Technical Specification

Healthcare Record Application Implementing Blockchain 3rd Year Project

Traian Svinti 14432128

David Talan 14387991

Supervisor: Geoff Hamilton

Table of Contents

Introduction 3

1.1 Overview 3

1.2 Glossary 4

System Architecture 5

2.1 Web Application Architecture 5

High-Level Design 6

3.1 Context Diagram 6

3.2 Data Flow Diagram 6

Problems and Resolution 8

Problem 1 – Understanding of how Blockchain works 8

Problem 2 – Connecting Django with non-SQL database 8

Problem 3 – Setting up users in Django with a one-to-one information viewing 8

Problem 4 – Running Django, MongoDB and BigChainDB altogether 8

Problem 5 – Displaying information from the database for the specific person logged in. 8

Installation Guide 9

Start-Up Guide 10

Blockchain Node Set Up 10

Django Set Up 10

Introduction

1.1 Overview

In the 3rd year project the implementation that was decided was to create a Medical Database while implementing Blockchain technology that can be accessed from wherever it is needed while still being secure and free from any sort of attempted fraudulence. This document describes different features and how they were implemented to work as intended.

The main objective was to create a secure, user-friendly and appealing UI which could be accessed by the patient and modified only by a doctor.

This project proved to be very challenging as a lot of the major tools being used are new and the project is a unique idea with a unique implementation.

Description

- When the main page is accessed the patient/doctor is given an option to select "Existing Patient" or "New Patient".
    - When "Existing Patient" is accessed, a login page is returned to login as a patient and view records.
    - When "New Patient" is accessed, only a doctor (admin) may log in to create/edit patient information.
- Once logged in as a patient, information is shown is simple tabs such as General Information, Previous Doctors, Illnesses and Medication.
    - This information is all retrieved through Django, our python backend, from the database MongoDB. The queries from Django which are SQL are translated for the non-SQL database that is MongoDB using Djongo. MongoDB is connected to the Blockchain database, BigChainDB.
- If a doctor/admin wants to log in, they are presented with a admin login page where once they are logged in can add new patients and edit patient information.

1.2 Glossary

- Blockchain:
    - These are blocks of information stored in a database and all the blocks are linked through hash chains.

- BigChainDB:
    - This is the database we will use to store the blockchain.

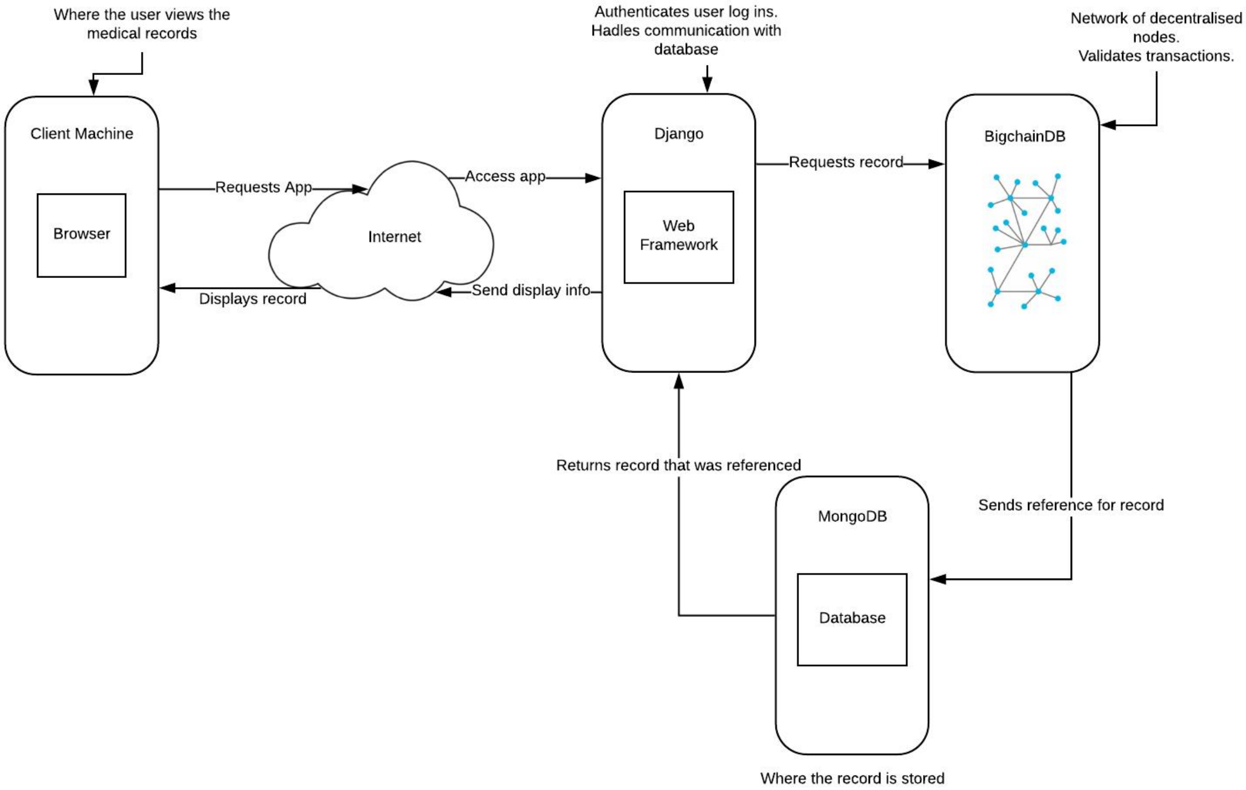
- MongoDB:
    - The non-SQL database used to communicate with the UI and BigChainDB.

- Django:
    - The python backend used to communicate with the UI and the database, MongoDB

- Djongo:
    - As Django is only compatible with SQL database Djongo translates the queries for our non-SQL database, MongoDB

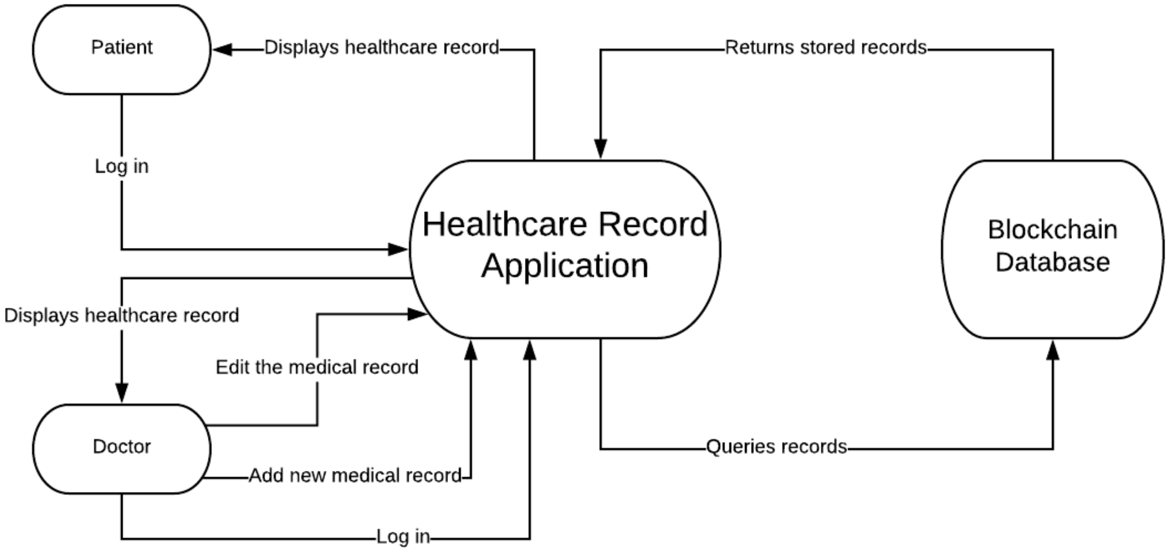
System Architecture

2.1 Web Application Architecture


Where the user views the medical records
Client Machine
Browser
Requests App
Displays record
Internet
Access app
Send display info
Authenticates user log ins. Hadles communication with database
Django
Web Framework
Requests record
Network of decentralised nodes. Validates transactions.
BigchainDB
Returns record that was referenced
Sends reference for record
MongoDB
Database
Where the record is stored

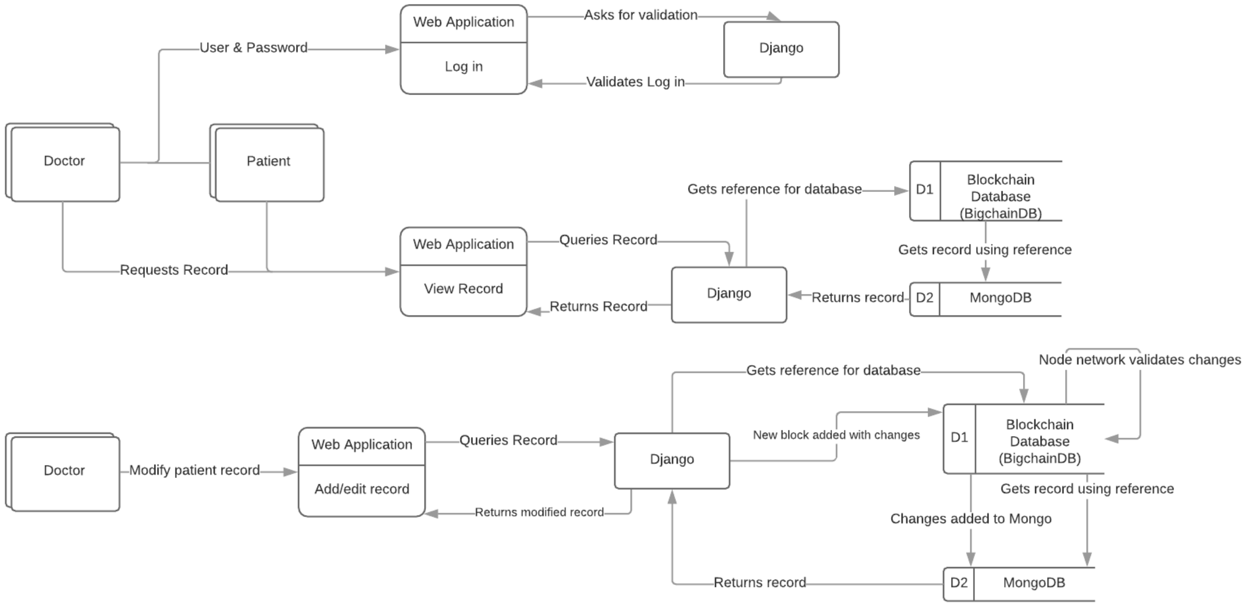
High-Level Design

3.1 Context Diagram



3.2 Data Flow Diagram

DATA FLOW DIAGRAM



Problems and Resolution

Problem 1 – Understanding of how Blockchain works

The first problem encountered was the misconception of how blockchain works and is linked to the main database.

Solution

Once more information was researched and discovered the implementation was restructured.

Problem 2 – Connecting Django with non-SQL database

When setting up Django, it was tested with a build in SQL database. It proved to be a major problem to change databases as no streamlined way of doing this is available.

Solution

With lots of continued research and connecting errors, Djongo was found. Djongo translates the SQL queries from Django to MongoDB readable queries.

Problem 3 – Setting up users in Django with a one-to-one information viewing

Setting up user profiles that could only be accessed by the user logged in and a profile to be automatically created and linked once a user is created.

Solution

After more research and testing/debugging was done eventually this was incorporated very well.

Problem 4 – Running Django, MongoDB and BigChainDB altogether

To run BigChainDB, MongoDB must be running. Django was now connected to MongoDB through Djongo but BigChainDB would not run alongside all the other components.

Solution

After extensive research, the group found out that this could not be currently done. This is due to the fact that BigChainDB is relatively new and there are not enough resources and information for this implementation.

Problem 5 – Displaying information from the database for the specific person logged in.

Once Django was set up, displaying users information onto the UI proved to be a problem as it was not the same as the general way of displaying without the profiles.

Solution

Tweaking the user profiles and the views.py file found in Django, the group eventually found a solution for this with only the information of the person who is logged in being displayed.

Installation Guide

Below is a list of all tools needed to run the system

- *All installs use Python 3.6*

Python 3.6

1. sudo apt-get install software-properties-common python-software-properties
2. sudo add-apt-repository ppa:jonathonf/python-3.6
3. sudo apt-get update
4. sudo apt-get install python3.6

PIP

1. sudo apt-get install -y python3-pip

BigChainDB

1. sudo apt-get install libffi-dev
2. sudo apt-get install libssl-dev
3. sudo python3.6 -m pip install –upgrade setuptools
4. sudo python3.6 -m pip install bigchaindb_driver

MongoDB

1. sudo apt-key adv –keyserver hkp://keyserver.ubuntu.com:80 –recv 2930ADAE8CAF5059EE73BB4B58712A2291FA4AD5
2. echo "deb [arch=amd64,arm64] https://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.6 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.6.list
3. sudo apt-get install -y mongodb-org

Django

1. sudo python3.6 -m pip install django

Djongo

1. sudo python3.6 -m pip install djongo

DjangoToolBox

1. sudo python3.6 -m pip install git+https://github.com/django-nonrel/djangotoolbox

Start-Up Guide

Blockchain Node Set Up

Open terminal - sudo mongod –replSet=bigchain-rs

Open new terminal - bigchaindb -y configure mongodb

- bigchaindb start

Open new terminal - cd medicaldatabaseimplementingblockchain/code/Blockchain

- python3 datab.py - this adds a block into the blockchain with the information inside the code in JSON format (‘data’). This also displays a long hashcode on the terminal. Copy the first hashcode.
- Open browser – url ‘127.0.0.1:9984/api/v1/transactions/(paste hashcode here)’
- This is the web api BigchainDB provides that lets you view the data that’s been added to the blockchain

Django Set Up

Open terminal – cd 2018-CA326-tsvinti-medicaldatabaseimplementingblockchain/code/medicalDatabase

- python3.6 manage.py makemigrations userprofile
- python3.6 manage.py migrate
- python3.6 manage.py runserver

Access url – http://127.0.0.1:8000/personalInfo