High Level Design Document:
Dbate

Prepared By: Red Team

Group Members:

Christian Flores-Rogel 013924454 (Team Leader)
Deivis Leung Liang 014110497
Luis Meza 014325959
Keanna Mae Vitug 014699514
John Cayton 014108690

Professor: Vatanak Vong

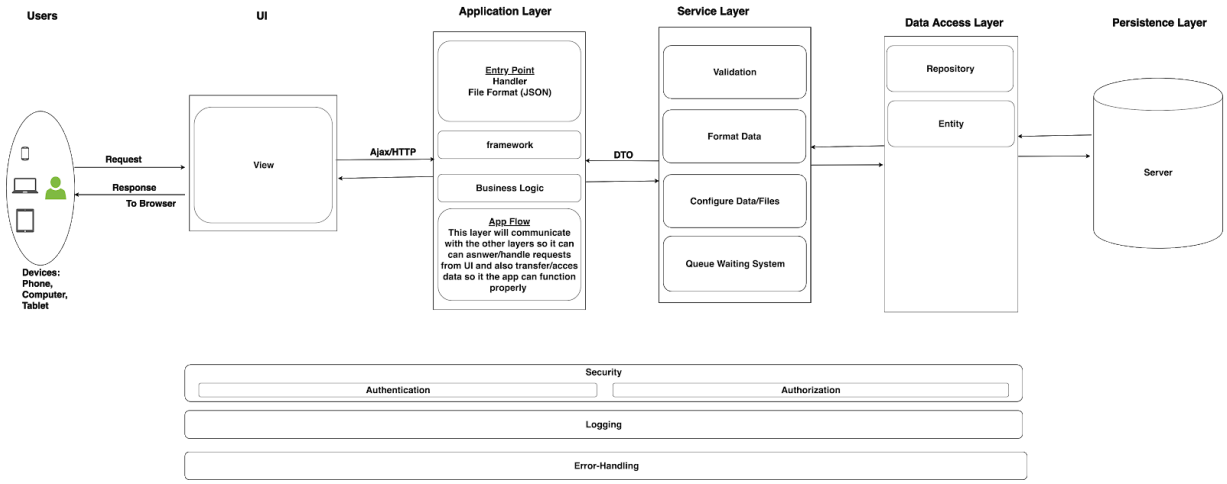
Date: December 11, 2018

Table of Contents

Introduction

The following design document provides a high level architecture overview of the web application Dbate. The purpose of this document is to provide insight on how Dbate will interact with the user. Dbate is a web app that allows its users to debate with a random topic in a moderated environment. It will be designed as a Single Page Application (SPA) which allows for dynamic interaction between user and system that rewrites the current page as opposed to loading new pages from a server. The Web API we will be utilizing is ASP.NET Web API which is a pervasive framework specifically geared towards HTTP based services that can be accessed from a variety of applications on different platforms. ASP.NET Web API is similar to ASP.NET MVC Web application in many ways. However, it differs in that rather than sending data through an HTML view, it is received as a response. In addition, ASP.NET Web API is a web service, but it only supports HTTP protocol. The data format that we will be using is JSON since it would complement the fact that this web app front end will be based in Javascript and it is faster in performance than other file formats such as HTML and XML due it's being more lightweight.

Layered Diagram

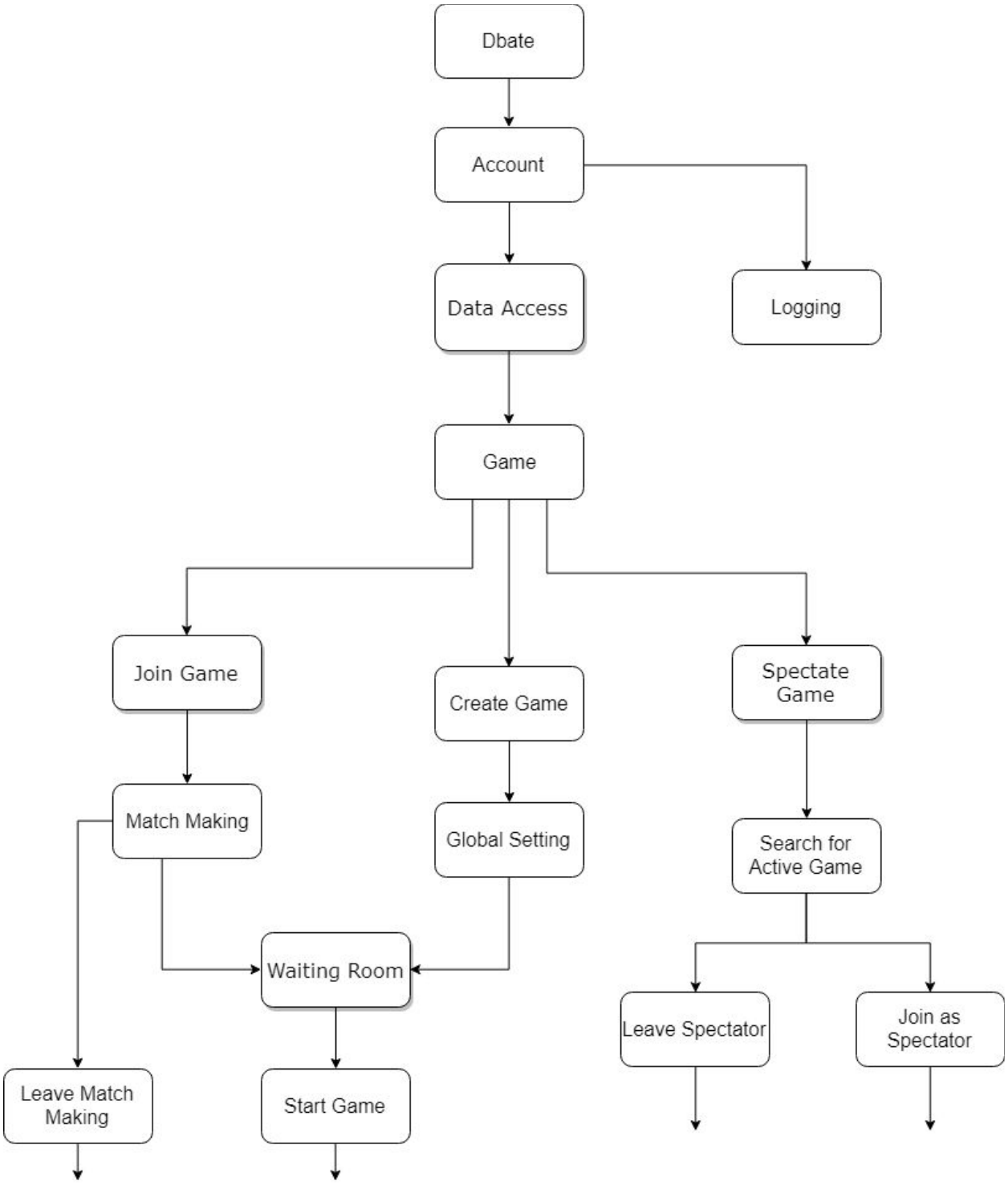


Description:

This layered diagram projects the architecture of the system we plan to create. By displaying the structure of User Interface, Business, Data Access, and Data Store/Persistence layers, the developer is able to understand how our system will function. By having this architecture diagram designed in layers, it will help the developers during the implementation stage of the web application. The layered diagram first starts by in the *Users* section which shows how the users devices will send/receive responses and requests so the users can gain access to the front end of the web app (*UI layer*). The *UI* layer has the view which shows the design of the app that the client/user views. In the diagram, the *UI* sends/receives AJAX/HTTP responses from the application layer. The application layer handles the request and also has the interface, business logic and framework which run's the app. When the UI receives the responses the UI changes according to what the application layer has. The diagram also show how the Application and service layers receives and sends Data Transfer Objects(DTO) between each other. DTO means that it moves data between each layer so each layer can receive them and manipulate however it is supposed in their respective layers. The service layer contains stand alone operations that web app needs that are reusable such as validation and authorization. The next layer in the diagram is the *Data Access Layer* which pretty much provides an easier way on how the data is accessed. Between these layers, data access objects are used to access and retrieve information from the database without exposing data stored within it.

The last layer is the *Persistence Layer* which is just the server. The diagram also contains security, error handling, and logging which are common operations that all layers may have.

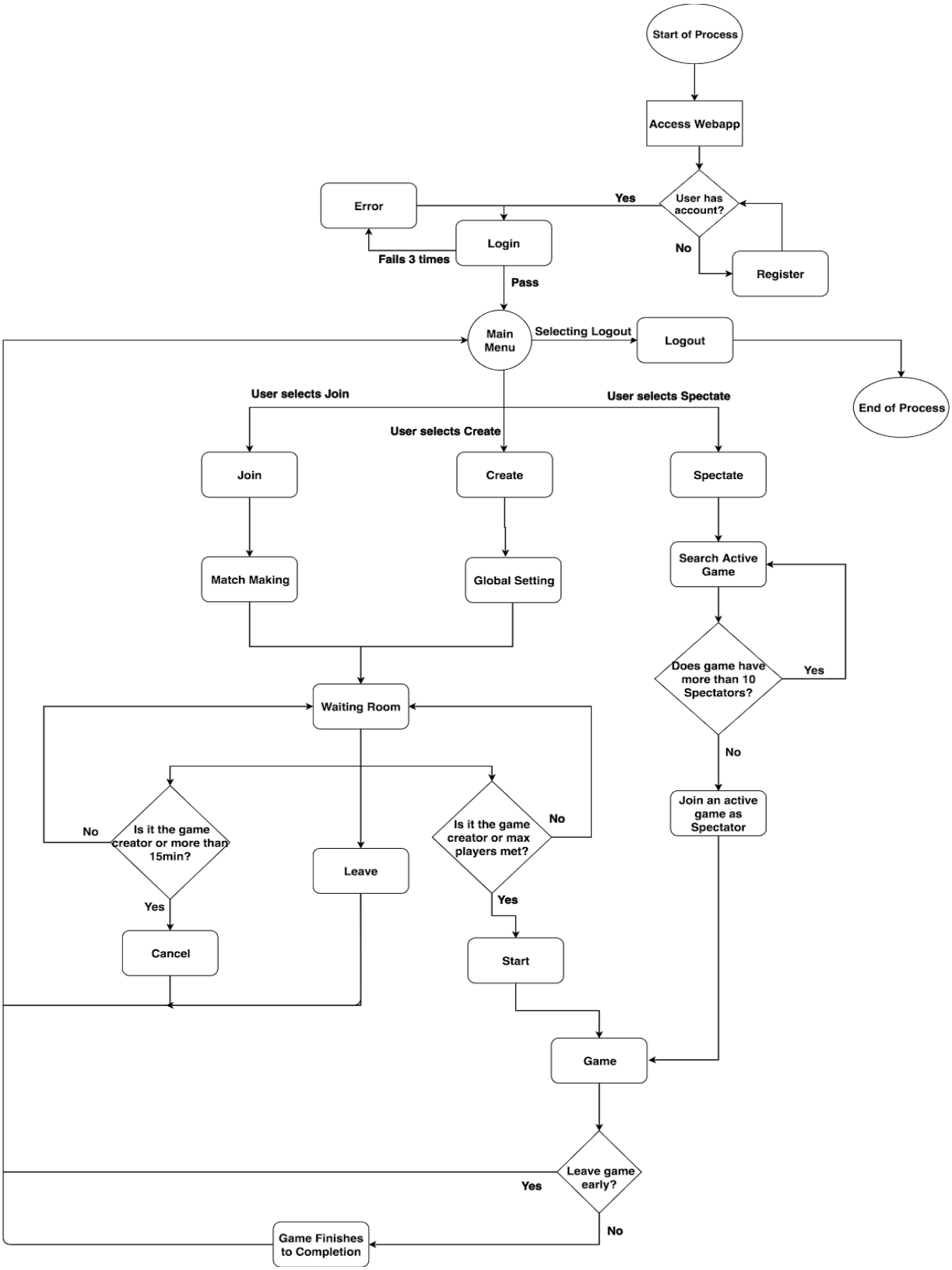
Dependency Tree



Description:

The dependency tree allows the client and users to understand the dependencies of the different functionalities in our web application. It provides a high level view of what features will depend on those above it in order to properly function. For our example, in our web app, every other major feature depends on a user having an account. Without an account, users will not be able to do any of these other actions. In addition, it shows a high level view of what features might be affected by a modification to another feature and as a result, have to be modified as well.

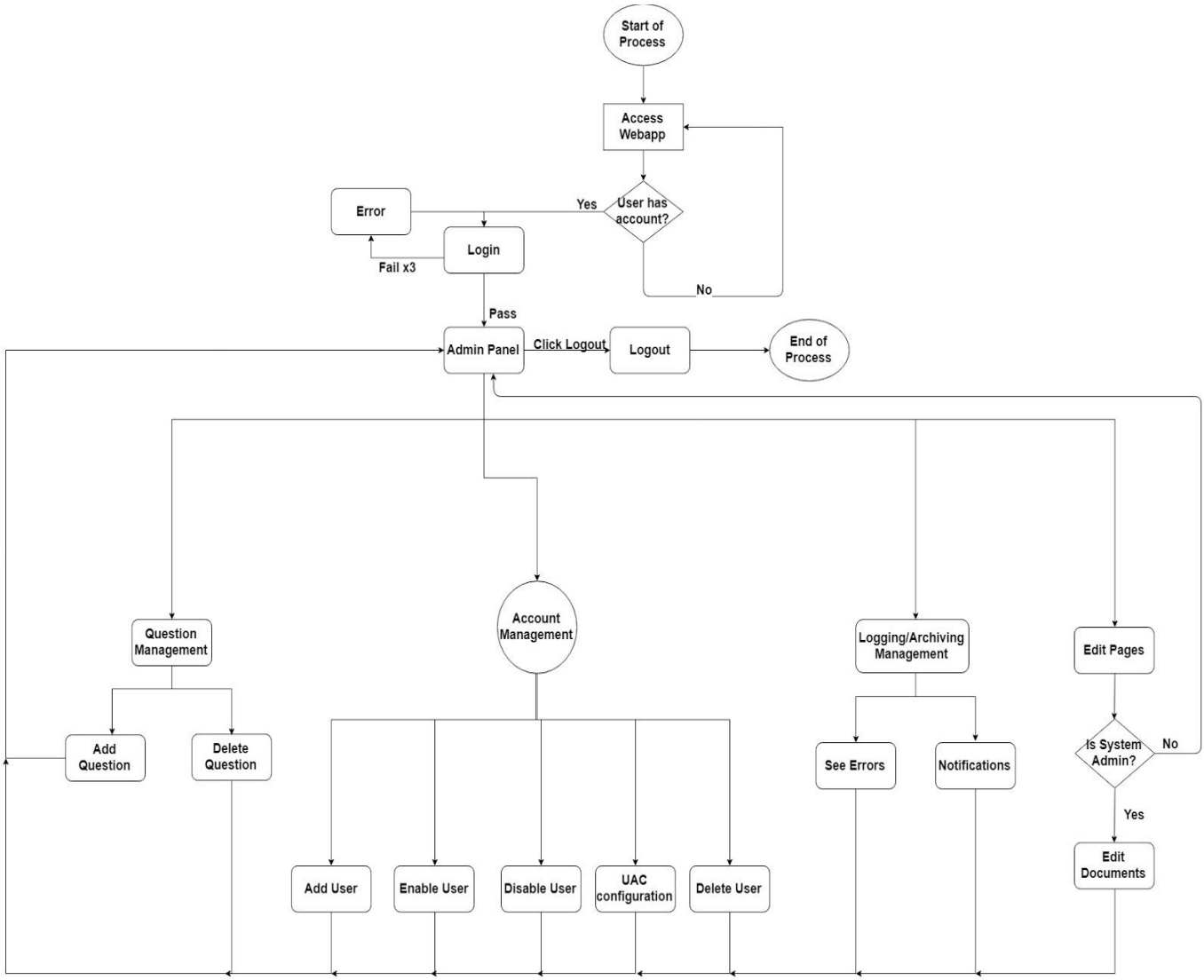
Client Workflow Diagram



Description:

The client workflow diagram shows an overview of the flow of how an ordinary user will interact with the web application. It shows the steps that a regular user will go through as they progress through the application until they finish using the app. The process begins when the client first accesses the web application. Arrows will indicate the general flow or direction of the process in order to show the progress that a user has made in our system. Diamonds and circles indicate that there are multiple possibilities at a certain point. Normally, these are related to the user's choice on what occurs next. The most important step in this process is whether the user has an account, which is necessary to access all of our application specific features. Finally, after the user has finished using the system, the process will end once the user logs out and exits the application.

Administrator Workflow Diagram



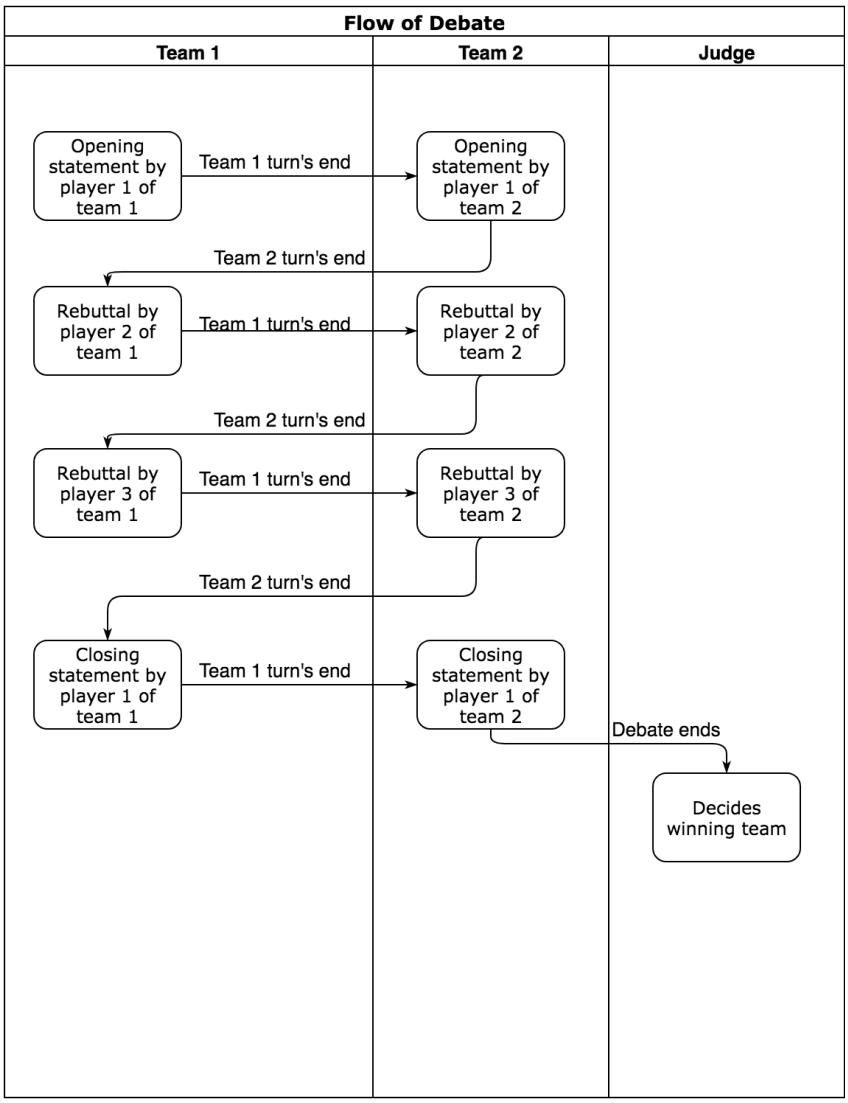
Description:

The administrator workflow diagram shows an overview of the administrator specific actions that can be performed by administrators and system administrators. This diagram shows the process of how administrators can manage questions, manage the accounts that the web app has, manage the logging/archiving data while also being able to edit specific pages. The diagram also shows that editing pages will only be allowed if the person logged in is a System Administrator.

This workflow diagram shows the flow on how the admin will interact with the web-app. The starting process begins when the admin access the web-app. The arrows indicated the

direction on how the interaction will go. The diamond squares will indicate an if/else statement of the diagram. The process will end once the client logs out of the web-app.

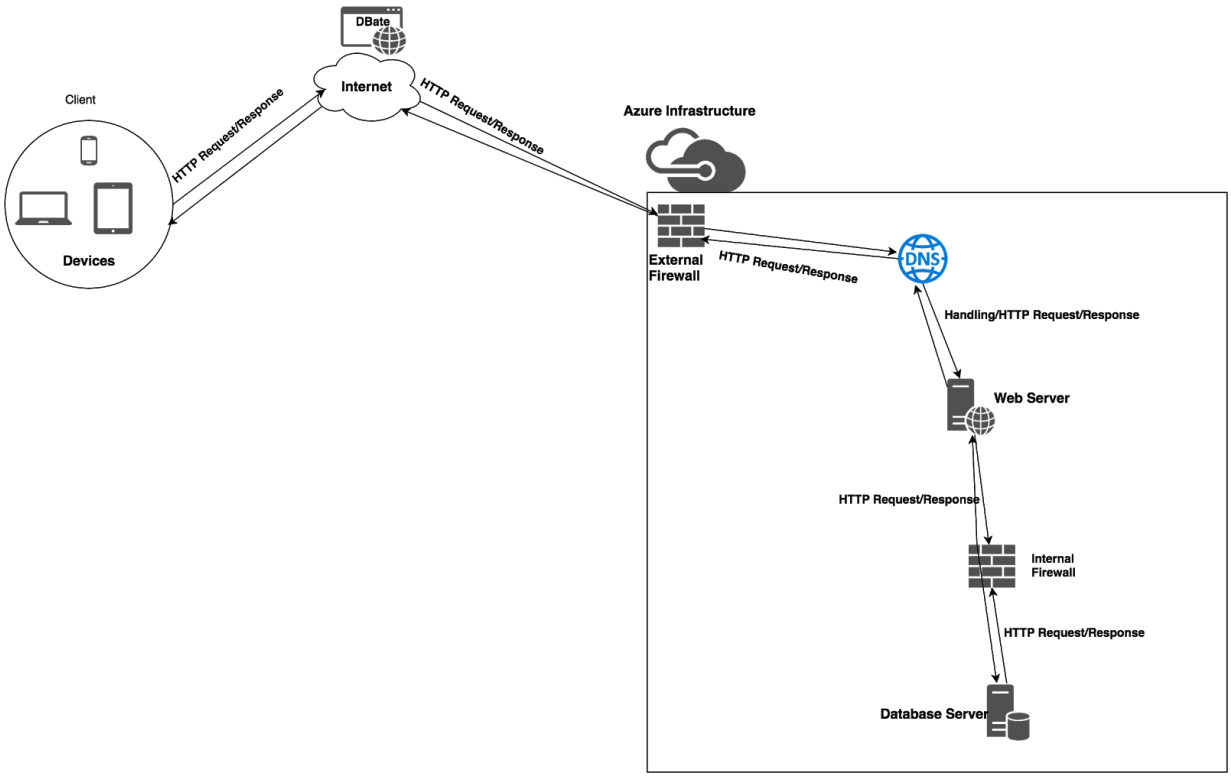
Flow of Debate Swimlane Diagram



Description:

This swim lane diagram demonstrates the flow of our web app's debate room feature. It shows how the debate will take place in an active chat room and how the rounds will flow. For the first round, player 1 in team will give his opening statement. Once player 1 turn's has ended, player 1 from team 2 gives their opening statement. This flow of turn will continue for the 2 Rebuttal phases and the closing statement. Finally, once the player 1 of team 2 gives his or her closing statement, the flow will go to the judge/s where he or she decides the winner of the debate.

Network Diagram



Description:

The network diagram shows how the clients devices connect to a network so it can have access to the web application. The diagrams shows how when devices enters the address to reach the web app, its sending HTTP requests in order to access it. So the app can be access through the internet, it has to be deployed through a cloud service. The diagram states how app is stored on Azure a cloud service that is used to deploy and maintain apps. Before it goes further, an external firewall checks whether the requests are trustworthy and not potentially harmful. After passing through the external firewall, it passes through the dns so it can convert the domain address to a readable ip address and vice-versa so it can access the data for the web app. After the DNS, it passes through the web server which stores and processes network requests while the database server in the infrastructure is in charge handling data queries so the other servers can

retrieve data that the other server need. All these servers interconnect with each other so the app can function and be available through the internet.