

Deployment Manual

Setup GOSH FHIR API

Please review the following deployment guide link which contains the instructions and source code for the FHIR standard access at Great Ormond St Hospital:

https://github.com/goshdrive/FHIRworks_2020

Follow the instructions in the README of the above Git repo to set up the GOSH FHIR API on your localhost.

Credentials for the above .Net Application

"Instance": "https://login.microsoftonline.com/{0}",

"Tenant": "ca254449-06ec-4e1d-a3c9-f8b84e2afe3f",

"ClientId": "0f6332f4-c060-49fc-bcf6-548982d56569",

"ClientSecret": "ux@CJAaxCD85A9psm-Wdb?x3/Z4c6gp9",

"BaseAddress": "https://gosh-fhir-synth.azurehealthcareapis.com",

"Scope": <https://gosh-fhir-synth.azurehealthcareapis.com/.default>

Creating an Azure account

The next step needed is to create an Azure account. Azure is used in this project as an area to store all the generated Word documents containing patient data.

Visit [this link](#) and click 'Start for free' to create your own Azure account.

Sign in with your Microsoft or GitHub account or create a free Microsoft account.

On the About you page, select your correct country or region. Enter your first and last name, email address, and phone number. Depending on your country, you might see additional fields, such as a VAT number. Select Next to continue.

On the Identity verification by phone screen, select your country code, and type the number of a telephone to which you have immediate access.

You have the option of text or call back to obtain a verification code. Select the relevant button, type the code in the Verification code box, and select Verify code.

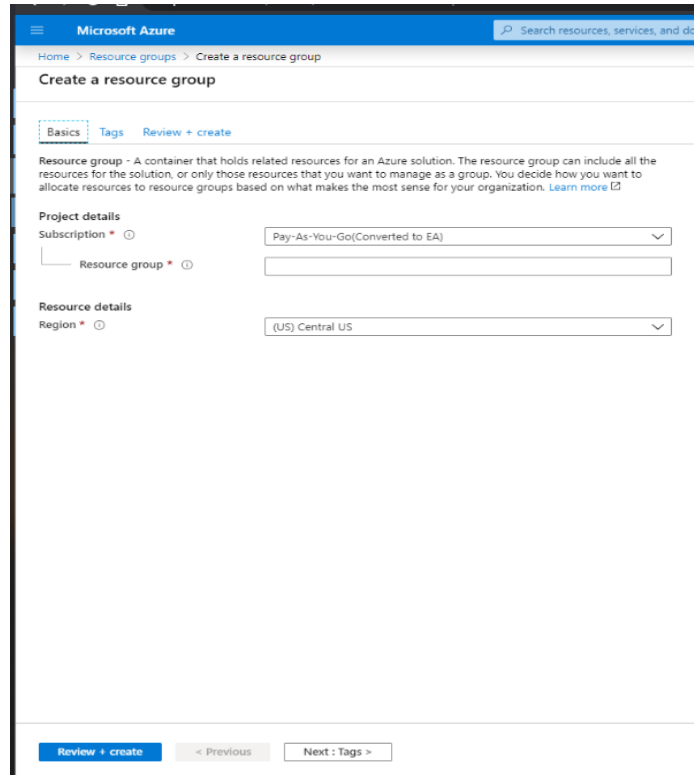
If the verification code is correct, you're asked to enter details of a valid credit card. You will not be charged anything for signing up to Azure as Azure operates on a pay as you go basis. Azure also provides you with £150 of free credit when you sign up for the first 30 days. Enter the card information and select Next.

The last step is to review the agreement and privacy statement then select Sign up.

You have now successfully set up a free account on Azure and should be on the Azure portal home page.

Create a Resource Group

Now you have an Azure account the next step is to set up a resource group. A resource group is a collection of resources that share the same lifecycle, permissions, and policies. To create a resource group first click on 'Resource groups' on the Azure portal home page. Then click on 'Create resource group'. Choose a subscription to use and enter the name you want to call your resource group into the field called 'Resource group'. Then for the field 'Resource details Region', select the appropriate region for you from the drop-down list. Then click 'Review and create', followed by 'Create'.

The screenshot shows the 'Create a resource group' page in the Microsoft Azure portal. The page has a blue header with the 'Microsoft Azure' logo and a search bar. Below the header, the breadcrumb trail is 'Home > Resource groups > Create a resource group'. The main heading is 'Create a resource group'. There are three tabs: 'Basics' (selected), 'Tags', and 'Review + create'. A descriptive paragraph explains that a resource group is a container for related resources. The 'Project details' section includes a 'Subscription' dropdown menu set to 'Pay-As-You-Go(Converted to EA)' and a text input field for the 'Resource group' name. The 'Resource details' section includes a 'Region' dropdown menu set to '(US) Central US'. At the bottom, there are three buttons: 'Review + create' (highlighted in blue), '< Previous', and 'Next : Tags >'.

Microsoft Azure

Home > Resource groups > Create a resource group

Create a resource group

Basics Tags Review + create

Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. [Learn more](#)

Project details

Subscription * ⓘ Pay-As-You-Go(Converted to EA) ▼

Resource group * ⓘ

Resource details

Region * ⓘ (US) Central US ▼

Review + create < Previous Next : Tags >

Creating Azure blob storage container

[This link](#) contains instructions on how to set up a storage account on Azure but there are a few changes that need to be done. In the section 'Create a storage account', on step 7, for the field 'Account kind' don't leave it as 'StorageV2 (general-purpose v2)'. Instead change this to be BlobStorage. Step 8 can be skipped.

Other than the above changes, follow the steps in the link to create your storage account. Once you have done this go to the Overview page in your storage account resource. Click on 'Containers'. You then need to create a new container by clicking on the '+ Container' option. Name the container 'feedbackforms'. This is where documents for collecting patient feedback will be stored. Add another container called 'patienthealthdata'. This is where documents with visualisations for patient health data will be stored.

You now have a storage account set up on Azure.

Fill in template code with storage account details

GUI.py

In the folder 'PatientDocumentGenerator' open GUI.py in an IDE or text editor. On lines 9 and 10 you need to replace 'storage_account_name' and 'storage_account_key' with the name of your storage account and its key respectively. When you are in the storage account resource, down the left under settings you will find an option called Access keys. Choose key1 as your storage account key.

In the folder 'PatientDocumentAPI' open AzureBlobStorage.py in an IDE or text editor. On lines 4 and 5 you need to again replace 'storage_account_name' and 'storage_account_key' with the name of your storage account and its key respectively. These should be the same as the ones in GUI.py

User Manual

Abstract

This project is something that would be used by GP clinics. Each clinic would set up their own Azure storage account and set up their own API that would use the details of their storage account. They could then use the GUI to type in patient IDs and get back a document they could print off and hand to the patient to collect patient feedback with. They could also generate a document with visualisations of the patient's health data such as weight, heart rate, BMI and respiratory rate.

Endpoints

/FormFiller/feedback?id=patient_id

This endpoint will find the name and address of the patient whose ID was provided and will create a Word document asking that patient for feedback on the services they received. This Word document will then be uploaded to the Azure account and is stored there. The endpoint will return JSON containing the patient data and questions used to make the patient feedback document as well as creating the document on Azure.

/FormFiller/feedbackDocumentData?id=patient_id

This endpoint will return JSON containing patient data and questions needed to generate a Document or form asking that patient for feedback on the services provided to them. It will not generate the word document and store it on Azure like the above endpoint.

/report/rawData?id=patient_id

This endpoint will return JSON regarding patient health data. It will return the values of readings, date of those readings and units of the readings for weight, BMI, heart rate and respiratory rate.

/report/patientReport?id=patient_id

This endpoint not only returns JSON containing information on the readings and dates of those readings for patient health data regarding weight, heart rate, BMI and respiratory rate, but also creates a word document containing visualisations of the data and saves it to an Azure storage

Running the project

If you followed the steps in the deployment manual then you should have the GOSH FHIR API running on localhost:5001. If not please run this API by following the instructions in the README of the Git repo provided at the start of the deployment manual.

You then need to open a terminal/command prompt in the 'PatientDocumentAPI' directory. Type 'python FormAPI.py' and hit enter to run the document generator API on localhost:5010.

Next open a terminal/command prompt in the 'PatientDocumentGenerator' directory. Type 'python GUI.py' and hit enter to run the GUI tool. In the patient ID field enter a patient ID such as:
8f789d0b-3145-4cf2-8504-13159edaa747

You can then generate files that will be stored on azure or get files already stored on azure and download them and store them in the 'PatientDocumentGenerator' directory.

Examples of what these files are provided.