Express Server	Web and API server on Raspberry Pi to handle requests from AWS Lambda functions and display interface for user
GNU Arm Cross Compiler	To compile code for the FRDM board
IBM Hyperledger	To store medical “wallet” data
Private-public key encryption	For physician on Blockchain via Telegram
Github	Code repository and version control

Technical Risk Areas & Risk Management

We have identified 5 major areas of potential risk in our design. They are briefly listed below, along with the planned mitigation strategy for each.

Risk: Learning curve for Mycroft may be steep and we may overestimate its abilities in the context of our application.
Mitigation: Dedicate one team member to learn and implement Mycroft starting from week one. Leave our implementation flexible enough to allow changes once Mycroft's abilities are determined. Worst case scenario, use Telegram instead.

Risk: Mycroft not performing responsively on the Raspberry Pi.
Mitigation: Transition to running Mycroft in a well-resourced Linux virtual machine on one of our team member's laptops.

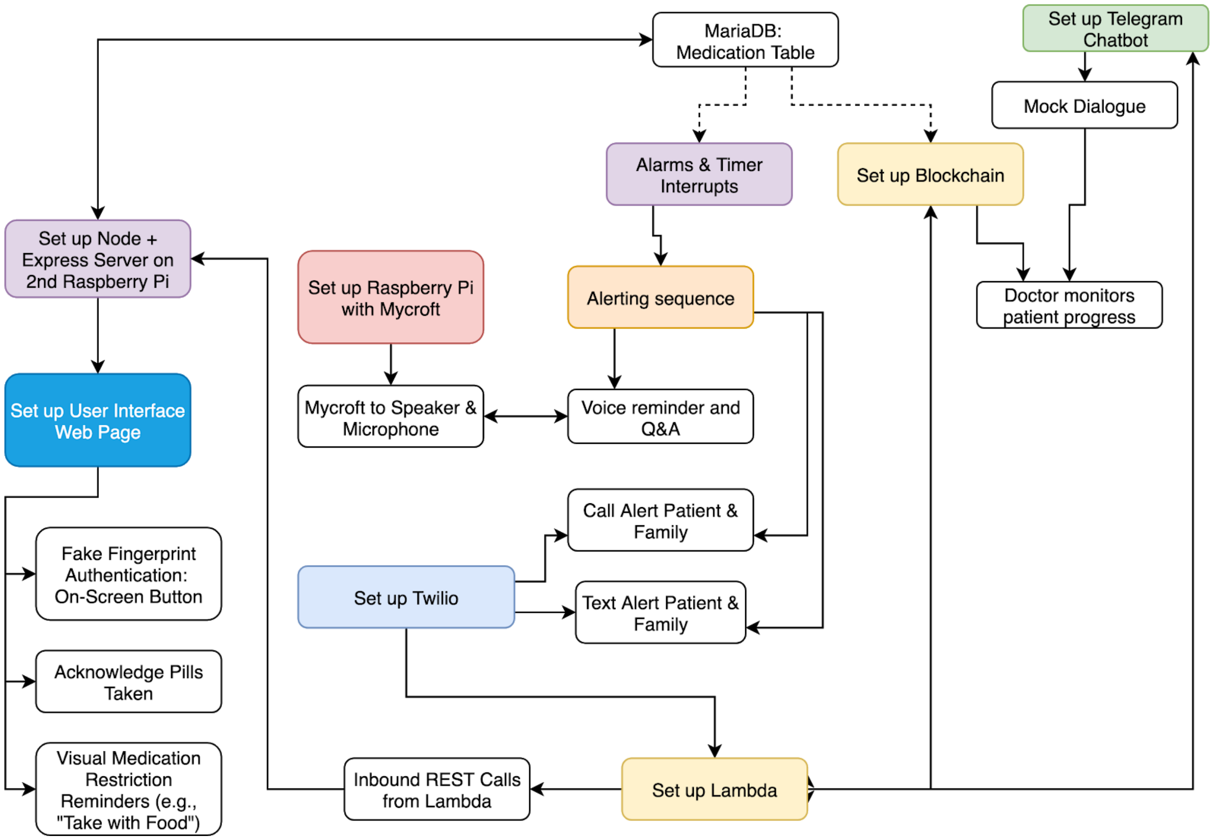
Risk: Integrating all of the cloud products to produce functional final results may prove more difficult than anticipated.
Mitigation: Dedicate two team members to learn and implement the cloud products starting from week one. Leave our implementation flexible enough to allow changes if integrating all of the cloud products is unsuccessful.

Risk: Connecting all cloud products to local software running on Raspberry Pi does not work as expected.
Mitigation: Plan to test this connectivity by week three so any challenges can be addressed quickly and changes can be made early.

Risk: Learning curve for Blockchain product may be steep and we may underestimate the amount of work required to implement it.
Mitigation: Dedicate one team member to learn and investigate the Blockchain product starting from week one. Leave our implementation flexible enough to limit the scope of use for the Blockchain product if it proves more complicated than expected.

Technical Approach

We will be prioritizing items of high risk, which includes the steep learning curve in Mycroft, complete lack of knowledge of Hyperledger, and the lack of familiarity with Telegram and AWS Lambda. As such, we are dedicating one team member to each of these three areas of concern to start working on them from week one. This will give us ample time to familiarize ourselves with these key technologies and potentially adjust our project scope if we discover they will not meet our needs as initially conceived. As a result, simpler items, such as setting up the webpage and visual reminders will be worked on last. Here is the section breakdown according to our flowchart.



MariaDB: Medication Table
Set up Telegram Chatbot
Mock Dialogue
Alarms & Timer Interrupts
Set up Blockchain
Set up Node + Express Server on 2nd Raspberry Pi
Set up Raspberry Pi with Mycroft
Alerting sequence
Doctor monitors patient progress
Set up User Interface Web Page
Mycroft to Speaker & Microphone
Voice reminder and Q&A
Call Alert Patient & Family
Fake Fingerprint Authentication: On-Screen Button
Set up Twilio
Text Alert Patient & Family
Acknowledge Pills Taken
Visual Medication Restriction Reminders (e.g., "Take with Food")
Inbound REST Calls from Lambda
Set up Lambda

Approach Overview & Team Members

Evelyn will be working on the chatbot feature via Telegram to retrieve information from the encrypted blockchain. She will also set up MariaDB on the Raspberry Pi to store patient medication data for access via Node. Finally, she will set up Amazon Lambda to interact with Telegram, Node, MariaDB, and get it ready to work with Hyperledger and Mycroft components. She has not worked with any of the above components, but will learn through tutorials.

Matt Boyd will be working specifically on Hyperledger, Twilio, alert scheduling on the Raspberry Pi via Node, connecting Node and Hyperledger to Lambda, and setting up the physical networking gear. He will also be working with teammate Evelyn to connect Telegram to Hyperledger via Lambda. Matt chose to work on Hyperledger because he is unfamiliar with any blockchain technology and wanted to learn about it in depth. Similarly, he has never used Twilio and wanted to learn how it works. Matt does have previous experience with networking, using APIs, and a significant amount of coding, so his knowledge will be helpful to get some of the necessary, but not “core” functions working together (i.e., networking and scheduling).

Yutong will be working on the Mycroft and the Google voice kit. He will burn the Picroft disk to Raspberry Pi and then connect the Raspberry Pi to the Google voice kit to realize the function of voice reminder. Then he will learn how to make the voice kit remind the user to eat pills on time. Yutong is a ME student. Learning how to use Raspberry Pi is very useful to his major. He also used the Raspberry Pi to do some programs before. But he never used the Mycroft before, and he wants to learn something about it.

After April 14th, all codes should be functioning. We will run through scenarios to test for bugs and troubleshoot issues we encounter. We also will spend some time working on the project write-up.

Telegram

An interface is necessary for doctors to access patient data stored on the blockchain. Evelyn will create a chatbot in Telegram so that doctors can ask questions regarding a patient, such as “Has Elena missed her medications in the last 2 weeks?” The bot will run in privacy mode and connect to Amazon Lambda to retrieve encrypted information from Hyperledger. Evelyn will start by setting up a Telegram developer’s account and request an authorization token for the Bot API via BotFather. Then, she will use her personal phone number to assume the faux identity of a doctor. She will then design a mock dialogue between the doctor and the chatbot to determine what Lambda functions and predefined commands are necessary to create. She will familiarize herself with the Bot API to implement the mock dialogue in a chat box. She will then have the bot call on Lambda functions according to user commands. Messages, commands, and requests can be encrypted using the Telegram API. Evelyn will work with Matt to connect Telegram to the Hyperledger via Amazon Lambda. Finally, if time allows, she will design a way to verify doctors’ identity before they access patient information on the chatbot.

Amazon Lambda

Evelyn will sign up for the “Lambda Free Tier,” which includes 1M free quests per month and 400,000 GB-seconds of compute time per month. We will be using Amazon Lambda for approximately 2 months. Evelyn will watch the tutorials for building simple lambda functions and read through Amazon’s technical documentation to learn how to develop and deploy them for an app in C# and Node.js. Evelyn will then set up an API Gateway using an IAM role with basic Lambda edge permissions. This will serve as the endpoint that receives Telegram user requests. The Lambda function will be integrated with the API and deployed. She will also set up webhooks to check confirmation messages from Telegram. She will test out her first Lambda function by coding for the bot to reply with the message it receives from a user. She will check the CloudWatch logs and Telegram chat box to make sure the Lambda functions are triggered appropriately.

MariaDB & Express.js

Evelyn will install MariaDB onto a Raspberry Pi 2’s Raspian, while the Raspberry Pi 4 is being used by Yutong to work on Mycroft. Online articles mention that MariaDB performs more efficiently than MySQL in Raspbian because of the Aria engine. She will then create medication tables to be loaded into MariaDB. She will install Node and Express.js and connect Node to MariaDB. Then, she will install the dependencies required for Express to connect to Amazon Lambda and prepare for deployment. She will then test and verify two-way communications between Node and Lambda.

Hyperledger

Setting up Hyperledger will be a significant challenge due to our complete lack of knowledge of blockchain technologies. Matt will first start by going through all of the tutorials available from IBM. He will then attempt to get his first Hyperledger system running in a VM. Once that is completed he will attempt to add “private storage” where the patient’s medication-taking history will be kept. He will then add permissions and perform local queries to demonstrate that only designated parties can see the medication information. Next will be ensuring the private data store is encrypted so it is not visible to outside users. The penultimate major step is adding the ability to query the private data store from outside the system (i.e., via a REST API call). And finally he will add the ability to insert medication transactions into the chain from a second node, the Raspberry Pi. Breaking the system implementation down into these smaller steps enables us to not only learn the system step-by-step, but to also ensure each piece is working correctly before adding additional complexity. It also facilitates the identification of any specific issues or shortcomings in our design as we test each step so we may adjust course as necessary along the way.

Twilio

We feel confident Twilio will meet our needs, it's just a matter of setup and integration. Similarly we feel confident Node will meet our needs as a UI, coordination, and scheduling server. Accordingly, Matt's responsibility for these tasks can take a bit of a backseat to the much bigger "question mark" that is setting up Hyperledger. We have decided that if we need to sacrifice some functionality because of time constraints, it's better to lose some of this peripheral functionality rather than have a partially working blockchain implementation.

Mock UI Screen

Evelyn will create a Node server web page to simulate the screen on the smart medication dispenser. She will add in a fake fingerprint authentication process just to trigger pill dispensing. The screen will capture and display vocal inquiries made by Mycroft to the patient in text format via Node and Lambda. It can also display the alerts, reminders, and pill names and other information retrieved from MariaDB.

Google AIY voice kit

Yutong will assemble the Google AIY voice kit and then connect to the Raspberry Pi by Picroft. Picroft supports Google AIY voice kit as a microphone to chat with people. When we set up our Picroft, we can directly make the Raspberry correspond to it. So, it's a good choice to use this voice kit as a microphone to realize the function of our pillpack.

Mycroft

Yutong will be responsible to Mycroft. Maybe it is a good challenge. First, he will burn the disk of Picroft to Raspberry Pi 4. Mycroft can be used as a good voice assistant. We won't use Google assistant even though we use Google AIY because Mycroft is open source, which will help us learn it easily. Second he will try to modify some existing codes to make it can remind people to eat pills. And then it can use the data stored in the blockchain to answer the question of patients and remind the patients. The data was received from the doctors.

Milestones

<table>
<tr><th>Date</th><th>Yutong</th><th>Evelyn</th><th>Matt</th></tr>
<tr><td>3/16</td><td colspan="3">● Agree on project vision and direction
● Research and decide on appropriate components:
○ Mycroft, IBM Hyperledger, Telegram, Twilio, AWS Lambda, Node
● Draft proposal & Distribute tasks</td></tr>
<tr><td>3/23</td><td>● Research Mycroft API
● Install Picroft on RPi and add Voice kit and verify working</td><td>● Create Telegram Chatbot and faux doctor account
● Design mock dialogue
● Setup Lambda account and service</td><td>● Primary Hyperledger components installed in Laptop VM
● Setup router to forward requests to RPi running Node & Hyperledger VM</td></tr>
<tr><td>3/30</td><td>● Make it be able to receive questions and return spoken result
● Make it have the ability to ask questions of the patient and process results</td><td>● Determine and create necessary Lambda functions for Telegram Chatbot
● Exchange messages successfully
● Mock medication tables on Excel</td><td>● Hyperledger permissions & roles defined
● Able to store patient medication data in Hyperledger
● Able to query patient medication data from Hyperledger via CLI</td></tr>
<tr><td>4/6</td><td>● Continue make it have the ability to ask questions of the patient and process results
● Add the ability for Mycroft to be invoked programmatically to say something to the patient.</td><td>● Install NodeJS and MariaDB on the 2nd Raspberry Pi
● Transfer medication data into MariaDB
● Test two-way communication between Node and Lambda</td><td>● Set up RPi to store patient medication data in Hyperledger
● Set up Twilio dev. account
● Connect Twilio to Lambda
● Set up Twilio SMS & Call
● Verify Twilio <=> Lambda communication</td></tr>
<tr><td rowspan="2">4/13</td><td rowspan="2">● Connect Mycroft to database and Node on 2nd Raspberry Pi</td><td>● Create Node server webpage (user UI):
○ Fake Fingerprint auth.
○ Let patient acknowledge pill is taken on screen
○ Display medication name / indications / restrictions</td><td>● Set up Node-based scheduling process for alerts</td></tr>
<tr><td colspan="2">● Connect Telegram to Hyperledger (via Lambda) and return medication data results</td></tr>
<tr><td>4/20</td><td>● Continued Mycroft testing</td><td>● Script testing and demo scenarios
● Start project paper</td><td>● Connect Node to Lambda for alerts and responses</td></tr>
<tr><td>4/27</td><td colspan="3">● Testing & Troubleshooting
● Paper completed</td></tr>
</table>