# Evaluation

The web application was successful in it’s goal to provide users with a platform which they could explore the vast contents of the D&D 5e universe from one simple and easy to use website. The website which is available at the web address defined within the project proposal is available for public consumption.

## Languages and frameworks

### Client

The client side of the application went very well and stuck to our project proposal almost to the letter. It was built using Angular JS and TypeScript to ensure we had an easy to maintain and scalable frontend solution which conformed to industry standards (Google, 2021). It successfully restricts routes to only users with correct levels of permissions and will redirect people without access to an area they can access.

It was also very good at ensuring dynamic content was loaded in a smooth and reliable manner which didn’t interrupt the user’s ability to interact with the application. It did this by ensuring all data was loaded using async calls and by using observables which allow for tracking the state of a variable and if they change, updating the DOM to reflect said changes.

The use of Angular Material ensured that our application was accessible and responsive on all tested devices and that we had a pleasant UI experience for all our users.

### API

The API had seen changes from the project proposal over the course of the project. The main one of these was the decision to build the API using PHP instead of the proposed us of Node and Express JS. The reasoning behind this is that I encountered issues with deploying a node solution to the web hosting platform provided. Luckily, this issue was discovered very soon into development and as such didn’t have as much of an impact on the project as could have occurred later in the pipeline.

The move to PHP encountered a few other issues such as ensuring we sanitised any data sent out to our database. To ensure our database was secure I used a PHP PDO for all database interactions. The reason for using this are defined in the PHP documentation as the secure way to pass user inputs into a database (The PHP Group, no date). This was done using SQL placeholders alongside the PDO.

## Testing

The application has been tested using simple black box testing methods of manually testing the web application against acceptance criteria and documenting the results. This is a good thing as it means that our features and functionality are working in the way we expect and that we have a documented process of this which can be audited in the future if the project wished to gain ISO 27001 status.

The website could however have been tested more thoroughly by developing a suite of unit tests for the custom API which interacted between the client and the database. UI tests for the client could also ensure that our UI is accessible and working as expected. The need for this however was limited due to Angular Material, the UI framework used for styling of the application, being open source and well tested by the community at large for issues and accessibly.

## Accessibility

Accessibility is a big topic on the web and rightly so. It is believed by both individuals and governments alike that everything on the web should be easily accessible to everyone, everywhere at any time. As such great care has been put into this project to ensure not only that the project adheres to the UK laws around accessibility and inclusion but also implements agreed best practises across the industry. This included but is not limited to: ensuring the website is accessible on all devices, ensuring that the site can be navigated by people with disabilities such as sight issues or hard hearing, allowing the site to be read aloud by a screen reader, making sure that the site’s elements are high enough contrast as to be seen easily, along with much more (UK Government, 2010).

The use of Angular Material makes this much easier that with raw basic HTML websites as a log of the accessibility tooling is built into the library itself. For example, all elements within the library are accessibility tested by hundreds of users across the globe and any issues identified are fixed within days thanks to its opensource nature (Google PLC, 2021).

## Modifications and Improvements

The application was an overall success but there are still some areas I believe could do with some modifications or improvements. First of all, I would want to recreate the API in Express JS or Laravel if I wanted to continue using PHP as these two frameworks are much better suited to maintainable software solutions than raw PHP and come with numerous perks as well (StrongLoop and IBM, 2017).

Another improvement I would like to add would be the use of GraphQL on the external API as they support it and we can then restrict the data we receive from them to only the bits we want to see, preventing having to deal with the huge JSON responses currently received from the requests (The GraphQL Foundation, no date).

Finally, as previously stated, I would like to add automated testing such as UI and unit tests to the frontend client and backend API so that they can be much more robust and more bugs and issues can be identified automatically without the need for an incident to be noticed by a user.

# References

* Google PLC (2021) Angular Material UI Components. *Available from:* [*https://material.angular.io/*](https://material.angular.io/)[Accessed 12December 2021]
* UK Government (2021) Equality Act 2010. *Available from:* [*https://www.legislation.gov.uk/ukpga/2010/15/contents*](https://www.legislation.gov.uk/ukpga/2010/15/contents)*.* [Accessed 12 December 2021]
* The PHP Group (no date) PHP:PDO – Manual. *Available from* [*https://www.php.net/manual/en/book.pdo.php*](https://www.php.net/manual/en/book.pdo.php)[Accessed 12 December 2021]
* Google (2021) AngularJS – Superheroic JavaScript MVW Framework. *Available from:* [*https://angularjs.org/*](https://angularjs.org/)*.* [Accessed 12 December 2021]
* The GraphQL Foundation (2021) GraphQL | A query language for your API. *Available from* [*https://graphql.org/*](https://graphql.org/)*.* [Accessed 12 December 2021]
* StrongLoop, IBM (2017) Express – Node.js web application framework. *Available from:* [*https://expressjs.com/*](https://expressjs.com/)*.* [Accessed 12 December 2021]