Kickstarting a charity with Serverless Technologies

James Randall

About me

https://github.com/JamesRandall

https://github.com/JamesRandall

https://www.azurefromthetrenches.com

Bookmark

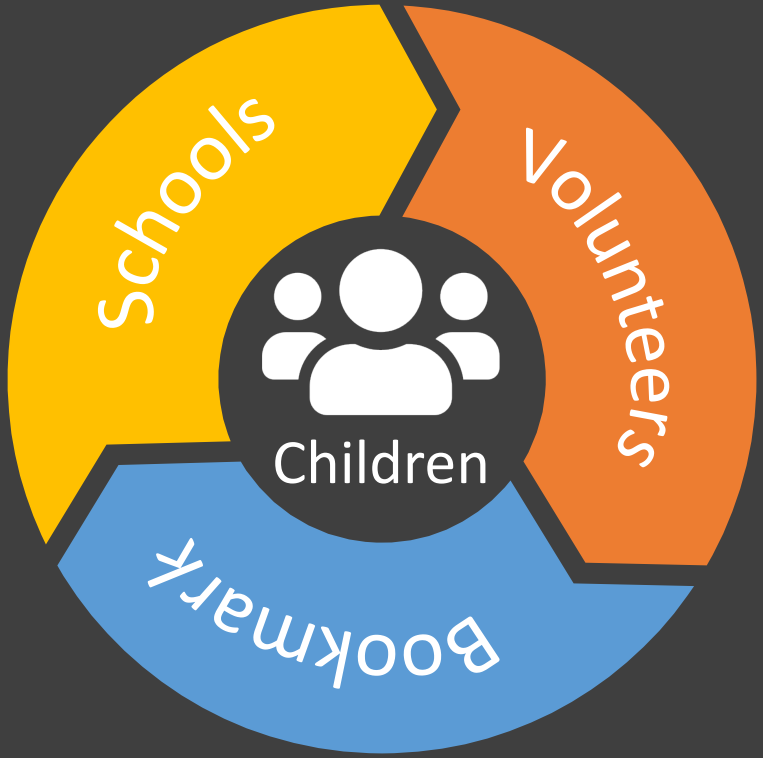
Hello, we're Bookmark.
We want every child to read.

IN AN AVERAGE CLASS, 8 CHILDREN LEAVE PRIMARY SCHOOL UNABLE TO READ WELL.

These children often struggle in school, and beyond. Reading isn't just about books. It's about reading a road sign, a safety manual, a birthday card. It's understanding a job application and the prescription that could help save your life.

The problem

- Connect schools, volunteers and Bookmark staff
- Three main areas to address:
  - Discovery
    - Schools finding volunteers
    - Volunteers finding schools
  - Scheduling
    - People live busy lives
    - Schools have constraints
  - Safety



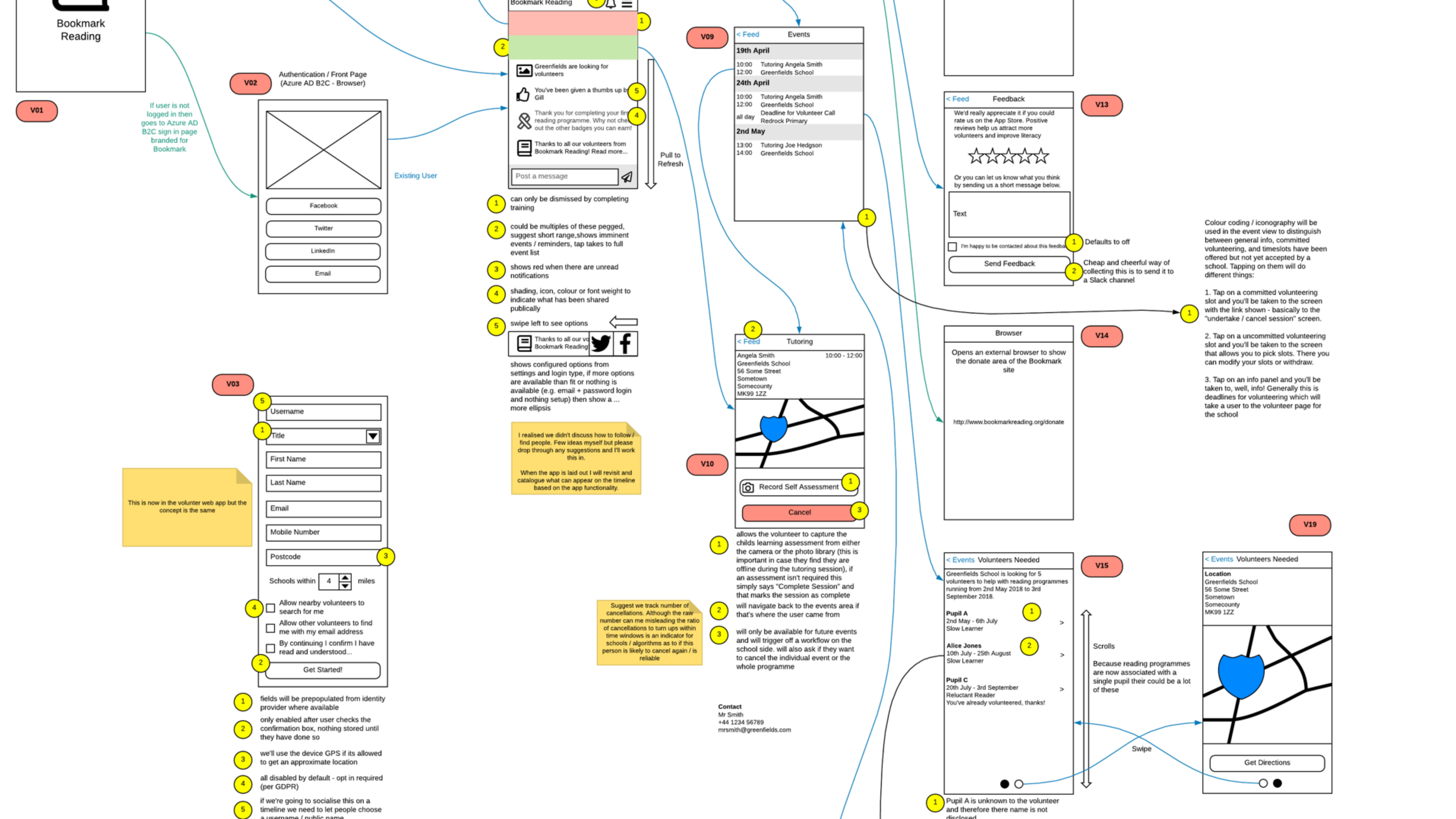
Constraints

- A capped and small development budget
- A small operational budget
- A small development team – for the most part: me
- A small operational team – for the most part: me
- A none-technical internal and external audience
  - Bookmark “back office”
  - Schools
  - Volunteers
- A deadline – we needed to be available inside of 5 months to beta in the upcoming school year

Beginnings

- We started with the front end – what were we building
- Fairly typical wireframing and storyboard process
- These were worked through with potential early adopters
- This teased out the main domains

- It also highlighted there was a lot of work to do!



If user is not logged in then goes to Azure AD B2C sign in page branded for Bookmark

Authentication / Front Page (Azure AD B2C - Browser)

Facebook

Twitter

LinkedIn

Email

Bookmark Reading

Greenfields are looking for volunteers

You've been given a thumbs up by Gill

Thank you for completing your first reading programme. Why not check out the other badges you can earn!

Thanks to all our volunteers from Bookmark Reading! Read more...

Post a message

1. can only be dismissed by completing training
2. could be multiples of these pegged, suggest short range,shows imminent events / reminders, tap takes to full event list
3. shows red when there are unread notifications
4. shading, icon, colour or font weight to indicate what has been shared publically
5. swipe left to see options

Thanks to all our vo Bookmark Reading

shows configured options from settings and login type, if more options are available than fit or nothing is available (e.g. email + password login and nothing setup) then show a ... more ellipsis

I realised we didn't discuss how to follow / find people. Few ideas myself but please drop through any suggestions and I'll work this in.

When the app is laid out I will revisit and catalogue what can appear on the timeline based on the app functionality.

This is now in the volunter web app but the concept is the same

Username

Title

First Name

Last Name

Email

Mobile Number

Postcode

Schools within 4 miles

□ Allow nearby volunteers to search for me

□ Allow other volunteers to find me with my email address

□ By continuing I confirm I have read and understood...

Get Started!

1. fields will be prepopulated from identity provider where available
2. only enabled after user checks the confirmation box, nothing stored until they have done so
3. we'll use the device GPS if its allowed to get an approximate location
4. all disabled by default - opt in required (per GDPR)
5. if we're going to socialise this on a timeline we need to let people choose a username / public name

< Feed Events

19th April

10:00 Tutoring Angela Smith
12:00 Greenfields School

24th April

10:00 Tutoring Angela Smith
12:00 Greenfields School
all day Deadline for Volunteer Call Redrock Primary

2nd May

13:00 Tutoring Joe Hedgson
14:00 Greenfields School

< Feed Tutoring

Angela Smith 10:00 - 12:00
Greenfields School
56 Some Street
Sometown
Somecounty
MK99 1ZZ

Record Self Assessment

Cancel

1. allows the volunteer to capture the childs learning assessment from either the camera or the photo library (this is important in case they find they are offline during the tutoring session), if an assessment isn't required this simply says "Complete Session" and that marks the session as complete
2. will navigate back to the events area if that's where the user came from
3. will only be available for future events and will trigger off a workflow on the school side. will also ask if they want to cancel the individual event or the whole programme

Suggest we track number of cancellations. Although the raw number can me misleading the ratio of cancellations to turn ups within time windows is an indicator for schools / algorithms as to if this person is likely to cancel again / is reliable

Contact
Mr Smith
+44 1234 56789
mrsmith@greenfields.com

< Feed Feedback

We'd really appreciate it if you could rate us on the App Store. Positive reviews help us attract more volunteers and improve literacy

☆☆☆☆☆

Or you can let us know what you think by sending us a short message below.

Text

□ I'm happy to be contacted about this feedba

Send Feedback

1. Defaults to off
2. Cheap and cheerful way of collecting this is to send it to a Slack channel

Browser

Opens an external browser to show the donate area of the Bookmark site

http://www.bookmarkreading.org/donate

Colour coding / iconography will be used in the event view to distinguish between general info, committed volunteering, and timeslots have been offered but not yet accepted by a school. Tapping on them will do different things:

1. Tap on a committed volunteering slot and you'll be taken to the screen with the link shown - basically to the "undertake / cancel session" screen.

2. Tap on a uncommitted volunteering slot and you'll be taken to the screen that allows you to pick slots. There you can modify your slots or withdraw.

3. Tap on an info panel and you'll be taken to, well, info! Generally this is deadlines for volunteering which will take a user to the volunteer page for the school

< Events Volunteers Needed

Greenfields School is looking for 5 volunteers to help with reading programmes running from 2nd May 2018 to 3rd September 2018.

Pupil A
2nd May - 6th July
Slow Learner >

Alice Jones
10th July - 25th August
Slow Learner >

Pupil C
20th July - 3rd September
Reluctant Reader >
You've already volunteered, thanks!

Because reading programmes are now associated with a single pupil their could be a lot of these

1. Pupil A is unknown to the volunteer and therefore there name is not disclosed

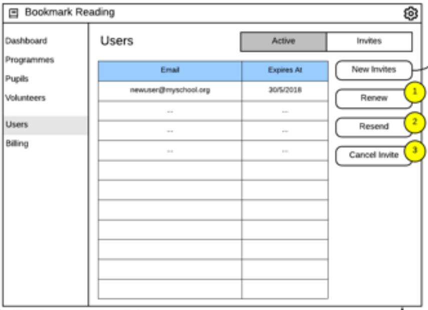
< Events Volunteers Needed

Location
Greenfields School
56 Some Street
Sometown
Somecounty
MK99 1ZZ

Get Directions

S21

This whole process is heavily audited.
The approach is one I've found to work well with schools in the past.



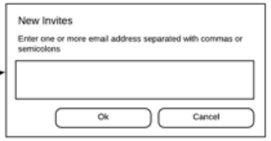
Email	Expires At
newuser@myschool.org	30/5/2018
...	...
...	...
...	...

1 Adds 30 days onto the selected invite

2 Sends a new email with the invite link

3 Cancels the invite and removes it from this list

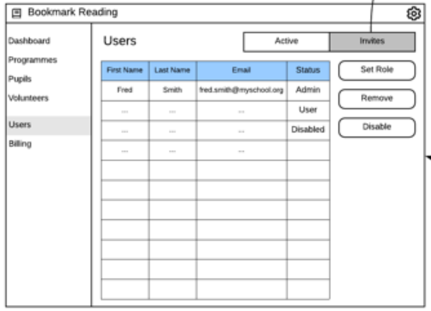
This will result in an invite link being sent to the supplied emails. Users will only be able to sign up with an identity mapped to the email entered here.



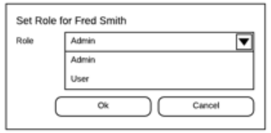
Enter one or more email address separated with commas or semicolons

S22

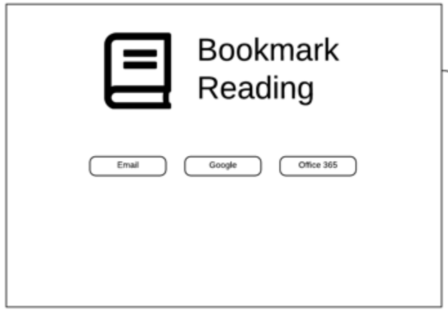
S20



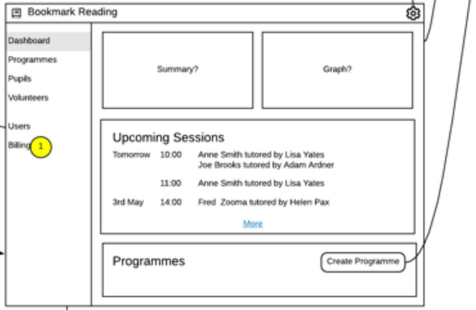
First Name	Last Name	Email	Status
Fred	Smith	fred.smith@myschool.org	Admin
...	...	...	User
...	...	...	Disabled
...	...	...	



Role: Admin



S01

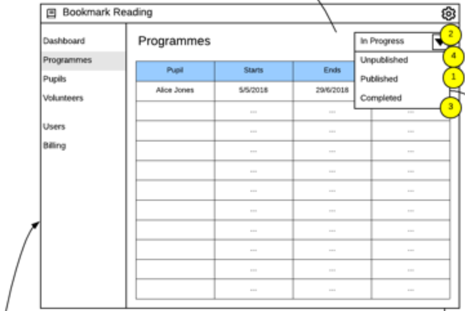


Upcoming Sessions

Tomorrow 10:00 Anne Smith tutored by Lisa Yates
Joe Brooks tutored by Adam Ardner
11:00 Anne Smith tutored by Lisa Yates
3rd May 14:00 Fred Zooma tutored by Helen Pax

1 We'll flesh billing out when we've picked a provider as its integration capabilities will inform this area

S12



Pupil	Starts	Ends	
Alice Jones	5/5/2018	29/6/2018	
	...	...	...
	...	...	...
	...	...	...
	...	...	...
	...	...	...
	...	...	...
	...	...	...
	...	...	...
	...	...	...
	...	...	...

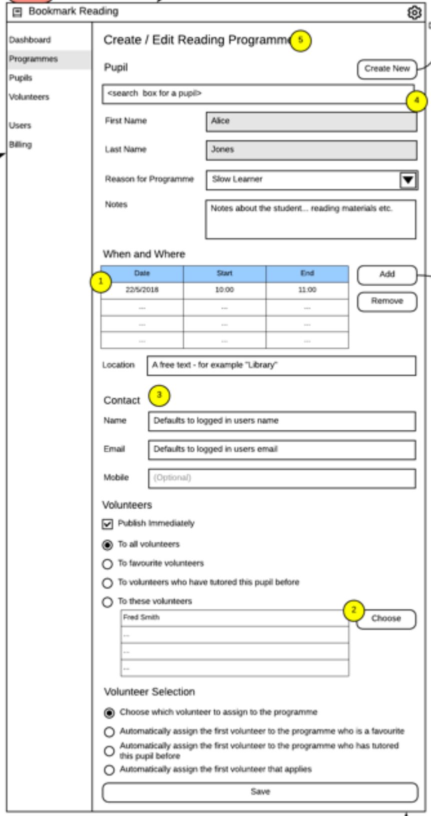
1 A programme is in progress when the volunteer call has ended and volunteers have been assigned to pupils but at least one session has not been marked as attended or cancelled

2 A programme is defined as being published if it has been setup for volunteers to respond to with their interest but no volunteers have yet been assigned.

3 A completed programme is defined as one in which all sessions have been marked as attended or cancelled

4 An unpublsihed programme has been set up but not yet shared with volunteers

S09



Create / Edit Reading Programme 5

Pupil

<search box for a pupil>

First Name: Alice
Last Name: Jones
Reason for Programme: Slow Learner
Notes: Notes about the student... reading materials etc.

When and Where

Date	Start	End
22/5/2018	10:00	11:00
...	...	...
...	...	...
...	...	...

Location: A free text - for example "Library"

Contact

Name: Defaults to logged in users name
Email: Defaults to logged in users email
Mobile: (Optional)

Volunteers

☑ Publish Immediately
● To all volunteers
○ To favourite volunteers
○ To volunteers who have tutored this pupil before
○ To these volunteers

Fred Smith
...
...
...

Volunteer Selection

● Choose which volunteer to assign to the programme
○ Automatically assign the first volunteer to the programme who is a favourite
○ Automatically assign the first volunteer to the programme who has tutored this pupil before
○ Automatically assign the first volunteer that applies

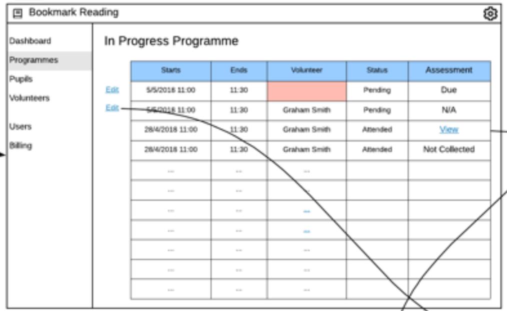
Dialog / Overlay

3 Cancels the programme. I've thought a lot about schedule editing but it causes a barrel load of complexity for *volunteers* - as it impacts choices they've already made and essentially you need to cancel their commitment and ask them to start again. Need to discuss. We could *add* new slots without issue.

4 Lets people flick through the volunteers and choose one for the programme to start with.

S14

This will be tidied up as we work on the app, illustrative for now as grids are limited in functionality when wireframing

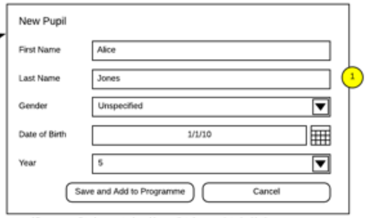


Starts	Ends	Volunteer	Status	Assessment
5/5/2018 11:00	11:30		Pending	Due
5/5/2018 11:00	11:30	Graham Smith	Pending	N/A
28/4/2018 11:00	11:30	Graham Smith	Attended	View
28/4/2018 11:00	11:30	Graham Smith	Attended	Not Collected
...	...	...		
...	...			
...	...	...		
...	...	...		
...	...	...		
...	...	...		
...	...	...		

We need to talk around the workflow of removing / replacing volunteers. And what happens if a replacement can't be found? Do we allow cancelling? Presumably as we need to indicate somehow that it can't happen.

Althgouh the idea is that n sessions are always run term breaks and "real life" might mean that some don't get run to completion.

An assumption here is that if a volunteer themselves cancels the same "publishing" settings will be used when the programme was set up to try and find a volunteer. If a user wants to do something different they can use the edit feature.



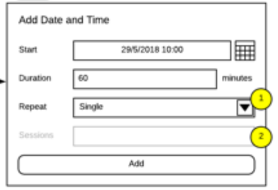
First Name: Alice
Last Name: Jones
Gender: Unspecified
Date of Birth: 1/1/10
Year: 5

1 Allows a pupil to be created and immediately associated with the programme. Fields are illustrative but a match of those in the main add pupil screen

S11

Dialog / Overlay

S10



Start: 29/5/2018 10:00
Duration: 60 minutes
Repeat: Single
Sessions:

1 Allows multiple repeating sessions to be added. Choices are single (just one session), daily excluding weekends, or weekly

2 Only enabled when repeat is not single

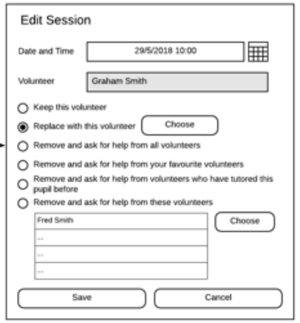
1 These are the slots the volunteers can pick from for this pupil

2 Disabled / hidden unless the user chooses to pick volunteers by hand by selecting "To these volunteers"

3 In a vNext we could look to route this through the applications but concerned that's a whole other system if we try to do it now.

4 This is a google-esque search box that allows a user to quickly find a pupil from any of their attributes. When yound there details appear below.

5 This screen is also used to edit unpublished programmes. Titles change appropriately.



Date and Time: 29/5/2018 10:00
Volunteer: Graham Smith

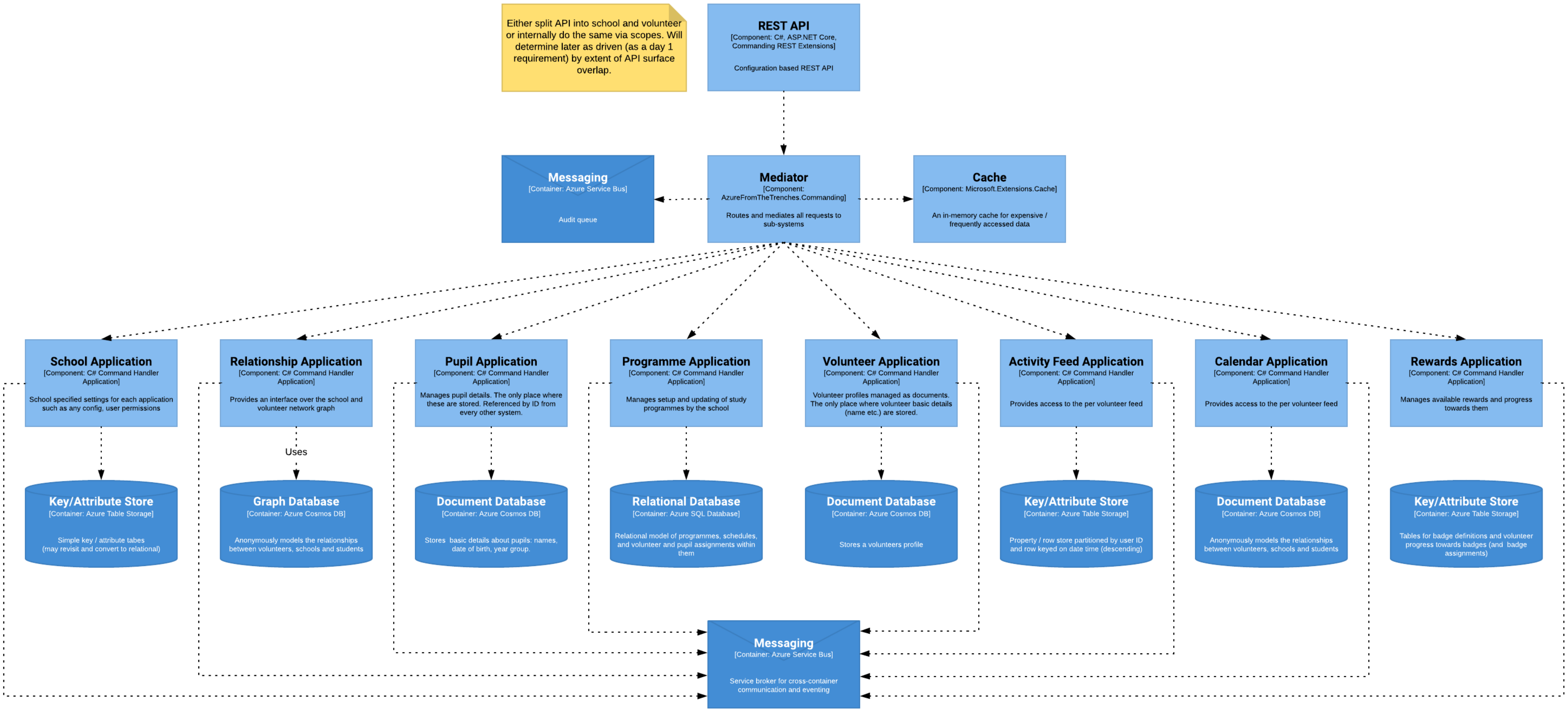
○ Keep this volunteer
● Replace with this volunteer
○ Remove and ask for help from all volunteers
○ Remove and ask for help from your favourite volunteers
○ Remove and ask for help from volunteers who have tutored this pupil before
○ Remove and ask for help from these volunteers

Fred Smith
...
...
...

S19

Modular Monoliths

- I wanted, needed, to ”have my cake and eat it”
    - Simple coding in a single codebase with low operational overhead and great support from tools
    - Strict separation of concerns and clear demarcation between bounded contexts
    - The capability to decompose later
- Subsystems broken down using Domain Driven Design
- The crossing of bounded contexts always takes place via a command



Either split API into school and volunteer or internally do the same via scopes. Will determine later as driven (as a day 1 requirement) by extent of API surface overlap.

REST API
[Component: C#, ASP.NET Core, Commanding REST Extensions]
Configuration based REST API

Messaging
[Container: Azure Service Bus]
Audit queue

Mediator
[Component: AzureFromTheTrenches.Commanding]
Routes and mediates all requests to sub-systems

Cache
[Component: Microsoft.Extensions.Cache]
An in-memory cache for expensive / frequently accessed data

School Application
[Component: C# Command Handler Application]
School specified settings for each application such as any config, user permissions

Relationship Application
[Component: C# Command Handler Application]
Provides an interface over the school and volunteer network graph

Pupil Application
[Component: C# Command Handler Application]
Manages pupil details. The only place where these are stored. Referenced by ID from every other system.

Programme Application
[Component: C# Command Handler Application]
Manages setup and updating of study programmes by the school

Volunteer Application
[Component: C# Command Handler Application]
Volunteer profiles managed as documents. The only place where volunteer basic details (name etc.) are stored.

Activity Feed Application
[Component: C# Command Handler Application]
Provides access to the per volunteer feed

Calendar Application
[Component: C# Command Handler Application]
Provides access to the per volunteer feed

Rewards Application
[Component: C# Command Handler Application]
Manages available rewards and progress towards them

Uses

Key/Attribute Store
[Container: Azure Table Storage]
Simple key / attribute tabes
(may revisit and convert to relational)

Graph Database
[Container: Azure Cosmos DB]
Anonymously models the relationships between volunteers, schools and students

Document Database
[Container: Azure Cosmos DB]
Stores basic details about pupils: names, date of birth, year group.

Relational Database
[Container: Azure SQL Database]
Relational model of programmes, schedules, and volunteer and pupil assignments within them

Document Database
[Container: Azure Cosmos DB]
Stores a volunteers profile

Key/Attribute Store
[Container: Azure Table Storage]
Property / row store partitioned by user ID and row keyed on date time (descending)

Document Database
[Container: Azure Cosmos DB]
Anonymously models the relationships between volunteers, schools and students

Key/Attribute Store
[Container: Azure Table Storage]
Tables for badge definitions and volunteer progress towards badges (and badge assignments)

Messaging
[Container: Azure Service Bus]
Service broker for cross-container communication and eventing

Component Model for API Application

Managed by [Container: API Application] - WORK IN PROGRESS, INCOMPLETE
Last modified: Monday 30th April 2018

Step in Serverless

- Pay for what you use
- Scale to billable zero
- Event based programming model
- Highly managed with low operational overhead
- We used a 100% serverless compute platform

No Silver Bullet

- We combined it with a constant focus on high value implementation patterns

Serverless Compute

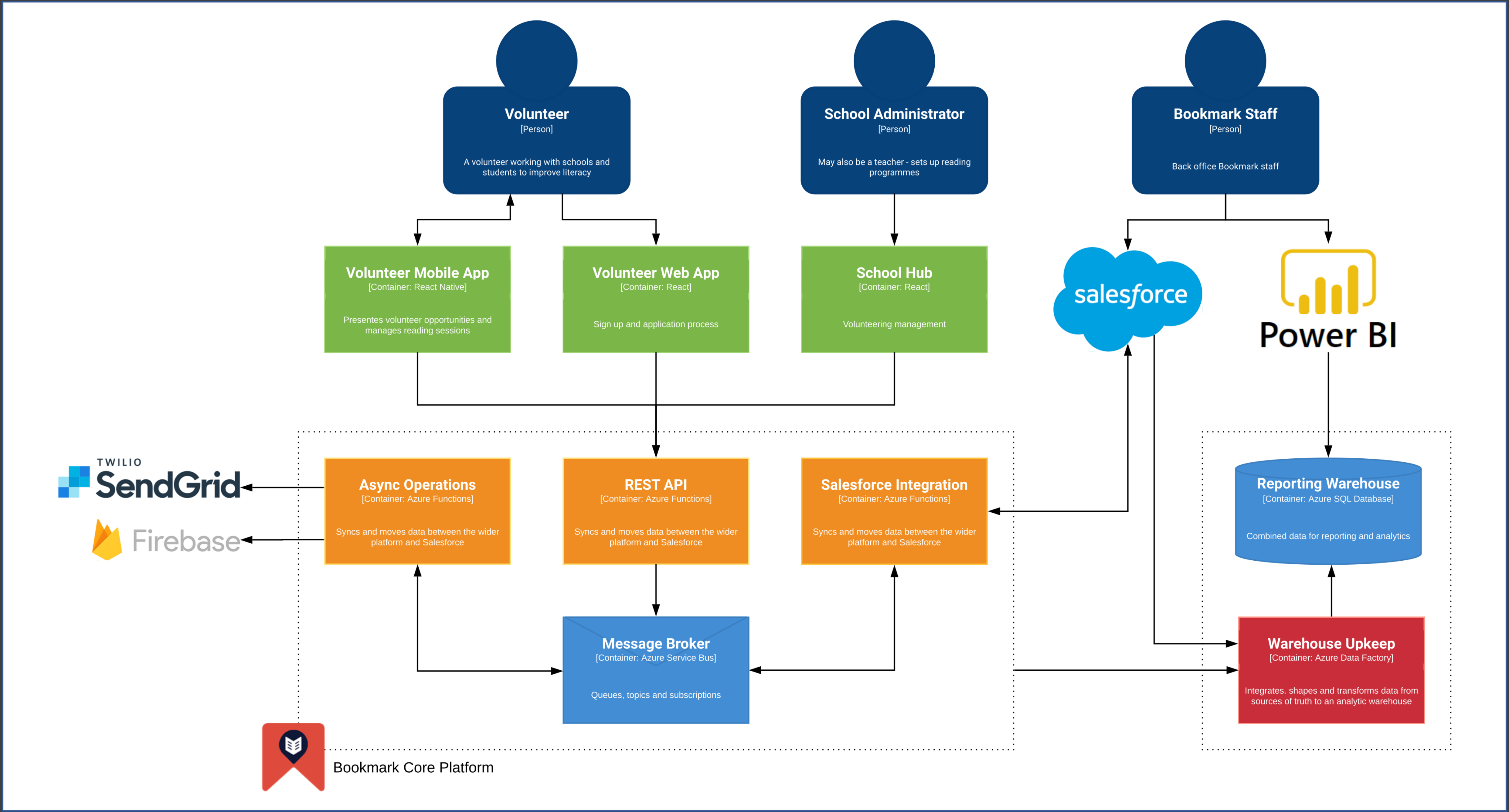
- Azure Functions
- Logic Apps
- Data Factory

Storage

- Azure Storage (blob and table)
- Cosmos DB (graph and document / SQL)
- Azure SQL Database

Other Services

- Application Insights
- Azure DevOps
- Azure Key Vault
- Auth0



Volunteer
[Person]
A volunteer working with schools and students to improve literacy

School Administrator
[Person]
May also be a teacher - sets up reading programmes

Bookmark Staff
[Person]
Back office Bookmark staff

Volunteer Mobile App
[Container: React Native]
Presentes volunteer opportunities and manages reading sessions

Volunteer Web App
[Container: React]
Sign up and application process

School Hub
[Container: React]
Volunteering management

Async Operations
[Container: Azure Functions]
Syncs and moves data between the wider platform and Salesforce

REST API
[Container: Azure Functions]
Syncs and moves data between the wider platform and Salesforce

Salesforce Integration
[Container: Azure Functions]
Syncs and moves data between the wider platform and Salesforce

Reporting Warehouse
[Container: Azure SQL Database]
Combined data for reporting and analytics

Message Broker
[Container: Azure Service Bus]
Queues, topics and subscriptions

Warehouse Upkeep
[Container: Azure Data Factory]
Integrates. shapes and transforms data from sources of truth to an analytic warehouse

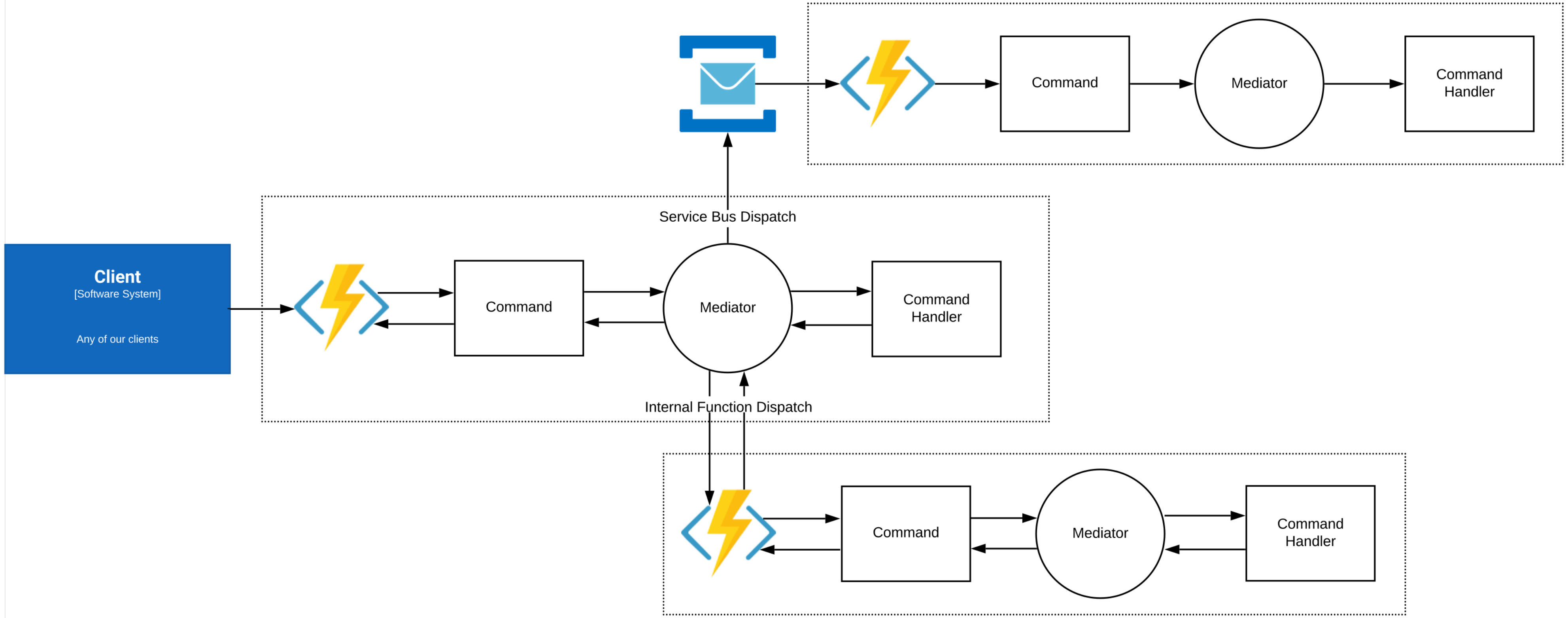
Bookmark Core Platform

Commands and Mediators

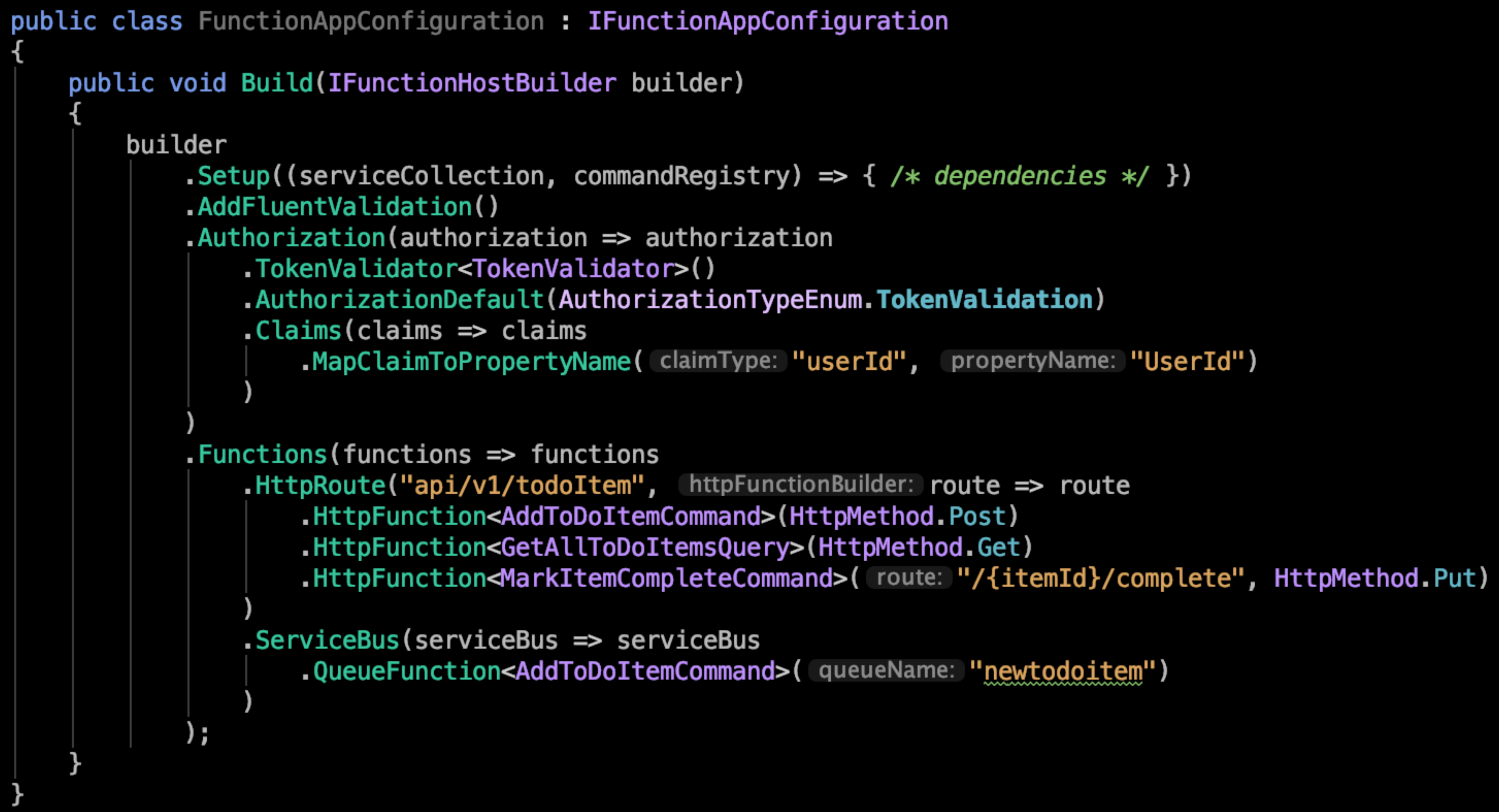
- A command is simple state associated with a C# type
- A command is dispatched for execution to a mediator
- The mediator invokes the configured executer:
  - In process via a command handler
  - Out of process via dispatch for remote execution (other Functions, Service Bus, APIs)
- Cross cutting concerns addressed consistently and once within the mediator

What does this look like?

- This was all supported with the Function Monkey library https://functionmonkey.azurefromthetrenches.com

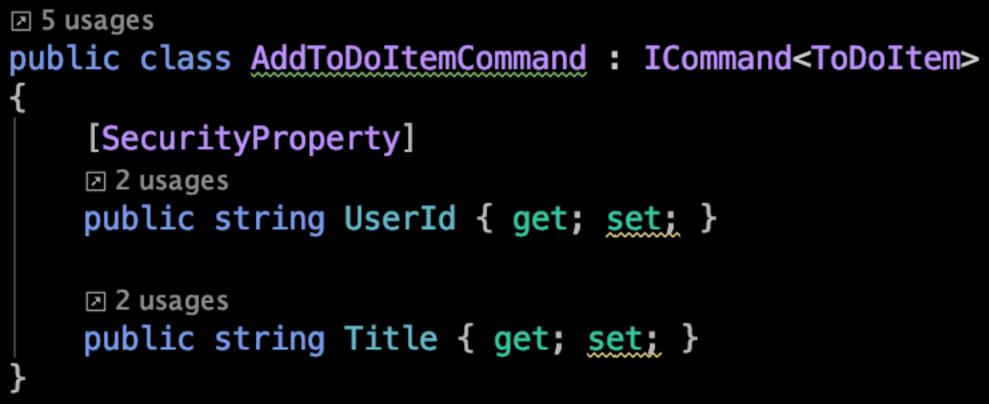


Sample Function Block



```
public class FunctionAppConfiguration : IFunctionAppConfiguration
{
    public void Build(IFunctionHostBuilder builder)
    {
        builder
            .Setup((serviceCollection, commandRegistry) => { /* dependencies */ })
            .AddFluentValidation()
            .Authorization(authorization => authorization
                .TokenValidator<TokenValidator>()
                .AuthorizationDefault(AuthorizationTypeEnum.TokenValidation)
                .Claims(claims => claims
                    .MapClaimToPropertyName( claimType: "userId",  propertyName: "UserId")
                )
            )
            .Functions(functions => functions
                .HttpRoute("api/v1/todoItem",  httpFunctionBuilder: route => route
                    .HttpFunction<AddToDoItemCommand>(HttpMethod.Post)
                    .HttpFunction<GetAllToDoItemsQuery>(HttpMethod.Get)
                    .HttpFunction<MarkItemCompleteCommand>( route: "/{itemId}/complete", HttpMethod.Put)
                )
                .ServiceBus(serviceBus => serviceBus
                    .QueueFunction<AddToDoItemCommand>( queueName: "newtodoitem")
                )
            );
    }
}
```

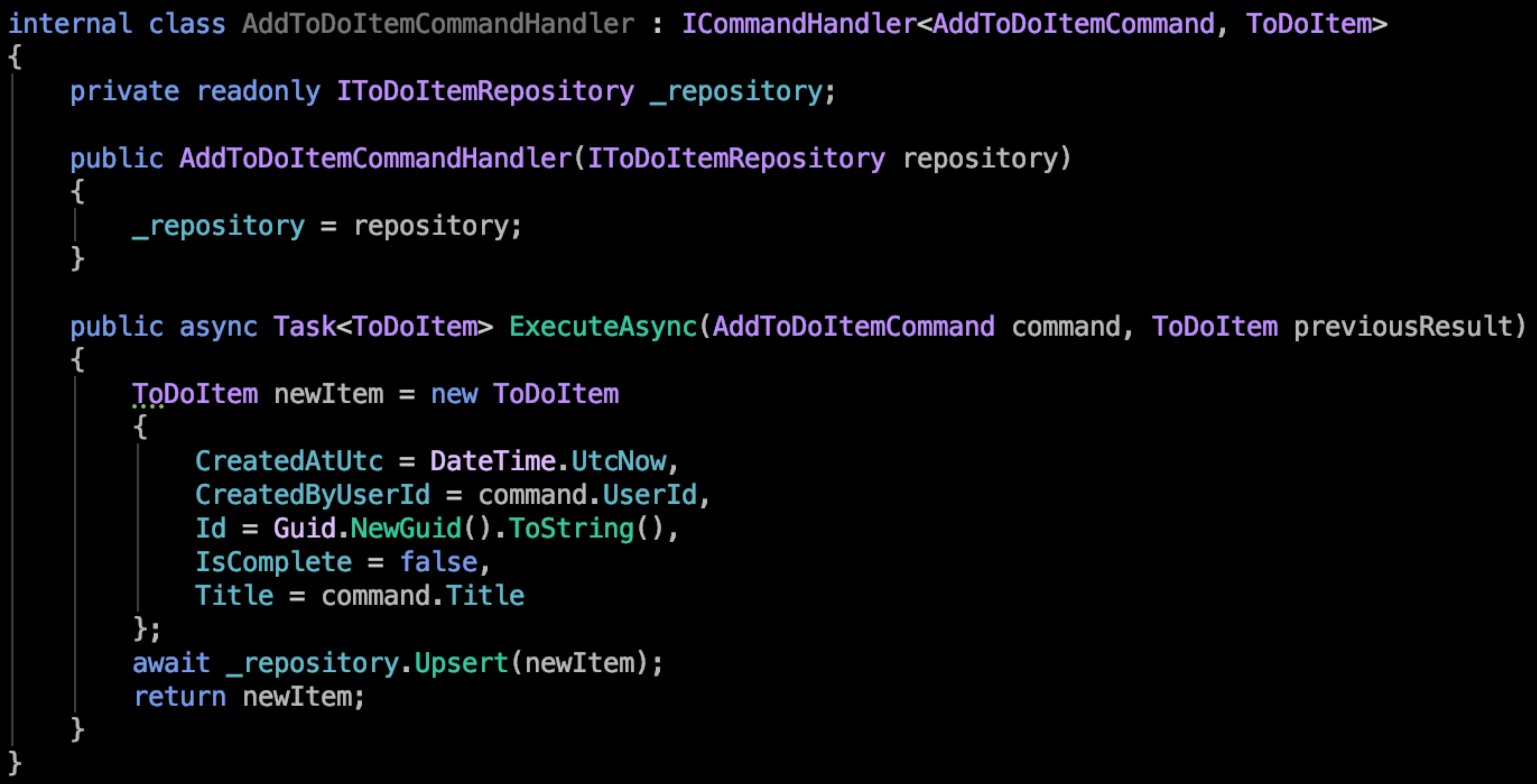
Sample Command



```
5 usages
public class AddToDoItemCommand : ICommand<ToDoItem>
{
    [SecurityProperty]
    2 usages
    public string UserId { get; set; }

    2 usages
    public string Title { get; set; }
}
```

Sample Command Handler



```
internal class AddToDoItemCommandHandler : ICommandHandler<AddToDoItemCommand, ToDoItem>
{
    private readonly IToDoItemRepository _repository;

    public AddToDoItemCommandHandler(IToDoItemRepository repository)
    {
        _repository = repository;
    }

    public async Task<ToDoItem> ExecuteAsync(AddToDoItemCommand command, ToDoItem previousResult)
    {
        ToDoItem newItem = new ToDoItem
        {
            CreatedAtUtc = DateTime.UtcNow,
            CreatedByUserId = command.UserId,
            Id = Guid.NewGuid().ToString(),
            IsComplete = false,
            Title = command.Title
        };
        await _repository.Upsert(newItem);
        return newItem;
    }
}
```

- Our compute code is very lean
  - 95% + addressed business concerns
  - No boilerplate
  - Consistent
  - Its easy to move and repurpose
- Its easy to test – both acceptance and unit
- Its easy to change
- Operationally its been a breeze – nothing to do, everything is automated and was easy to automate
  - We had Azure DevOps Pipelines set up from the very start
  - Everything is a highly managed Azure service

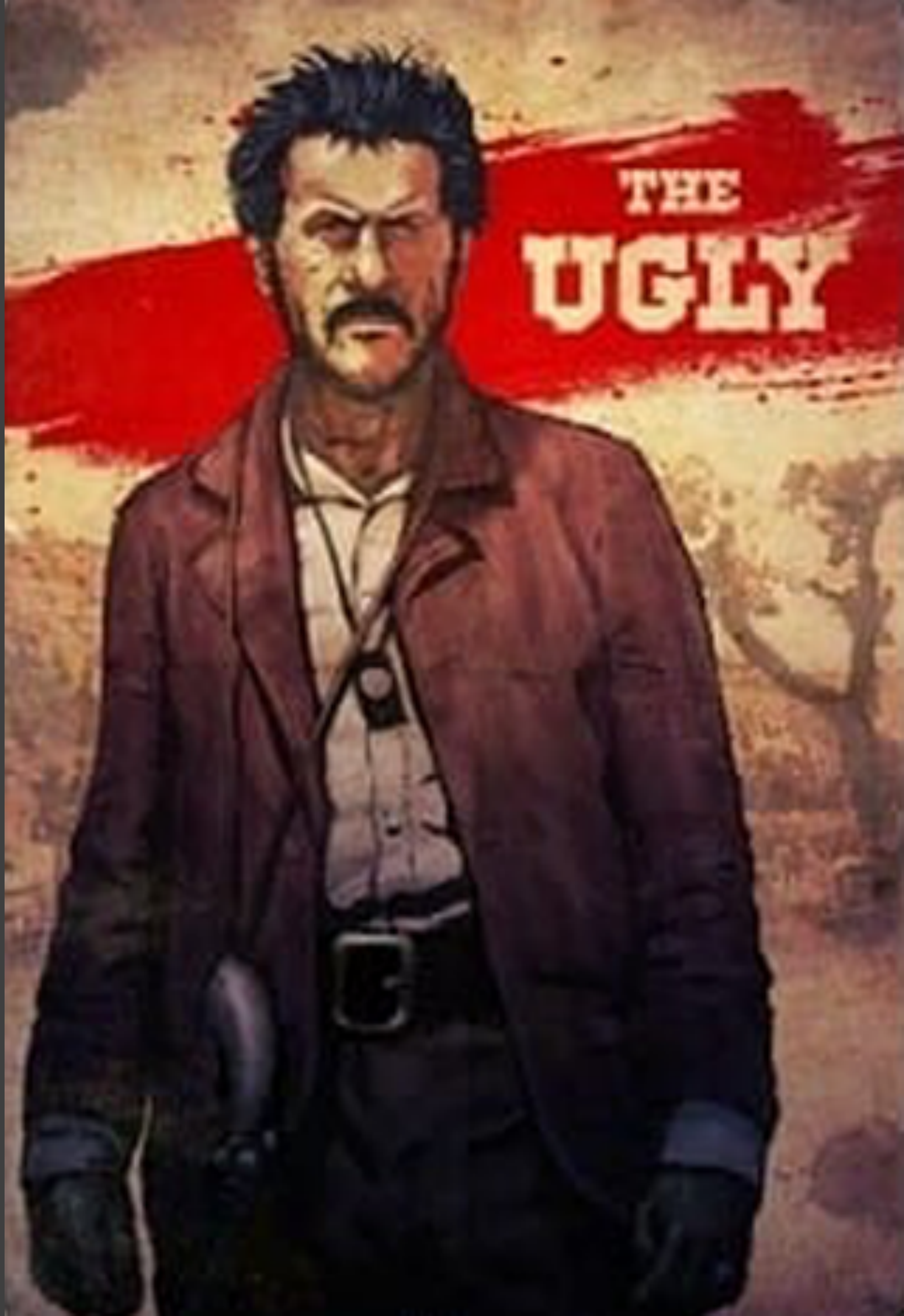
- Operational costs are low – we have three always available environments running (dev, preview, live) and spend little
- Support has been straightforward
  - We really benefited from the consistent addressing of cross cutting concerns
- Its easy to change – we went through two significant redesigns of scheduling in response to feedback and it was fairly straightforward
- The Service Bus acted as a low-UI operation manager

- Other than the function declarations our compute platform is completely decoupled from Functions themselves
- If required we can pick it up and drop it in a container
  - No intention to do so!

- Until “run from package” was released deployments were unreliable
  - Azure App Service locked file issues
- Requirement changes led to Cosmos capabilities going largely unused but we are paying its minimum 400RU footprint

Or rather: To do!

- Make use of API Management
  - Prior to its serverless variant this was disproportionately expensive
  - Would have cost more than the rest of the platform combined
- More data to integrate in the Warehouse from other external systems
- Move the React apps to Netlify
- Find a better meme!



Links

Function Monkey -
https://functionmonkey.azurefromthetrenches.com

Function Monkey dev.to Tutorial -
https://dev.to/jamesrandall/elegant-azure-functions-development-in-c-with-function-monkey-1ea7

Slides -
https://www.azurefromthetrenches.com/serverlesslondon/

Mediator -
https://commanding.azurefromthetrenches.com

Bookmark Reading -
https://www.bookmarkreading.org