**Self-reflection**

The program allows a user to sign up for an account using their username, name, and password. Using the random number generator service to create an ID for the user, these details are then saved into a database. The user will then be instructed to log in. The username and password are sent to the orchestrator service, if the details are verified a success message is sent, if it is not an error message is sent back to the client. Once the user has logged in successfully, they can do several actions. Create a Trip, the user can enter the dates they want and the location to create a trip which will display for all users. Query a trip, the user can search for trips by entering the locations from and to. Check the weather, before proposing a trip the user can check for the weather on each day of the trip. Express interest, the user can express interest in specific trips from other users, this expression will then be sent to the user who proposed the trip. Finally, the user can check interest on the trips they proposed, this will show the interested user’s name and details about the trip. The user would be able to then accept the interest, this functionality is incomplete.

The program implements REST services which the client uses to complete actions. The program consumes two external services, one being the random number generator service and the other being the weather service. All communication is done in JSON. The HTTP methods deployed are POST and GET. POST is used when data parameters are needed and GET is used when no data parameter is needed. A GUI was created for the project and all data was stored in a derby database.

A diagram of a user flow

Description automatically generated

**System Analysis**

The current solution is a web application using a service-centric architecture. The solution relies on 2 services and a relational database. The services communicate through RESTful web services, and the orchestrator application is currently deployed on Tomcat locally. Currently, the solution’s orchestrator is deployed locally, this poses several challenges. The locally deployed infrastructure limits the ability to scale up or down due to the demand for the website. This can lead to the solution having slow response times or in the worst-case crashing. Secondly, the current solution means there will be a single point of failure. If for some reason the hardware for the solution fails, then the whole website will crash and be unusable until the hardware is fixed.

Migration to the cloud will mitigate some of these challenges. An approach would be to host the solution’s orchestrator service on the cloud, through a Microsoft Azure virtual machine. This hybrid cloud solution integrates local infrastructure with cloud services. This approach offers advantages to do with flexibility, robustness, and cost-efficiency. Allowing the services to be deployed on the cloud while the client is local will allow it to access the services on a more fault-tolerant platform. For cloud migration to be useful the cloud service must have features that enhance security when sending data to the local host and support database scalability. The current solution sends sensitive data from the client to the orchestrator, this data must be secure. Moreover, since the orchestrator handles the data and places this data into a database, the database being on the cloud instead of locally will allow for quicker backend operations due to no latency. Azure addresses these issues by supporting databases using Azure database, and by having Azure application gateway. Azure application gateway has features such as a firewall, data encryption, and decryption. This ensures that the data sent is secure. Furthermore, Azure supports the database being hosted on the cloud. This allows for the database to be more scalable. These features ensure the solution will be better hosted on the cloud than locally.

However, some challenges may arise in terms of latency. Having the solutions orchestrator service on the cloud while the client is locally deployed may impact communication speed. This may be because of the physical distance between the server and the locally deployed infrastructure. A solution to this will be to ensure the Azure region selected is close to the physical infrastructure. This will mean there will be less physical distance resulting in less latency. Furthermore, Azure ExpressRoute could be used. This extends a physical infrastructure network to the cloud through a private connection. This results in less latency and a faster connection.

Quality of services, or QoS, plays a vital role in establishing application reliability and responsiveness. Key QoS measurements, including availability, fault tolerance, and response time are essential for maintaining optimal system performance. Attaining high QoS levels requires optimal load balancing, user resource allocation, and robust fault management. High QoS levels are essential for user satisfaction. One area of QoS that can be improved in the hybrid model is scalability. Although some scalability issue was resolved by migrating the orchestrator to the cloud not all issues were resolved. Due to the solution being in a hybrid environment, there are challenges due to physical resource capacities and potential latency. Implementing an elastic load balancer will address the limited resource capacities. An elastic load balancer dynamically distributes incoming requests across available resources. This ensures optimal resource utilisation and prevents overload on specific resources. By leveraging Azure’s auto-scaling features, the solution will be able to adjust resource allocation based on demand automatically. Furthermore, as mentioned, Azure ExpressRoute can be used to address the latency issue. By connection being private and not public, faster data transfer can occur between the cloud and physical infrastructure.

**Analysis of Big Data**

Here the solution will be evaluated based on big data. The 8V’s, Volume, Velocity, Value, Veracity, Variability, Visualization, Validity and Variety are useful tools to dissect the nature of big data. In this analysis, I will scrutinise 4 of the 8V’s and suggest a cloud deployment model.

Volume

Volume highlights the vast amount of information continually produced by applications. The solution for the travel buddy application includes many user profile data, trip data, weather data and interest data as well as other data. The vast amount of information requires proficient storage strategies as regular storage solutions will not be able to handle such large amounts of data. As a result, adopting scalable solutions such as NoSQL, in addition to having distributed storage solutions, such as the Cosmos database becomes essential.

Velocity

Velocity highlights the rate at which data is collected and processed. The travel buddy application may have a constant large stream of data collected and processed. This could include weather data, user data, interest data etc. Meeting the demands of such immediate data flow requires robust processing capabilities and well-optimised network infrastructure. Having microservices based on the cloud can prove to be an effective solution for addressing these challenges. Each functionality of the app, such as trip submissions, can be encapsulated as independent microservices. These microservices operate autonomously, enabling parallel processing of tasks.

Value

Value alludes to the business value of the data collected. In the context of the travel buddy application emphasises the significance of translating large amounts of data into tangible benefits for the end user. The aim will be to extract value by analysing user preferences, feedback, interests, and past data. Understanding the value each user seeks in their travel experience allows the application to tailor recommendations. Using cloud-based solutions such as Azure HDInsights can be instrumental in extracting useful data from large volumes of data.

Veracity

Veracity conveys the reliability and accuracy of data; this is crucial for maintaining the end user’s trust and making informed decisions in the application. In the Travel Buddy application, data quality assurance measures are implemented to address this. By ensuring the veracity of data, such as trip proposals and user information, data will be more accurate for the user. Using cloud tools like Azure data quality service veracity can be kept high.

Infrastructure and software implications & solutions

Big data applications have both software and infrastructure implications. Firstly, scalability issues. Big data applications require scalable infrastructure to handle the varying workloads. For example, the Travel Buddy application in the months leading to summer will most likely have a heavier workload than during the middle of winter. If the application is not scalable it may cause it to slow down and potentially crash for the end user. A cloud-based solution allows the Travel Buddy application to scale up or down based on the demand. This will always mean a satisfying user experience and is more cost-effective than buying more expensive infrastructure.

Secondly, big data applications have storage capacity issues. Purchasing physical storage for big data is very expensive due to the vast amount of data. The volume of data generated by the Travel Buddy application could be extensive, from user data, to travel data, a lot of data will be generated and potentially stored. This can be very costly, and the storage will eventually become full. Cloud-based storage services will be beneficial for providing a cost-effective, scalable storage solution.

Lastly, big data applications offer tremendous opportunities for insights and innovation however they also must be secure. These applications deal with massive volumes of diverse and often sensitive data, so ensuring robust data security measures is crucial to protect user privacy and comply with data regulations.

Cloud deployment

Cloud deployment can be a crucial strategy to solve these big data challenges. Considering the varying workload of the travel buddy application a hybrid cloud deployment model would be most suitable. This model will be the most flexible and cost-effective model for the application. The deployment model can deal with scalability using public cloud resources to dynamically scale up. Furthermore, by using distributed databases hosted on the cloud, the hybrid model will be able to adapt to the volume of data generated by the Travel Buddy application during peak holiday times. Finally, sensitive user data can be kept within the private physical infrastructure, this provides enhanced security and complies with data security regulations.

Conclusion

To conclude the 8V’s unravel the challenges of applications that generate big data. The dimensions of Volume, Velocity, Value and Veracity shed light on the diverse obstacles encountered in features like storage, processing, and extracting useful information. Tackling these challenges needs advanced solutions such as distributed databases, microservices etc. Cloud computing is a catalyst for innovation in the big data space and can provide many solutions to its challenges.