

## Introduction

In this scenario, you are responsible for creating the test automation frameworks for the web applications to empower the whole engineering group to deliver faster with confidence in product quality. You would need to cover things from technology choices to architectural decisions, to implementation strategies, to testing automation education and coaching.

### Our advice

Explain your thinking, even if you don't know all the answers. Saying something like "I think XX library would work well here, but I'd need to research more around XX areas" is fine.

Clearly state your assumptions—both the ones we should know while reading your submission and those you made while developing the solution.

Ensure you explain the reason behind your choice of technologies.

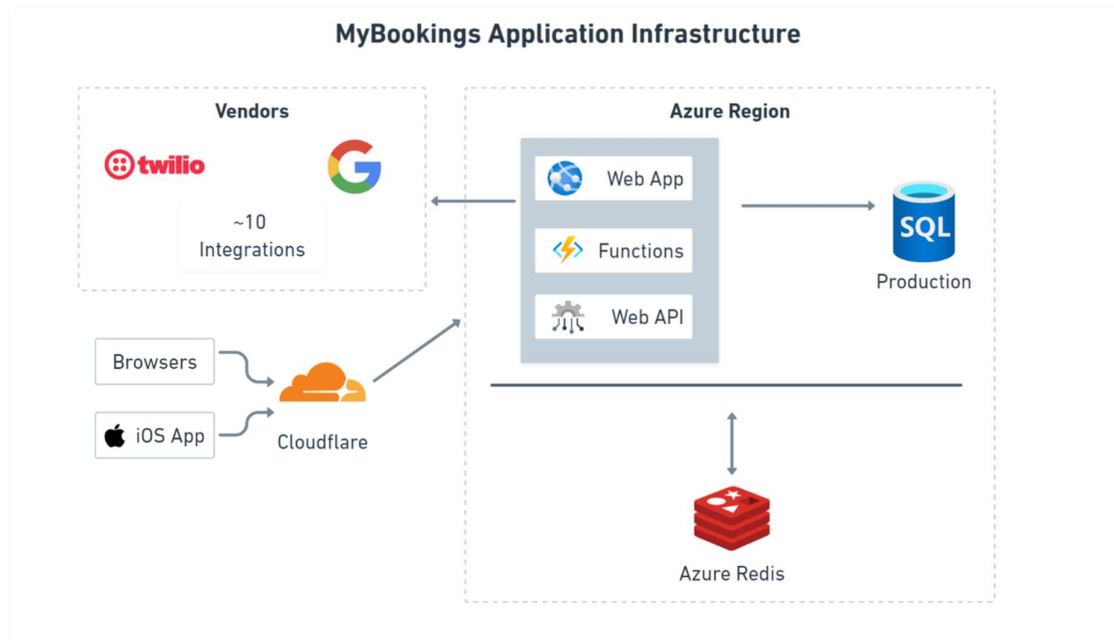
Consider the users of the test automation framework.

Don't spend more than 2 hours.

## Overview

MyBookings web application provides basic functionalities for organizing calendar activities. It can be used for any authorized user to manage appointments in the book.

The main software tech stack includes .NET 6+ web application as backend, React as front end, SQL Server as production database, Redis as caching layer. The following solution architecture overview shows the 3<sup>rd</sup> party integrations, network routing infrastructure and its usage of other Azure services.



## Exercise One

Please refer to the [Wireframes attached](#) to this document.

The main usage of the application is:

- Log the user in
- Add a new appointment
- Fill in details for the new appointment
- Save the appointment
- The new appointment is visible and has the entered details
- Email notification is sent out to the user

Based on your experience with tools/libraries, create the basic UI end-to-end test automation framework for the main web application that is shown above, including test cases for the functionalities shown above. We encourage you to consider various potential paths for test cases, other than only the main one.


*The exercise doesn't require all the code to be fully running. The folder structure of the framework and key modules are essential and pseudo code is ok for non-critical areas as long as you make it clear in the source code.*

## Exercise Two

Yay~ the test automation framework you just developed has been widely used for web applications and more and more tests are built by engineers in the organization with it. However, test configurations have got wildly out of control with scattered configurations, too many test cases and it is super hard to understand the automation test coverage. There are flaky tests popping up often in CI pipelines which makes the engineering group lose trust in the value of they bring.

What would you consider addressing those problems? Then how would you go about addressing them? Please include the why and how, with examples from your experience if you can.

# Wireframes for exercise One

 My Bookings App

Log in



Email

Password

Log in

Sign up

[Forgot your password](#)

 My Bookings App	<div>New appointment</div>					Username 
Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday

Wireframes for exercise One (continued)

My Bookings App

New appointment

Username

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday

New appointment

X

Appointment name

Date

Time

Save

Cancel

My Bookings App

New appointment

Username

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
		<div>11:30am - 12:30pm Example appointment</div>				

Appointment saved!