Vercel: A Modern Hosting Solution

Aaron Doyle
Department of Computing
SETU Carlow
Carlow, Ireland
C00272515@setu.ie

I. Introduction

This report aims to explore and discuss Vercel and how it is revolutionising web application hosting. Vercel is a cloud platform that aims to facilitate the hosting and instantaneous deployment of modern front end web applications. In this modern age trying to find a hosting service for web applications, especially ones that offer free hosting is an extremely tough task.

Unlike platforms like AWS or Azure who offer a hosting service for a full stack web application, Vercel has a different target audience. Vercel is purely focused on the front-end developers and how easy and seamless they can make the process of deploying their front-end web applications to the web. This is where they differ from the other cloud deployment services available. Unlike AWS and Azure where the end user is required to do a lot of server configuration which can be very confusing for the user and leads to big problems later down the line if it isn't configured correctly, Vercel handles all of this for the user as it is serverless for the user. This is a big benefit for front-end developers who might not have the knowledge of setting up the server correctly for their needs.

Vercel have some very big companies included as their use cases such as the likes of Chick-Fil-A, Stripe, Sonos, Adobe and Loom just to name a few.

I have used Vercel myself for some personal projects as for front end web applications the process of deploying it is seamless. Traditional deployment can be very time consuming as you need to specify every aspect of your application and manually deploy every time a change is made to your application. Fast and automated deployment is so important in the modern age as the tech world is progressing and expanding at such a rapid rate and the expectations and deadlines that are put on developers to create and ship new features and products is increasing exponentially. My use of Vercel is what made me want to choose it for this report as I would like to further my knowledge on the platform so that I can find out where it excels and where it falls.

Vercel excels as when linked to your GitHub account it can automatically deploy the updated version of your application when a push to main is made which takes the need for manual deployment away from the user. This can be very useful as manual deployment can lead to human error in the deployment process compared to when it is automated. In this report I will discuss the benefits and detriments of Vercel and how it is a leader in modern age front end web application hosting and deployment. I will also be delving into and discussing the architecture of the platform and how it works at a deeper level.

II. Literature Review

A. Serverless vs Traditional Hosting & Deployment [1]

This section of the Literature Review aims to discover and highlight the key differences in Serverless vs Traditional hosting and how it can benefit Vercel and their target audience. The first piece of research that I wanted to carry out so I could build a foundation of knowledge on this topic was the strengths and weaknesses of the different types of hosting and deployment. Vercel is very much a serverless hosting platform, but I wanted to see what the benefits of each were. There are great benefits for both, starting with traditional hosting, this Is a better method if you need full control over the entire environment. You might need this for a few reasons such as the software you are using might need specific requirements on the server or you need to have very monitored control over the hardware resources if it's a high intensity application. The issue with traditional hosting methods is that it is all down to the user to configure and manage throughout the lifecycle of the application. For certain applications this won't be a big issue but for smaller applications this can be a lot of needless overhead. This brings us on to the serverless hosting and deployment services such as Vercel.

Vercel have made the right choice here in making this a serverless platform as for their target audience and projects you don't want a huge overhead of managing the specifics of the server. A big benefit of having serverless hosting is that it auto scales. What is meant by this is that as the application starts to gain more traction Vercel will automatically up the number of resources that the app has available so that you won't have an issue with it running out leading to downtime on the application. Having a serverless system also leads to much faster deployment as it can be set up to use Continuous deployment and integration. This means that the user isn't having to manually having to deploy their changes each time it is just automatically done by a trigger which is usually something like a commit or push to a certain branch in a repository for example.

Kubernetes [2] is a container orchestration platform that offers autoscaling within its features. I have used Kubernetes and it is great for deployment it is still more of a infrastructure based method as you do have to still manually configure and setup the auto-scaling feature to get it started. This is great but I think that Vercel's method of auto-scaling is way better suited for their application. This is because the way Kubernetes implements this feature can still be quite confusing and overwhelming to setup initially whereas in Vercel is this all taken care of by Vercel for the user. There isn't any user configuration required to get it working and for a front-end developer whose focus is just the application that they are working on this is a huge benefit to using the application.

Understanding both methods of deploying a project really helped me in understanding why Vercel chose to go with a serverless model. It fits perfectly with their target audience and projects as they aim their services to front-end applications and developers and most of these projects aren't going to have huge amounts of logic and its ease of launch is perfect for a front-end developer who mightn't be used to server management and the overhead that comes with it. Having the automatic scaling is also a huge benefit to Vercel as it allows them to be prepared for any size of application that comes their way whether it's a personal portfolio site or dealing with projects for Adobe with millions of users. In conclusion of this section, I think that Vercel have excelled in their method of deployment in choosing serverless as their target audience will want to deal with as little of the server side of things as possible, so Vercel's rapid deployment and auto-scaling is perfect for this.

B. Continuous Integration and Continuous Deployment

This section aims to discuss and research how Continuous Integration and Continuous Deployment work and how Vercel incorporates them in its service.

Continuous Integration [3] is the process of automating the building and testing of code. This is an extremely efficient method of development when setup as when a developer pushes their code it will automatically run the regression tests to ensure that the new code hasn't broken any existing features or functionality within the application. The automation of this process takes away the risk of human error in the process. It also means that everyone is conforming to the same process as it is forced to be the same for everyone.

Vercel uses Continuous Integration within the process in a few ways. It can connect with GitHub, GitLab and Bitbucket so that it has the ability to see every push, pull and commit that is made to the project repository. This allows it to run the newest version of the code every time a change is pushed in the repository. It also has the ability to run regression tests at each push if it is configured. The user can specify in the Vercel build commands that you want it to run the regression test so that the user can catch bugs earlier on rather than them going unnoticed in production.

Continuous Deployment [4] is the process of automatically deploying updated code to a live environment. This method of deployment speeds up the process of pushing code and features to production massively as it takes the steps of the end user having to go and manually configure and deploy the code by themselves.

Vercel does this extremely well, as discussed when talking about Continuous Integration, Vercel is able to connect to Git services which means that when a developer makes a change and then pushes it to the specific branch that Vercel is running off, it can then automatically deploy the desired changes to the live environment for the end user. This is great for Vercel as for their target audience of front-end developers they may not be equipped to handle the manual deployment process, and this allows them to simply push the code and then check the live environment to ensure that their changes were pushed as expected and not have to manually change the server which could lead to major issues.

For the issues that I have listed above I think integrating Continuous Integration and Continuous Deployment is a key part of Vercel's functionality and success with their service

C. Edge Computing

Edge computing [5] is the method of storing data and running code closer to the end user using different servers around the world dependent on where the end user is.

This works when a company has a lot of server locations which can be referred to as Edge servers. Having this system where the users can run their projects from the nearest edge server leads to big benefits for the end user such as the user getting much lower latency on their projects. This is a massive benefit for the user as if they were using a more traditional method where everything is hosted from the origin server it could take a serious toll on load times depending on where the user is in the world compared to the origin user.

Edge computing also allows the users projects to become more scalable which is another big benefit of using it as when a user's project becomes bigger the workload of handling the project on the server side can be spread out rather than having all of that usage on the origin server. Edge computing also makes it easier for the applications to be able to show location-based content as this is a lot tougher when you are only running from the origin server which could be halfway across the world. For example, users in Brazil could be shown different content than users in Ireland and this can all be done without having to go back and forth with the origin server as it is all local to the user.

Vercel uses Edge Computing [6] or they like to call it their 'Edge Network'. This method of server hosting for Vercel is highly beneficial for the way that they structure their service. Vercel's network includes over one hundred Points of Presence (POP), these are essentially virtual servers which they have set up all around the world to handle incoming requests from the users in their vicinity, this allows them to reduce latency massively as they can cache content here, so they aren't having to ping for content across the world constantly.

Vercel also utilises that they call Vercel functions, Vercel functions are lightweight serverless functions that run on their edge network. These carry out small tasks so that the main origin server isn't being overloaded with thousands of requests if it doesn't need to be. This includes checks like seeing if a user is authenticated, redirecting users based on their location, showing different versions of a page, or showing different ads based off the user's geolocation.

This method of handling the location of hosting and deployment perfectly ties in with Vercel's audience as this leads to a much faster and seamless experience for the end user. This also allows Vercel to scale a project much faster and easier which is perfect for their target users which are front end developers as they don't want to have to deal with the backend hassle of manually scaling their server usage amounts so they can let Vercel do it for them.

III. System Architecture

This section aims to discuss and outline the system architecture for Vercel [7] and how it works. Vercel is a front-end focused serverless deployment and hosting platform. As discussed in the sections above Vercel has tailored their whole system around the needs of their users which are primarily front-end developers. In doing this they have created a seamless process of Continuous Integration and Continuous Deployment where their users can focus on creating their UI based applications and Vercel can handle the server side of things.

Vercel uses Git integration within its service so that it can seamlessly deploy a web application without the need of it being manually triggered by the end user. Every time a push is made to the specified branch that Vercel is watching, it builds the updated code and makes a deployment that can be then seen by the developer without them having to do any extra steps.

Vercel's build system is extremely intuitive as it can detect over 35 languages and frameworks instantly and then automatically apply the correct build commands so that the build and deployment process can instantly start. This is once again a great method of carrying this out for the user of Vercel as it means they don't have to manually deal with setting up their build commands if they don't want to.

As much as Vercel's process of deployment is fully automated, they do give their end users the option of manually inputting build commands and allowing them to run tests and scripts at build time too. This option is another great feature of Vercel's build process as it allows the users to regression test their code at each push.

Once the code has then been built and is ready for deployment, this is where Vercel's Edge Network comes in. As previously discussed in this report their edge network allows users to get the lowest latency possible by their project being hosted to the nearest possible server to them to which there are over 100 across the globe. This and their Edge functions enable this low latency and allow features such as instant redirects, geolocation features, and location-based ads. All of this just adds to the seamless integration and deployment of a Vercel app.

So in summary, the steps of Vercel's deployment process can be seen in the diagram below:

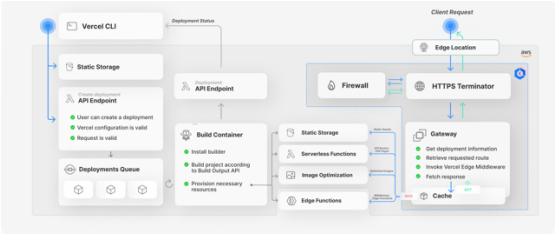

Fig 1. Vercel Architecture

As can be seen in the diagram above it outlines the steps that I have been talking about. First of all the deployment is triggered from a push in the Vercel CLI, then the API endpoints validates this deployment request and once verified it then goes to the build container where the new code is built and updated. This seamless process is what appeals to Vercel's front-end developer audience as everything is handled for them in the background.

Every deployment made with Vercel is labelled and documented. This is an extremely important point to note as it means that if a push is made that contains errors and needs to be rolled back, as everything is documented and labelled this is extremely easy to do which can put the developers mind at ease when pushing to a live environment. This architecture allows Vercel to provide that seamless rollout process that they market to their users.

IV. Use Case

In this section I will deploy an extremely simple HTML site just to demonstrate how easy it is to deploy using Vercel. For this demonstration I will be focusing on what kind of sites Vercel aims to focus on which is front-end sites so all I will be using is HTML and CSS. I will have a basic site which just displays Vercel Demo with a coloured background.

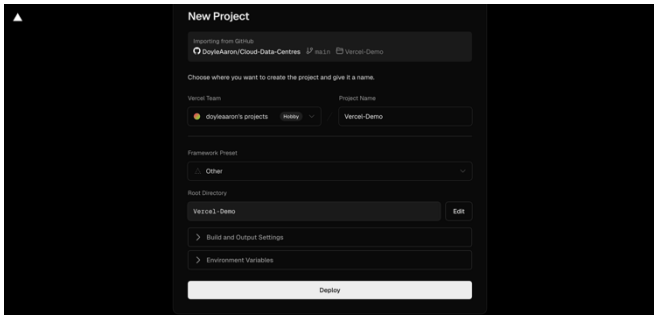

Fig 2. Vercel Setup Screen

As can be seen in the figure above this is how simple the deployment process is for a Vercel application, the only things that had to be changed here was the location to run the project from and the name of the project which I could have kept the same I just chose it for demo purposes. This level of ease of deployment is a huge part of why Vercel have been so successful there isn't any other application that has a deployment service than can be set up in three clicks. The deployed version can be seen in the figure below:

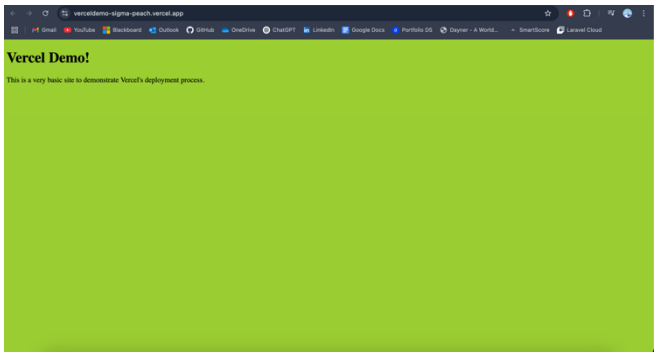

Fig 3. Deployed Vercel Site

There was no manual configuration required for setting this site up. Vercel automatically recognised that this was a HTML and CSS project and applied the correct build commands based off that. Now I have a fully deployed web application that can be accessed from anywhere in the world instantly because of Vercel's Edge network.

This is obviously a very simple and easy use case, but it does a great job in demonstrating how easy the process is to deploy an application. This use case demonstrates the topics that have been discussed previously in this report and how Vercel is accessible for front end developers who may even how no experience in back-end development or server management.

V. Challenges & Limitations

Vercel offers a lot of great features and resources for its users however there are some downsides [8] to Vercel's structure of deployment too.

Vercel does impose strict time limits on the free tiers of its service during build time. This may not seem like a big issue however as your project is expanding and the codebase is getting bigger and bigger this is eventually going to cause issues with their limits forcing users to pay for their upgraded tiers. The pricing system for Vercel is based off your usage on the server, that means if your app ends up getting very popular the prices could end up exponentially increasing very quickly without the knowledge of the user.

One of Vercel's big benefits is that it handles all of the backend configuration for you however for some users this may be a detriment to the service. This means that because Vercel is handling everything it doesn't leave the user with a lot of control over the backend functionality. If this is more of an intense app where a user may need to edit things such as any of the backend processes, they also don't have access to any server level logs which for some developers can be an issue.

Vercel's serverless functions also do at times tend to be prone to 'cold starts' [9]. Cold starts are when a function hasn't been used in a long time and then gets called, it must then start a new container, load the function, and then execute it. This restart process can take a little while and this will affect the performance of the application which can be an issue if the app uses any real time features.

Another downside that does impact Vercel is that they don't offer database hosting. This is inconvenient as it means that the user must go to another platform and then host and connect to their database there. They then must reconfigure their code to integrate with this new database. It isn't the biggest problem, but it is still an inconvenience for the user who then has to go and research where they can host their database that matches their needs.

The final big issue that I will discuss with Vercel is that it doesn't support a lot of languages. This is a problem I have personally run into with Vercel. If a project isn't in Vercel's list of accepted languages and frameworks the service simply can't be used. Some of the languages that aren't supported by Vercel are very popular such as Python, Ruby, Java and Go just to name a few. For example, if a user has a web application that they have running through Vercel, they then have no choice to migrate to another service if they want to add any Python code which is commonly used in web development. This is a major inconvenience to the user and is a major downfall of using Vercel. This is one of the factors that turns a lot of people away from using Vercel as modern web applications contain a multitude of languages and frameworks and Vercel simply doesn't have the capability to handle all of them.

These challenges and downsides to Vercel may not affect their target audience of front-end developers but it does hinder them from expanding into a larger audience of developers as there isn't a lot of support for bigger web applications that contain multiple languages.

VI. Conclusion

This report explored and discussed the web application service Vercel, its positives, negatives and how it works. I wanted to choose Vercel as I had used it in the past and found it to be so easy to set up for my simple web app and I wanted to see how it worked behind the scenes. The goal for this report was to outline how Vercel can seamlessly deploy web applications and integrate continuous integration and continuous deployment within their service.

This report has outlined how Vercel integrates with Git to allow for seamless deployment every time a push is made to the branch that the app is being hosted from. Vercel's serverless architecture allows for their audience of front-end developers to deploy their apps in less than five clicks and not have to stress about the management of the servers in the background. Vercel's edge network allows them to provide low latency performance on all their user's sites as they can host it form the nearest server to them no matter where they are in the world. With over 100 different server locations across the globe latency never ends up being an issue and this also allows for geolocation-based features on the site with ease. The use case in this report really highlights how simple and seamless Vercel's deployment service is with a front-end web application. Vercel's ability to automatically detect the framework that is being used for the application and then apply the correct build commands really adds to the seamless experience as it takes another step that traditionally would have to have been done by the user out of their hands. This use case showcased what had been discussed previously as to how Vercel seamlessly deploys the application within only a few clicks.

However, Vercel isn't perfect and that has been highlighted throughout this report too. Vercel has some big downsides to their seamless approach of deployment. Vercel does have limited backend control which for a bigger web application that needs a bit more control this could spiral into a big issue very quickly. The next big issue that Vercel has is that it doesn't support a lot of very popular languages such as Python, Ruby, Go or Java just to name a few. Some of these languages are very popular within web development so this completely rules Vercel put as a hosting choice if your project includes these languages. Vercel also prices based off usage in their paid models which can lead to users racking up very expensive bills quite quickly. Vercel also does not offer database hosting, this may not seem like a big issue, but it means that their users must go elsewhere to host it which is inconvenient for the end user as they must reconfigure their database to host it elsewhere. These issues may not affect every user, but they can be an issue for a lot of developers which could end up turning them away from using the service.

In conclusion, Vercel is a fantastic choice for front-end developers who want to host their smaller applications. Vercel's seamless deployment process by linking with Git to push when new code is uploaded to a branch is an intuitive method of deploying and takes a lot of the effort away from the end user. However, teams building bigger applications that contain different languages may have to look elsewhere for a hosting solution as Vercel currently does not have the capability to support them. Vercel moving forward could look to implement more support for these types of projects.

REFERENCES

[1] GeeksForGeeks, Difference between traditional hosting vs serverless hosting, GeeksForGeeks.com, 16th April 2024, available at: https://www.geeksforgeeks.org/difference-between-traditional-hosting-vs-serverless-hosting/

[2] Extio Technology, The power of Kubernetes auto-scaling, Medium.com, 20th June 2023, Available: https://medium.com/@extio/the-power-of-kubernetes-auto-scaling-scaling-your-applications-with-ease-cb232391400c

[3] GeeksForGeeks, What is CI/CD, GeeksForGeeks.com, 14th April 2025, available at: https://www.geeksforgeeks.org/what-is-ci-cd/

[4] Stephanie Susnjara & Ian Smalley, What is Continuous Deployment?, 28th August 2024, Available: http://ibm.com/think/topics/continuous-deployment#:~:text=Continuous%20deployment%20is%20a%20strategy,directly%20to%20the%20software's%20users.

[5] Akamai, What Is Edge Computing?, Akamai.com, Accessed at : 2nd May 2025, Available: https://www.akamai.com/glossary/what-is-edge-computing

[6] Vercel, Edge Network, Accessed at 2nd May 2025, Available: https://vercel.com/docs/edge-network

[7] Vercel, Behind The Scenes Of Vercel's Architecture, Accessed at: 2nd May 2025, Available: https://vercel.com/blog/behind-the-scenes-of-vercels-infrastructure

[8] Sushrit Pasupuleti, How,When and Why you should switch from Vercel, 5th February 2022, Available: https://medium.com/@sushrit.pk21/how-when-and-why-you-should-switch-from-vercel-to-a-different-hosting-provider-especially-for-8ba25e439788

[9] Vercel, How can I improve function cold start performance on Vercel?, Vercel.com, Accessed at: 2nd May 2025, Available: https://vercel.com/guides/how-can-i-improve-serverless-function-lambda-cold-start-performance-on-vercel