

A Virtual Reality eLearning Authoring Tool Prototype

Euan Macqueen

School of Computing Science

Sir Alwyn Williams Building

University of Glasgow

G12 8RZ

A dissertation presented in part fulfillment of the requirements of the Degree of Master of Science at the University of Glasgow

06/09/2019

**Abstract**

Virtual reality has been one of the most significant emerging technologies of the last decade. Though, with the key areas of study being hardware and software systems, the creation of virtual reality content is limited to those with knowledge of programming. A key area with known major advantages of utilising virtual reality is education; more specifically, eLearning. E-Learning is the accessing of educational content via an electronic device. It aims to enhance learning in comparison to traditional classrooms with increased interactivity, realistic practice and contextually meaningful scenarios.

This paper proposes a code-free virtual reality eLearning authoring tool web application which hopes to offer inexpert users the ability to author virtual reality eLearning content. The authoring tool will incorporate user and course management, an authoring tool “scene editor,” and will export content to modern learning management systems with well adopted eLearning formats, xAPI and SCORM.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic form.

Name: Euan Macqueen Signature:

Acknowledgements

Firstly, I would like to thank my supervisor Michele Sevegnani for his feedback and guidance throughout each step of this project.

Additionally, I would like to thank all participants who supported the evaluation process. The feedback you all contributed assisted in shaping the final product. Without your support, it would not have been achievable.

Furthermore, I would like to express my sincerest gratitude to my friends at The University of Glasgow for their constant support.

Finally, I would like to thank my partner, Amanda, and my family for continuously supporting me through all my studies.

Contents

Chapter 1 Introduction 1

1.1 Overview 1

1.2 Project Objective 1

Chapter 2 Background 2

2.1 Virtual Reality 2

2.2 E-Learning 2

2.3 E-Learning Authoring 3

2.4 Existing Authoring Tools 3

2.4.1 CenarioVR 3

2.4.2 Adobe Captivate 2019 4

2.4.3 Uptale 4

2.5 Requirement Gathering 5

2.5.1 Overview 5

2.5.2 Interviews 5

2.5.3 User Stories 5

2.6 Requirements 6

2.6.1 Key Features 6

2.6.2 Wireframes 7

2.6.3 Justification 7

Chapter 3 Design Approach & Implementation 8

3.1 System Architecture 8

3.2 Database Architecture Design 10

3.3 Design 11

3.3.1 User Centered Design 11

3.3.2 Responsive Design 11

3.3.3 Scene Editor 11

3.4 Implementation 12

3.4.1 Scene Editor 12

3.4.2 Database Implementation 12

3.4.3 Repository-Service Pattern 12

3.4.4 VR Asset 13

3.4.5 Export to xAPI/SCORM 13

Chapter 4 Testing 14

4.1 Unit Testing 14

4.2 Smoke Testing 15

4.3 Usability Testing 15

4.4 Results 16

4.5 Known Bugs 17

Chapter 5 Future Work 18

5.1 Scene and Asset Duplication 18

5.2 Background Video 18

5.3 Rearrange Scene Order 18

Chapter 6 Conclusion 19

6.1 Final State 19

Appendix A User Stories 1

Appendix B - MoSCoW Statements 3

Appendix C - Wire Frame Development 5

Appendix D - Mock Scene for Usability Testing 7

# Introduction

## Overview

Virtual Reality (VR) is one of the most dramatic emerging technologies of the last decade [1]. Virtual reality is a modern technology which offers digitally produced environments to be experienced by users through tricking the user’s senses into believing the fictional environment is real [2]. By using dedicated hardware and software, users can experience virtual environments which would be deemed unsafe, costly to visit or impractical to be experienced in real life [3]. Though virtual reality is becoming more accessible, the creation of virtual reality eLearning content is still limited to those with knowledge of programming, and the few solutions that exist are primarily focused towards those with design or technical knowledge.

An industry that has the potential to benefit from virtual reality is E-Learning.   
E-Learning offers users access to educational content via an electronic device.   
During the industries inception, the intended users of eLearning systems were students in early education such as primary and secondary school; however, in the last decade, the audience has extended towards companies to deliver online courses to their employees [4]. With such an increased acceptance of eLearning, the authoring of eLearning content has naturally become an anticipated area of study. As virtual reality is becoming a widely adopted technology, this study believes that the use of virtual reality eLearning content will too become widely adopted within an educational context.

A fundamental challenge when authoring any eLearning content is engaging the learner. This is apparent through the various research studies on increasing learner engagement in eLearning [5] [6] [7]. The engagement of the learner refers to the level of attentiveness that a learner shows when progressing through activities [8]. The inclusion of virtual reality in eLearning has the potential to greatly increase learner engagement as research shows that learner engagement and retaining information rely significantly of the methods used for content delivery [9]. These methods are “providing learners with realistic practice, spaced repetitions, contextually-meaningful scenarios, and feedback” [9]. This study believes that virtual reality eLearning content can offer these methods.   
Though VR accessibility is increasing, the designing and coding of virtual environments is still a complex task.

## Project Objective

The purpose of this study is to develop a virtual reality eLearning authoring tool to offer virtual reality eLearning content design capabilities to inexpert users.   
The development stages of this paper are Background Research and Requirements Gathering, Design Approach and Implementation, Testing, Future Works and Conclusion.

# Background

## Virtual Reality

As previously mentioned, Virtual reality (VR) is a technology which offers digitally produced environments to be experienced by users by means of dedicated hardware and software [10]. Progress regarding the accessibility of VR has allowed for the technology to be successfully adopted in various industries such as entertainment, education, architecture and many more [11]. Recent webVR development frameworks, such as A-Frame, webVR, Primrose, etc., allow for VR web applications to be developed by including a JavaScript API/Library [12] . Due to these new frameworks, VR is now at a stage where virtual reality can be incorporated into web applications with minimal overhead.

## E-Learning

E-Learning offers users access to educational content online. It offers both employers and academics a cost-effective method of providing training/education to staff and students [13]. Though eLearning is widely adopted by various industries, there are some doubts surrounding eLearning. The most common doubts are the self-discipline required by users, the lack of face-to-face interaction and a lack of flexibility [14]. However, many studies confirm that the assistance of technology in education/training can greatly support a learner’s education [15]. The relationship between learner engagement and retaining information has been a well understood concept for decades: unengaged learners retain less information than engaged learners [16].

Enhancing learner engagement with virtual reality is a fairly new concept, but one with significant research to support it, with Conrad and Donaldson summarizing the work of learning theorists; Burner, Vygotsky and Piaget, as all embracing the philosophy that humans learn through interaction [17]. Additionally, Egdar Dale’s most acknowledged contribution to education was the cone of experience [Figure 1], which suggests that learners using abstract actions such as reading, hearing, etc. retain 60-80% less information than actions nearer the concrete activity, such as simulating the real experience [7].

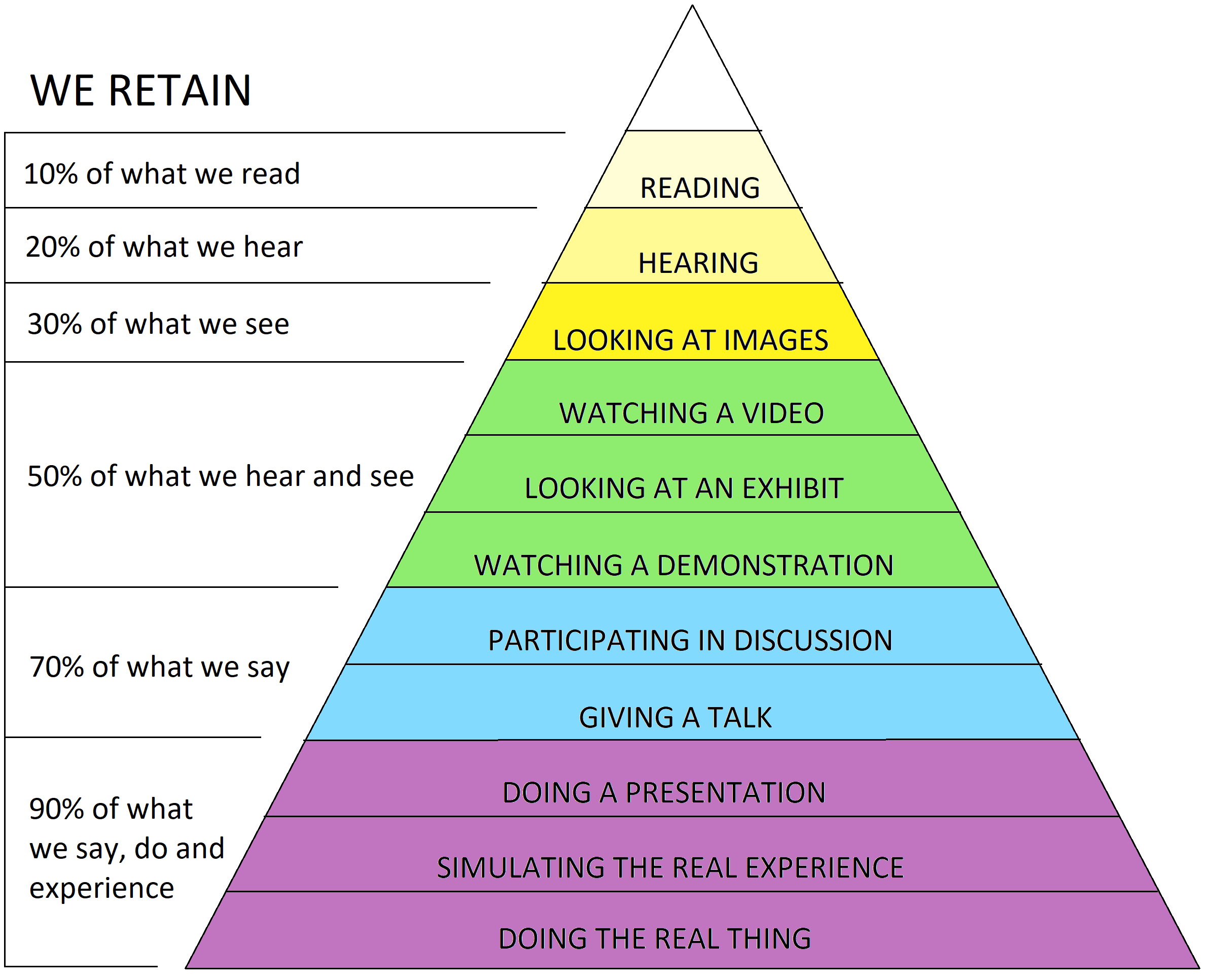


Figure 1 : Adaptation of Edgar Dale's Cone of Experience [18]

## E-Learning Authoring

As eLearning plays a large role in countless organizations, standards were introduced to assist with the authoring of eLearning content to offer interoperability between Learning Management Systems (LMS). Learning Management Systems are the applications used to access and manage eLearning content [19] . The most prevalent data standards are SCORM, xAPI (Tin Can API) and AICC [20]. The earliest of the standards, SCORM, offers basic user tracking such as tracking quiz scores and pass or failure of a quiz; however, a newly emerging data standard, ‘xAPI’, offers developers complete data tracking flexibility with JavaScript statements formatted with an ‘Noun, ‘Verb’, ‘Object’ [21]. The noun is the current user, the verb is the event, and the object is the relative object, i.e. – “The author has viewed Virtual Reality Scene 3’s hotspot object.” This flexibility of user tracking provides a solid foundation for virtual reality within an eLearning context.

## Existing Authoring Tools

An essential area of this study is the analysis of similar software systems. The strengths and weaknesses of these systems will be analyzed and evaluated to provide a solid foundation when progressing to the requirements gathering stage.

With the rapid increase of VR accessibility, VR authoring applications are on the rise. Although there are several web applications that offer VR authoring capabilities, the three most popular solutions will be discussed below. The following authoring tools all offer exported content as xAPI, SCORM, cmi5 and HTML.

### CenarioVR

CenarioVR is a virtual reality authoring tool which allows users to create courses which contain numerous virtual scenarios. The application dashboard offers users a clean display of the courses available for editing as well as search and filtering capabilities; however, as the navigation bar contains headings, ‘My Scenarios’, ‘Assigned Scenarios’, ‘Public Scenarios’ and ‘Shared Scenarios,’ there is some ambiguity regarding the navigation throughout the application.

Within the scene editor, users can add basic assets such as background image/video, hotspot, question card, info card, audio, image and video to the scene. Users are then able to edit the properties of the assets such as editing position, rotation, scale as well as unique functionality dependent on the object such as font, font colour, etc. In comparison to other VR authoring tools, CenarioVR’s asset availability and customization lacks complexity; However, this offers increased usability by not overwhelming the user. Additionally, there is an abundance of flexibility triggered by hotspots, i.e. A link to another page, play media, activate an animation, hide an element, show an element, etc.

To edit assets within the virtual scene, assets are selected directly with the mouse and controls appear for rotating and positioning. This offers usability in terms of enabling editing; however, it lacks the accuracy a lot of inexpert users may require when designing.

Aside from asset editing, CenarioVR’s user interface is very easy to use, with a clear summary of existing scenes and assets. However, only small scenarios are visible for demo purposes and it is possible that once scenarios become too large, it may be difficult to manage a course or scene.

### Adobe Captivate 2019

Adobe Captivate 2019’s newest feature, as part of a recent upgrade, provides the ability to create virtual reality eLearning projects. It offers enhanced asset availability in comparison to CenarioVR; with each asset type offering varying output such as three different types of text objects, hotspots which offer numerous events, multiple types of media uploads and more. Though the editing capabilities of Adobe Captivate 2019 are never ending, the interface offers too much functionality which may be daunting to new inexperienced users. Like Adobe’s other applications, it requires technical and design skills to navigate and create professional content. Furthermore, eLearning content designed with AC 2019 often comes out with very little personalization as each hotspot lacks a flexible design. Furthermore, the editing capabilities of AC2019 are far more than the average user requires; however, this application seems to be targeted towards far more advanced users.

Though the icons within a scene provide little personalization, they provide a clear view of interactive objects and offer an uncluttered scene, so users are not overwhelmed.

### Uptale

Uptale is one of the highest rated VR authoring tools available as it offers extensive asset availability and flexibility when creating content. In comparison to Adobe Captivate 2019, Uptale’s content offers additional customization and personalization. However, Uptale’s interface severely lacks clarity and the application navigation is very unclear, as shown in their video [22]. Uptale’s scene editor contains various controls which could easily overwhelm users. Furthermore, the scene editor page displays small icons, small font and various dropdowns whilst wasting large amounts of the screen on white space. User reviews have also mentioned that Uptale has a steep learning curve and, to use all the features it offers, technical knowledge is required [23].

## Requirement Gathering

### Overview

Following on from the background research and review of existing systems, this chapter aims to discuss the various requirement gathering techniques employed to retrieve functional and non-functional requirements. As this was a greenfield project, due to it being a new project and not being constrained by previous work, the most applicable project management methodology was found to be agile [24]. Agile was selected as it provided “the ability to create and respond to change,” [25] and as this application was heavily reliant on research and followed an iterative development plan, it was highly anticipated that the development would undergo frequent alterations. The waterfall model was also an available management option as it offered sequential stages which aligned with this projects development plan, namely being requirement gathering, analysis, design, coding/implementation, Testing, Deployment and Maintenance [26]; however, as waterfall does not allow short notice changes and alterations to previous iterations [26], agile was selected as the most suitable management option.

The requirement gathering techniques adopted were user personas and client interviews, which were developed into both user stories and non-functional requirements. To document the initial requirements, a requirement document was created. This document contained information regarding the project scope, assumptions, constraints, hardware/software requirements, non-functional requirements, early user stories, development plan and covered early prototyping of the wireframes. The requirement document is discussed more thoroughly later in this chapter. Once a solid foundation of the project was made, the requirements were developed into user stories using Trello to provide the product backlog.

### Interviews

After reviewing the existing systems, interview questions were generated for two clients: a construction firm and a manufacturing company; both looking to incorporate virtual reality into the training of their staff. As the application is aimed towards inexpert users, no scene editing users were suitable for interview as they themselves did not understand the requirements. The interviews allowed for varying perspectives of the necessary requirements of the system and offered a hypothetical walkthrough of the potential scenarios that the application may encounter. The interview documents can be found in the attached as “Construction Interview.doc” and “Manufacturing Interview.doc.”

### User Stories

Once the interviews were complete, the results were shaped into the requirement document which contained the early user stories. Once a firmer understanding of the system was set, the user stories were transferred to Trello to initiate the applications development. The user stories stored on Trello were used as a product backlog which displayed the prioritized stories as well as notifying the author of the current state of the applications progress. This provided a platform to assist with meeting the minimum requirements and displaying the areas of functionality still to be tested or completed. The user stories were assigned levels of must, should, would and could have to provide further clarity of the minimum requirements and can be found at both Appendix A and Appendix B.

## Requirements

The initial draft of the requirement document outlined the essential non-functional requirements of the system as well scope, assumptions, constraints, hardware/software requirements, early user stories, early prototyping of the wireframes and an iteration plan. The document provided an overview of the aims of the system and ensured the non-functional requirements were always kept in mind. Following is a list of the key non-functional requirements.

**Usability**   
- User-Friendly Interface & Navigation   
- Clear Error Display  
- Informative Design  
- Information Density

**Scalability  
-** Sensible Database Design **-** Sensible Database Queries

**Interoperability  
-** Content must offer controls to multiple VR hardware devices. **-** Exported Content must be exchangeable between LMs’ using modern standards.

**Security  
-** User authorization  
 **Maintainability  
-** Efficient system and data architecture to allow for change.  
- Loose coupling and High Cohesion.

**Manageability**:  
- Standard users must not be able to edit user and or course data.  
- Control over course and scene management

After reviewing the first draft of user stories, an iteration plan was created to assist with time and progress monitoring. The plan was made to allow for 7 weeks of development, aiming to complete development on 23/08/2019 to allow for a 1-week for tidying up. All aspects discussed above can be found in the document “ReaLearn Requirements.” The actual development progress is discussed in chapter 3.4.

### Key Features

As the project developed, the features of the application were more thoroughly understood, and user stories were updated accordingly. The final types of actors associated with the application were general user, standard user, administrator, scene designer and the passive content viewer.

The general user actor was used for user stories relating to user authentication. Standard users were actors whose accounts had been added by admin users. Standard users had limited access privileges but could edit courses that had been assigned to them by admin users. The admin acted as the main account holder and held control of the editing, deleting and adding of courses and standard users. Scene designers could be both admin or standard users and were included for all user stories regarding the scene editor features. As the application’s core functionality was scene editing, an excessive number of user stories regarding scene editing were generated to make sure all essential features were included in the final product. The content viewer was included as a passive actor to serve as any user viewing the exported content. This actor assisted in providing a new perspective of the system and supported a deeper understanding of the requirements regarding scene interactions. The final user stories can be found at appendix A.

The key features of the application are influenced by the previous requirement gathering stage and are as follows:

1. Allow Secure User Authentication.
2. Offer User Management with regards to registration, updating, deletion and adding new users.
3. Course Management with regards to adding, updating, deletion.
4. Scene Management with regards to organization, adding, updating, deletion.
5. Asset Management with regards to usability, organization, adding, updating, deletion.
6. Universal design for inexpert users to offer increased usability.
7. Scene Editing (MINIMUM REQUIREMENTS):
   1. Add Background, Text, Image, Video, Audio and Questions to scene.
   2. The application must offer editing capabilities of object position, rotation, scale and colour.
   3. Hotspot objects must allow the following events:
      1. Show object on click or view.
      2. Hide object on click or view
      3. Play media
      4. Stop media
      5. Link to Scene
8. Assign courses to standard users.
9. Export to appropriate standard: xAPI.

### Wireframes

As this application is primarily targeted towards inexpert users, usability is one of the applications most vital non-functional requirements. Because of the significance of usability, wireframes were developed to provide early prototypes and offered frequent review and alterations to be made quickly. The wireframes were designed to cover the primary areas of functionality within the application: namely, the scene editor and the asset editor boxes. As the scene editor’s role in the application was significant to user experience and usability, various versions of the wireframes were developed. The development of the wireframes can be found at Appendix C.

### Justification

For appropriate application management, user authentication is required as it provides the appropriate identification which offers a link between user accounts and courses. The management features within the proposed application support the non-functionality requirements usability, security and manageability. As the application seeked to provide a more-straight forward solution in comparison to the existing systems, universal design is of the utmost importance to allow inexpert users to create content seamlessly. The minimum requirements for the scene editor have been proposed as each asset type provides numerous benefits to the user. Though this is a large amount of functionality, to remove any one of the assets would hinder the application’s core purpose. Assigning courses to standard users is a necessary stage in eAuthoring as it provides the desired management of content and assists with security regarding access rights and account privileges. As the application employs virtual reality, a flexible data tracking standard such as xAPI is vital in any modern eAuthoring tool.

# Design Approach & Implementation

Following the requirement gathering stage, this section will discuss the proposed system architecture, database design and front-end design. Following these areas, the implementation of the system will be discussed, highlighting key areas of functionality.

## System Architecture

System architecture refers to the organization and design of a system. It encapsulates topics such as design patterns, services, layers, frameworks, libraries and the relationships amongst system elements. To achieve a maintainable and scalable code structure, as set out by the requirements, a well-defined system architecture must be implemented [27]. To design a well-structured system architecture, layers must be introduced to separate the concerns [28]. Various sources conclude that the introduction of layers into a system architecture provides improvements in flexibility, maintainability, testability and scalability [28] [29].

During early prototyping, the application adopted the ASP.net platform which offered various tools and libraries to assist with the application’s development [30]. It also offered the .net MVC framework which provided the desired tiered architecture by adopting the MVC design pattern.

The MVC design pattern was chosen as it offered the separation of the model, the view and the controller, which allowed the application to adhere to various design principles, namely Divide and Conquer, Increase Cohesion and Reduce Coupling [31]. Alterations made to any one of the separated elements ensured that the modification did not affect the rest of the solution [32].

* The **Model** layer represents the logic of the application’s data domain: data models stored in the database.
* The **View** layer represents what’s displayed to the client: the user interface. The user interface retrieves data from the model via the controller layer.
* The **Controller** layer handles user interaction and controls how data is handled between the view and the model.

Further research into design patterns uncovered that an additional design pattern, the repository-service pattern, offered additional layers which could support the non-functional requirements scalability and maintainability in addition to testability [33]. According to Microsoft’s documentation, the service layer is inserted between the controller and the database [34]. The service layer is then responsible for data validation before accessing the repository, which acts an abstraction of the data layer and is responsible for accessing the database [34]. Data validation is a vital part of developing an application as it ensures that all data requested/returned is clean and accurate [35]. To use the service-repository pattern correctly, the controller accesses the service layer which validates requests and, if valid, sends requests to the repository layer, which then sends to request to the database. As the application relied heavily on database requests for courses, scenes and various scene assets, the decision was made to implement the repository-service pattern.

To implement the pattern with loose coupling, a technique known as dependency injection was required [36]. Dependency injection is a technique where a class is made independent of its dependencies [37]. Dependencies (objects providing a service) are passed to a class from another area of the system without the need to be instantiated each time they are required [38].

However, ASP.net MVC did not offer support with dependency injection. Because of this, at week four of development, the decision was made to migrate to ASP.net Core MVC 2.2. This offered a suitable framework fordependency injection whilst maintaining the benefits of ASP.net MVC. Once the migration was complete, the application employed both the MVC and the repository-service design patterns. Below, figure 2, displays the process of storing a scene’s background image with the implemented Model-View-Controller and the Repository-Service layers.

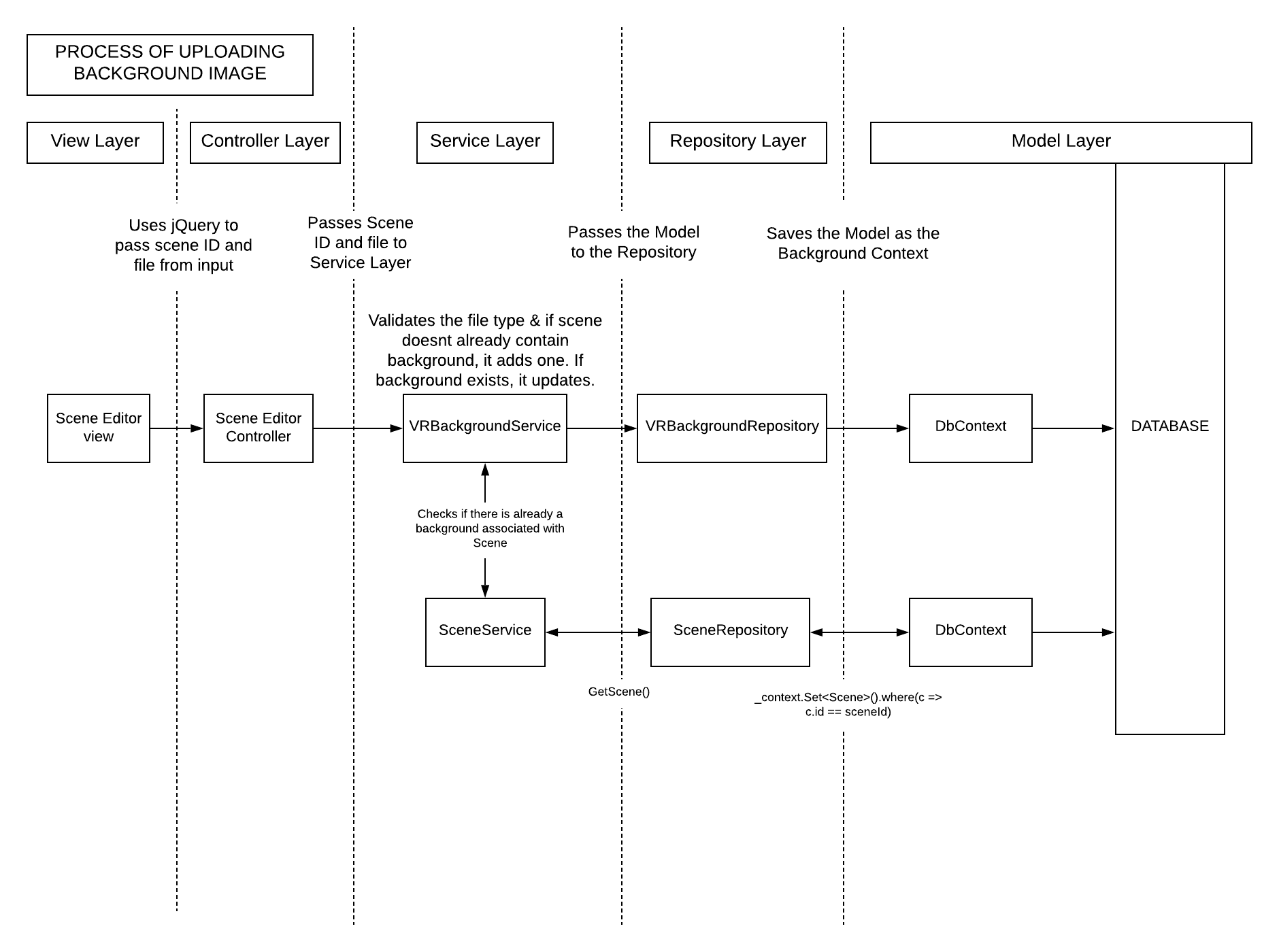


Figure 2 – Function call with MVC with Repository Service Layer

Due to a lack of experience regarding application security and authorization, the Identity Framework was utilized to provide functionality for user security and authentication. The framework provided a robust database structure for users as well as functionality and views for user authentication. This provided the application with a foundation for meeting the non-functional requirement of security. Another framework which supported the application was the Entity Framework. The Entity Framework is an object relational mapper which offers both code-first and database-first approaches for database design. The benefits of code-first were increased productivity, cleaner code and no need to manually access the database [39]. Some advantages of the database-first approach are the avoidance of data loss and it can be easier to handle more complex data models. Due to the productivity benefits and this application not anticipating complex data models, the decision was made to follow the code-first approach. To implement the virtual reality features, the webVR framework A-Frame was adopted. This was due to its extensive documentation, it’s online support and, as its functionality is based on HTML and JavaScript, this made it suitable of eLearning content. Furthermore, A-Frame offered controls to HTC Vive, Oculus Rift, Oculus Quest, Oculus Go, Google DayDream, Samsung GearVr, Vive Focus as well as mobile. This made it the suitable choice to meet the requirement of interoperability between VR devices.

The above frameworks along with the MVC design pattern served as the first steps of the solution meeting the non-functional requirements of maintainability, security, reliability, interoperability and scalability. Implementation of the above design patterns and scene editor are discussed in Chapter 3.4.

## Database Architecture Design

Database design is often one of the most important areas of an applications software development. This is because the database is responsible for vital data handling of the system. Without an appropriate database design, maintainability, reliability, security and almost all other aspects of an application suffer.

To optimize data access and maintain data integrity, the final database architecture was logically divided into 8 elements: Customer, Application User, Roles, Course, Scene, VR Object, VR Background and VR Question Response.   
By utilizing inheritance during development, entities in the database were subtyped from hierarchizing entities. This allowed for entities to extend functionality. For example, image assets extended from the VRObject asset which offered shared control over different types of assets and, as a result, increased usability when editing scenes within the application. This also offered an increase in code maintainability through increasing cohesion. The final relational database design is displayed below in figure 3. Primary keys are identified with red, foreign keys are identified with yellow.

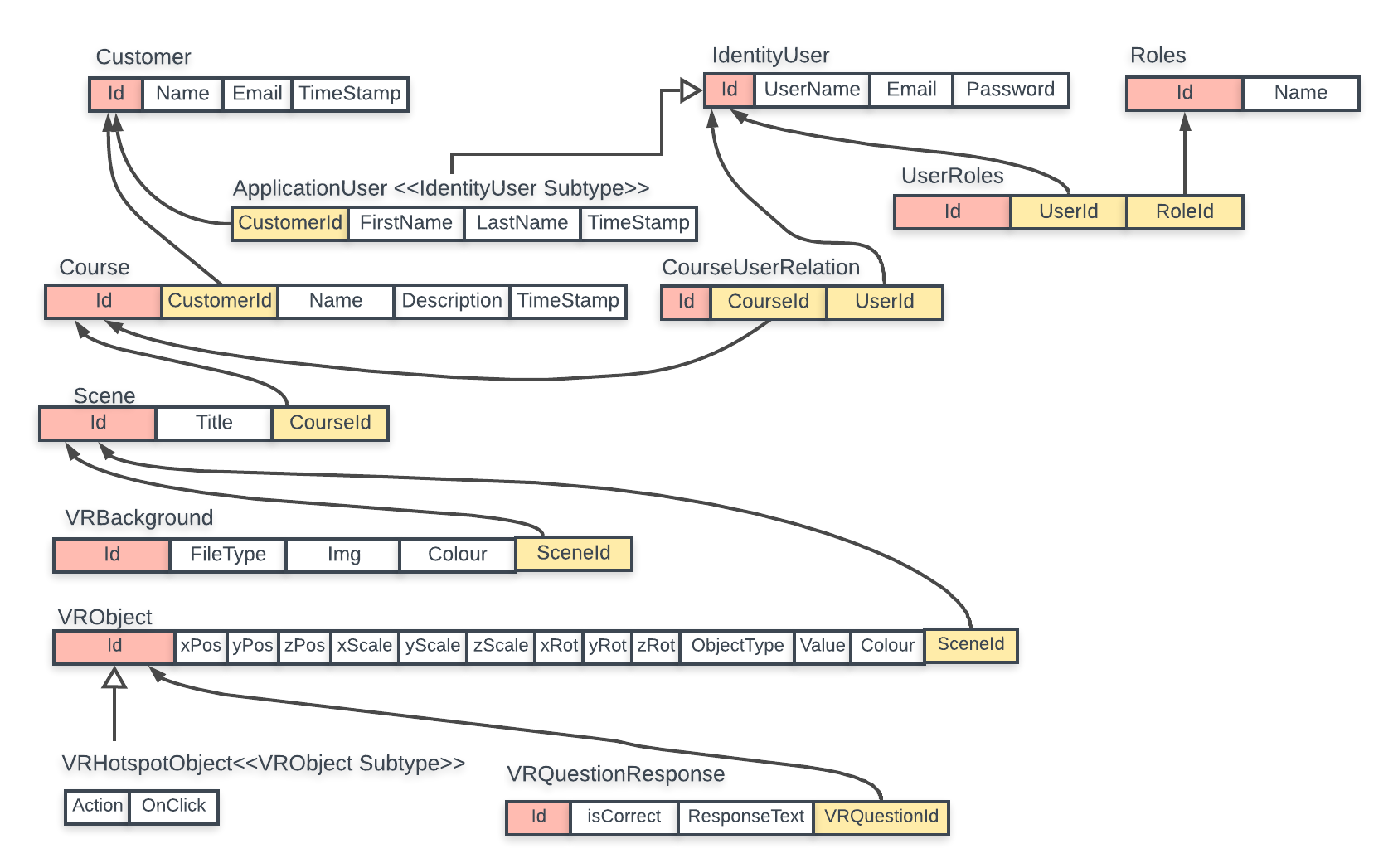


Figure 3- Relational Diagram of ReaLearn’s Database Structure

As the adopted Entity framework supplied data models with a compulsory ID attribute to offer unique tuple identification, the above database is in the third normalization form. This is apparent due to all attributes being atomic, thus being in the first normal form; the database having no partial dependencies, thus being in the second normal form; and having no transitive dependencies, thus being in the third normal form [40]. The removal of partial and transitive dependencies considerably reduces the number of update, insert and delete anomalies and increases data integrity [41]. Though the database design offers normalization up to the third normal form, it isn’t in the Boyce-Codd Normal Form due to some attributes not being solely dependent on the primary key. For example, as a user’s email and username are both unique, a password could be retrieved from either the email, the username or the ID; however, as the use of ID’s was compulsory, the database could not be normalized further.

## Design

As revealed during the requirements capture, usability was emphasized as one of the most important requirements. Prior to implementation, the author considered various designs of the application through wireframes and prototyping. To provide a usable experience to inexpert users, the author aimed to adopt a user-centered design approach.

### User Centered Design

User centered design is an approach that focuses on the needs of the user during development. To properly adopt a user-centered design, four steps must be taken: identifying and understanding the user of the system, specifying the user’s needs, creating the design and then a thorough evaluation to understand their usability [42]. All design elements must serve a purpose to the user and not clutter the view. To follow a user centered design, users should be involved through each design iteration and provide feedback [43]. Due to limitations, this was not an option. Instead, each design iteration was thoroughly tested through smoke testing and user evaluations were included during the final two design iterations to ensure the requirements were met.

### Responsive Design

Responsive design is an important aspect of modern web applications. Responsive design allows users the flexibility to access websites on a variety of devices. The application aimed to offer a responsive design between large and small monitors. As the intended use is not applicable to mobile devices, they were not considered during the development process. Though the application is primarily targeted towards large monitors, smaller screens were still considered during development by employing Bootstrap 3.4.1.

### Scene Editor

As the scene editor offered a large amount of functionality, it presented the possibility of overwhelming the user. To negate this, the scene editor was decomposed into 3 elements: existing scenes, the scene and the asset editor carousel. Each element was then reviewed individually which allowed for a deeper understanding of its core functionality and how the user would interact with it. The existing scenes component’s only functionality was to provide scene management within a course. Through analyzing the component as an individual entity which was only to provide this functionality, all cluttering elements were removed. The only elements remaining in the component were the scenes, an add scene button, a launch scene button and a remove scene button.  
The scene component displayed the active scene along with controls for adding assets, saving the scene, exporting and changing background properties. To provide all necessary controls to meet the requirements, whilst negating overwhelming the user, icons were implemented over text buttons. Through testing, this was confirmed that it maintained all must have functionality whilst ensuring the application exceeded in usability.

The most complex interface of the application was the asset property editor carousel. Its main purpose was to display each asset and the asset’s editable properties. A review during the usability testing highlighted that there were a lot of unnecessary buttons displayed in previous designs. This led to refactoring which adopted icons rather than text buttons which provided a cleaner interface with less clutter. User’s were provided various elements of functionality in editing assets, one of which was asset rotation. Controlling an objects rotation through the axes can be difficult for inexpert users. As rotation of assets was a must have requirement, it had to be implemented well. However, common HTML elements such as sliders or grab controls did not offer an appropriate design for users to handle rotation efficiently. Through trial and error of different approaches, it was decided that the best option for usability would be to provide users with a basic object, clearly displaying each of its sides, and mirroring the rotation action onto the asset within the scene. This was incorporated with use of the three.js library for rendering 3D objects in html. The functionality also originated from a solution on jsFiddle.net, which provided the mouse grab and 3D cube functionality [44].

## Implementation

This section will aim to discuss key implementation areas of the above application.

### Scene Editor

Due to its complexity, the scene editor view was separated into various partial views. The separation of views into partial views is recommended as it provides code reusability following the Don’t Repeat Yourself principle [45]. Furthermore, it allows a complex page to be decomposed into logical elements which offers more maintainable code [46]. Partial views were created for each modal popup, the scene, the existing scenes column and the asset editor carousel. This offered increased usability as it allowed for data models to be served to specific components via jQuery without interrupting the user’s workflow. An example of this is when a scene is added or removed, only the existingScenes partial view was updated. The separation of the modals into partial views allowed for easier maintenance of the specific functionality required for each asset. However, as some alterations made to the scene required the A-Frame framework to reload, this disrupted the user’s workflow. It is hoped that as the framework is in early development, future releases will improve this load time.

### Database Implementation

As stated in the ASP.net documentation, code-first migrations is the recommended method to develop your database schema [47]. The continuous method of code-first migrations allows you to incrementally evolve the database as the data models are updated throughout development. This supported a more productive workflow and allowed a cleaner control over the database entities and attributes by adopting code data models which were passed the ApplicationDbContext class.

### Repository-Service Pattern

The repository service pattern provides various benefits to the system, but only if implemented correctly. The purpose of the service layer is to handle all business logic and validation [48]. The repository layer serves as an abstraction of the data layer and should only be used for accessing and returning queries to and from the database [49]. Furthermore, each service class should only have a direct association with its one repository counterpart. For example, the CourseService should only call on the CourseRepository. All other database requests to alternative repositories should be routed to the desired service class to retrieve the data required. The above methodology for appropriate data handling between repository and service classes has been implemented as best possible.

### VR Asset

A key area of functionality within the scene editor was the adding of VR assets. As displayed in figure 3, the VR Object was the superclass. This allowed for various asset objects to inherit the required fields such as position, rotation, size and value. Furthermore, it allowed for a flexible design of the carousel by retrieving an IEnumerable of the VR objects. This offered various good practice design principles such as the DRY principle (Don’t Repeat Yourself), reusability and testability.

The asset editor boxes within the carousel also followed good practice regarding the DRY principle. For each VR asset returned from the database, the same card was generated in the view. As the VR assets stored the objectType as an attribute, the view could validate the type of object and select what properties to display on the card. This allowed each card to maintain a near-identical look with minute changes dependent on the type.

To add VR assets to a scene, the foundation of the entity was hard-coded, fig 4. This allowed for the values that designed an asset to be inserted into the pre-designed entity. This provided a flexible solution which allows for the possibility of adding new features in the future.

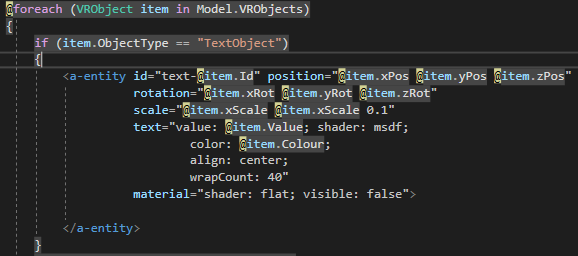


Figure 4 - VR Text Asset Code

### Export to xAPI/SCORM

To implement the exporting functionality of the application, the author requested a SCORM/xAPI wrapper from Rustici Softwares. Rustici provided a non-commercial xAPI wrapper which served well for test purposes. The wrapper contained various files which allowed for appropriate data tracking of content. To export a course, the ExportController was called which gathered the database entities associated with the course: scenes, assets, background images etc. From there, the ExportController looped through each scene and populated the Export.cshtml view with the assets and background image by using the scene’s primary key. Once the Export.cshtml had been populated with the scene’s assets, the page was read and returned a string which created a html page inside the ScormPackage/scormcontent folder, as was required by the xAPI wrapper [50]. This allowed for a flexible approach which did not limit the number of scenes a user could create.

# Testing

To confirm that various elements were correctly implemented, the system introduced unit testing. Following unit testing, smoke testing was incorporated as this allowed the application to walkthrough various test cases of expected functionality. Though this is a generalized form of testing, it is still an important stage in the testing process. Next, two stages of usability testing and evaluation commenced. This allowed the application to be tested for usability and general design. Due to time limitations, no integration testing was involved.

## Unit Testing

Unit testing is a form of testing where specific elements are tested whilst disregarding the elements dependencies [51]. This allowed for individual areas of functionality, “Units”, to be tested without concerning the database or the other methods. As the solution adopted the repository-service design pattern, this offered significant improvement in terms of testability.

To involve unit testing, the popular unit testing framework XUnit was adopted. To allow the unit testing project access to the applications classes, a separate unit testing project was created under the application’s solution. As unit testing is a very time-consuming task, only the four vital service layer classes were tested: CourseService, SceneService, VRBackgroundService and VRObjectService. As the service layer performed validation before accessing the repository layer and held the primary business logic of the application, this was deemed the most appropriate layer to test. However, if this application was to be deployed, integration testing the repository layer would have taken place. To test the service layer, returning functions such as GetCourse(int id) and GetBackgroundImageWithSceneId(int id) were tested. The author created separate test classes for the necessary service classes which provided more maintainable tests. As the application involved dependency injection, the Moq framework was adopted to mirror repository functionality whilst ensuring the dependencies did not corrupt the unit tests.

As applications undergo various scenarios, each scenario must be suitably tested so the system knows how to react. To test these various scenarios, numerous unit tests were created for each method being tested. An example of this is the CourseService’s GetCoursesWithCourseIds(List<int> ids) method. As the service layer should only return the courses where an ID is present, this method underwent testing scenarios such as sending ID’s, expecting success; sending 3 ID’s, where only two are expecting success; sending all incorrect ID’s, expecting null; and sending an empty list, expecting null. This style of testing various scenarios with different expected outcomes was completed for all methods involved in testing.

## Smoke Testing

Smoke testing is a generalized method of testing used to uncover basic errors in software. Smoke testing uncovers various basic errors at runtime such as “Does the add user modal open?”, “Does the asset add to scene?”, “Does this navigation button take me to the correct page?”. It provides a very quick means of testing and allows for the documentation of errors and the steps taken to replicate them. This allows for real scenarios to be tested for errors and, if found, removed.

Smoke testing was carried out continuously through development; however, with unit testing complete, the application was at a state where more in depth smoke testing was applicable. To adequately smoke test the exported content of the application, the author created a free learning management system environment with Rustici Software’s. This allowed the exported content to be uploaded as xAPI/SCORM and run on a real LMS environment. This type of testing uncovered a major misunderstanding in the application, as it was found that the A-Frame framework injected a camera into its unembedded scenes at a different height as expected. This allowed for alterations to be made in the scene editor to ensure camera position was as expected when designing scenes. Smoke testing uncovered various other bugs throughout the application; however, time restrictions did not permit these to be fixed. The documented bugs can be found in section 4.5

## Usability Testing

The system usability scale (SUS) is a simple method to evaluate the usability of a system. Once a user tests your system, they are presented with a 10-question survey on their experience. Each question is rated with a 5-point scale, with 1 being strongly disagree and 5 being strongly agree. Once the results are calculated, the closer the average result is to 100, the more usable the system is. Once the results are compiled and calculated, if a system is rated above 68 then it is above the average rating [52]. This rating system was employed due to its easy methodology and it offers simple questions for participants. If not restricted by limitations, this study would have provided a more in depth subjective testing method such as the NASA Task Load Index where users are evaluated on the mental demands, physical demands, temporal demands, own performance, effort and frustration [53]. However, the system usability scale is still a popular option when testing the usability of applications.

As the application is intended to be used by inexperienced users, it was important that usability testing involved suitable testers. Because of this, before participants were considered for testing, they were asked on whether they judged themselves to be experienced with computers, modern web applications or design software’s. All testers who considered themselves to be below the level of skilled were permitted to take part in testing. The first round of testing contained 6 participants, whereas, due to prior commitments and time restrictions, the second only had 4.

To test the usability, a basic mock scene was developed which employed various assets that would be expected in an eLearning VR scene, Appendix D. The participants were asked to replicate the scene as best they could. Once complete, the users completed their SUS tests and feedback forms. As discussed in section 3.3.1, usability testing was staged during the final two iterations of the design.

## Results

Unit testing results confirmed that most of the validation and functionality in the service layer was functioning correclty. As the service layer’s main purpose was validation, classes within the service layer were configured to ensure that if an object was not retrieved correctly then it should return null rather than an object of null(s).

The only faulting test case in the application found during unit tests was the VRBackgroundService’s GetBackgroundImageWithSceneId(int id) method. The method should have been configured to return null if a background image byte array was not found; however, the method was returning a byte array of null values. Without unit testing, this method may have caused an exception during deployment.

After gathering the first round of participants usability results, the author calculated the scores. To calculate scores using the SUS, all odd numbered questions results, being 1-5, subtract one from the participants response; all even numbered questions subtract the user response from 5. Once the participants results are summed, the total value is multiplied by 2.5. This provides a score of between 0 and 100, rather than 0 to 40 [52].

The first round of user evaluation resulted with varying results between 50 and 70. As usability is subjective, this gap can be expected due to different skill levels. The total average was 60. As this was below average, some serious redesign had to be considered. As the SUS is a quick means of testing usability and doesn’t provide a deep insight into the user’s considerations, users were requested to complete a feedback form discussing what worked and what didn’t work as much. After reviewing the feedback forms, it was clear that the adding and editing of assets hindered the usability of the scene editor heavily. This was due to the user having too wide a range of buttons to select from when adding new assets or editing objects. The feedback also commented on the difficulty when rotating assets. This introduced the implementation of the above mentioned, three.js library for controlling the rotation of the assets through mirroring rotation of a simple object.

Once the final design iteration was complete, the second usability evaluation was initiated. Due to prior commitments and time limitations, only two of the original testers could participate; however, an additional two testers were welcomed to test the application. The changes to be noted between usability evaluations are as follows:

1. Buttons containing text on asset editor boxes now use icons
2. Add new asset buttons removed and a central add button provides dropdown of all available assets.
3. Add new scene button repositioned to existing scenes partial view
4. Icon for identification of asset in asset editor carousel.

The second round of usability evaluation resulted with an average score of 77.5. This dramatic increase offered proof that the final design iteration provided a far more usable design. The full results of both stages of user evaluation can be found in the file ‘User Evaluation Results’. As two users participated in both the first and the second draft of user testing, their first and second testing results were compared and are as follow:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Evaluation One | | Evaluation Two | |
| SUS Question | User 1 | User 2 | User 1 | User 2 |
| I think that I would like to use this system frequently | 1 | 2 | 1 | 2 |
| I found the system unnecessarily complex | 3 | 2 | 1 | 1 |
| I thought the system was easy to use | 2 | 3 | 4 | 4 |
| I think that I would need the support of a technical person to be able to use this system | 3 | 2 | 2 | 1 |
| I found the various functions in this system were well integrated | 3 | 2 | 4 | 3 |
| I thought there was too much inconsistency in this system | 2 | 3 | 2 | 2 |
| I would imagine that most people would learn to use this system very quickly | 3 | 3 | 4 | 3 |
| I found the system very cumbersome to use | 2 | 2 | 2 | 2 |
| I felt very confident using the system | 3 | 3 | 3 | 4 |
| I needed to learn a lot of things before I could get going with this system | 2 | 2 | 1 | 1 |

The above table clearly displays significant improvements in terms of usability. With the most notable improvements being user 1’s “I found this system unnecessarily complex” response improving from an unresolved 3 to a strongly disagree and “I thought the system was easy to use” improving from a slightly disagree to slightly agree. Improvements regarding learning the system were disregarded due to the participants already having experienced the previous version of the system. Additionally, both users provided feedback stating that the inconsistency within the application was due to the constant refreshing of the page. Through verbal feedback, it was discovered that they found A-Frame’s forced reload of the framework to be the inconsistency. As this is outside of this applications control, this could not be remedied.

## Known Bugs

|  |  |  |
| --- | --- | --- |
| Bug | Cause | Threat |
| Editing Question Response – stores in database but doesn’t update scene dynamically | Due to response’s being stored separately in the database, their ID’s are unattainable just now without error. | Minor – still updates database when changed and saved. But page must be refreshed. |
| HotSpot list of objects not updating dynamically | Order of updating partial views with jQuery GETs. | Minor – Reloading the page updates the list. |
| Media Assets | The app has been designed for deployment. Video & Audio files are not stored in the solution but are instead stored in the C:/. This allows the location to be updated to suitable storage when deployed. | No Threat as it was deliberate. |
| Object Filter – doesn’t work after adding or removing assets. | Order of updating partial views with jQuery. | Minor – Reloading the page fixes problem. |
| Adding new scene when no assets doesn’t dismiss modal properly | A plugin used early in the project required an earlier version of jQuery. This may be serving other areas of the project and cannot be removed. | Major – can cause serious errors with page functionality. |

# Future Work

As the project developed, the application and the background of study were more firmly understood. This led to the development of additional user stories which were added to the product backlog. However, due to time constraints they were not able to be implemented. The following will discuss the potential additional features of the system both already contained in the product backlog and further ideas. Bug fixes will not be included.

## Scene and Asset Duplication

As found during the background research, one of the primary supporting methods for retaining information is spaced repetition [9]. To incorporate spaced repetition into the application, a duplicate scene and duplicate asset button could be provided. This would allow users to duplicate their assets and update the details accordingly; or, it would have allowed them to duplicate an entire scene along with all assets contained within it and allowed for updates accordingly. The back-end functionality for this would not be difficult as it would just be duplicating a tuple’s attributes from the database; however, the front-end would need to be understood more thoroughly.

## Background Video

A proposition for future work found early in the project was the requirement of 360-degree video. This would have been a great advantage to the project; however, before incorporating, a solid understanding of A-Frame and scene timing would need to be understood. As functionality would need to be included for various anticipated actions such as, “Would a question be displayed at the end of the scene?”, “Would the user automatically go to the next scene?”, “Would the application end?”. These are all conversations which would ultimately lead to a suitable implementation which would best serve the application’s core functionality.

## Rearrange Scene Order

For a vast increase of usability, the proposition of a ‘click & drag’ implementation for rearranging scenes in the scene editor would allow users to select a scene and rearrange it so the course follows a specific order. Though this has been countered by the current system employing HotSpot’s to link scenes together, the proposed scene order would provide a much cleaner and more sustainable solution.

# Conclusion

This chapter will discuss the final state of the project, stating the achievements, what wasn’t achieved, what could be improved and what could have been done differently.

## Final State

The project objective was to develop a virtual reality eLearning authoring tool to offer virtual reality eLearning content design capabilities to inexpert users. Below, table 1 displays the final burndown chart of the must have user story points which includes the user stories added throughout development. As can be seen, the user story points are far greater than estimated in the requirements document.

Table 1 – Actual Vs Planned Burndown Chart

Though the initial estimated user stories may have been relevant, their user story points were not accurate. This is believed to be due to the authors lack of understanding surrounding each individual user story and the introduction of unfamiliar technologies. Had the user stories in the requirements document been more thoroughly analyzed and broken down, the author may have realized the scope of the project was far too great for the planned 7-week iteration and the project may have been refactored to ensure the absolute necessities were thoroughly complete. Though the time planning and progress monitoring of the application was firm, as the requirements were updated, time limitations and unanticipated framework alterations restricted the project from completing all user stories. The remaining incomplete stories were changing an audio file after an audio asset had been added to the scene and updating a hotspots event after it had been added to the scene. It is understood that this project was ambitious for the 7-week iteration and, though it did not completely achieve what it set out to, it provides an incredible foundation and understanding for a future solution. The strong foundation is set through following good software development practices, maintaining well documented code, ensuring database integrity and creating a well-structured system architecture which can be easily built upon for future development.

In hindsight, the additional user stories should not have been included. This led to an unachievable product backlog which was the downfall on the project’s management. If this project was to be done differently, the core functionality of the scene editor would have been the main priority as it should have been. The unnecessary time allocation spent on remedial pages such as about us, contact us and help could have been better spent on fixing the known bugs. However, as the goal of the project was to deploy a functioning solution; these pages were thought to be necessary at the time. Furthermore, developing such an intensive project in an unfamiliar programming language, along with various other new technologies such a A-Frame and jQuery, did not start the project off well. Though the learning curve regarding C# was not huge, the accumulated learning curve of all new technologies contributed to a poorer understanding of the desired solution.

After overcoming the various pitfalls throughout the project, this project has still concluded with a functional eLearning authoring tool which has been proven to offer incredible usability to inexpert users. From following an agile methodology alongside a user-centered design, the author has created a highly usable web application. After a long three weeks of thorough testing, it was confirmed that the proposed solution met all non-functional requirements. Though the project had its drawbacks, the author is incredibly excited about the future of the application. Adjustments will be made, but ReaLearn will be deployed in the near feature.

References

|  |  |
| --- | --- |
| [1] | J. Kiruthika and S. Khaddaj, “Impact and Challenges of Using of Virtual Reality & Artificial Intelligence in Businesses,” in *2017 16th International Symposium on Distributed Computing and Applications to Business, Engineering and Science (DCABES)*, Anyang, IEEE, 2017, pp. 165 - 168. |
| [2] | S. Greengard, “The Many Shaped and Forms of AR and VR,” in *Virtual Reality*, Cambridge, MA, The MIT Press 2019, 2019, p. 54. |
| [3] | V. R. Society, “What is virtual reality,” Virtual Reality Society, 2017. [Online]. Available: https://www.vrs.org.uk/virtual-reality/what-is-virtual-reality.html. [Accessed 20 August 2019]. |
| [4] | P. Nicholson, “A History of E-Learning,” in *Computers and Education*, Dordrecht, Springer, Dordrecht, 20, pp. 1 - 11. |
| [5] | J. Herrington and R. R. T. Oliver, “Patterns of engagement in authentic online learning environments,” *Australian Journal of Educational Technology,* vol. 19, no. 1, pp. 59 - 71, 2003. |
| [6] | P.-S. Daniel Chen, A. Lambert and K. Guidry, “Engaging online learners: The impact of Web-based learning technology on college student engagement,” *Computers & Education,* vol. 54, no. 4, pp. 1222 - 1232, 2010. |
| [7] | C. Rabe-Hemp, S. Woollen and G. S. Humiston, “A COMPARATIVE ANALYSIS OF STUDENT ENGAGEMENT, LEARNING, AND SATISFACTION IN LECTURE HALL AND ONLINE LEARNING SETTINGS,” *Quartetly Review of Distance Education,* vol. 10, no. 2, pp. 207 - 218, 250-252, 2009. |
| [8] | M.-T. Wang and J. Degol, “Staying Enaged: Knowledge and Research Needs in Student Engagement,” *Child Development Perspectives,* vol. 8, no. 3, pp. 137-143, 2014. |
| [9] | W. Thalheimer, “Does eLearning Work? What the Scientific Research Says!,” Work-Learning Research, Inc., Massauchusetts, 2017. |
| [10] | P. Cipresso, I. A. Chicci Giglioli, M. Alcaniz Raya and R. Giuseppe, “The Past, Present, and Future of Virtual and Augmented Reality Research: A Network and Cluster Analysis of the Literature,” *Frontiers in Psychology,* vol. 9, no. 2086, 2018. |
| [11] | Expert Panel, Forbes Technology Council, “13 Of The Best Virtual Reality Uses in Industry To Date,” Forbes, 23 January 2019. [Online]. Available: https://www.forbes.com/sites/forbestechcouncil/2019/01/23/13-of-the-best-virtual-reality-uses-in-industry-to-date/#6e37ea7d220d. [Accessed 20 08 2019]. |
| [12] | Mozilla, “WebVR - Virtual Reality for the Web,” Mozilla, 23 03 2019. [Online]. Available: https://developer.mozilla.org/en-US/docs/Games/Techniques/3D\_on\_the\_web/WebVR. [Accessed 23 08 2019]. |
| [13] | L. Brown, E. Murphy and V. Wade, “Corporate eLearning: Human resource development implications for large and small organizations,” *Human Resource Development International,* vol. 9, no. 3, pp. 415-427, 2006. |
| [14] | V. Arkorful and N. Abaidoo, “The role of e-learning, advantages and disadvantages of its adoption in higher education,” *International Journal of Instructional Technology and Distance Learning,* vol. 12, no. 1, pp. 29 - 42, 2015. |
| [15] | P. Chen, A. Boenink and K. Guidry, “Engaging Online Learners: The Impact of Web-Based Learning Technology on Colleg Student Engagement,” *Computers & Education,* vol. 54, no. 4, pp. 1222-1232, 2010. |
| [16] | L. Taylor and J. Parsons, “Improving Student Engagement,” *Current Issues in Education,* vol. 14, no. 1, 2011. |
| [17] | R.-M. Conrad and J. A. Donaldson, Engaging the Online Learner: Activities and Resources for Creative Instruction, Updated Edition ed., California: Jossey-Bass, 2011. |
| [18] | E. Dale, in *Audio-Visual Methods in Teaching*, 3rd ed., R. &. W. Holt, Ed., New York, The Dryden Press, 1946, p. 108. |
| [19] | C. Pappas, “What is A Learning Management System? LMS Basic Functions And Features You Must Know,” eLearning Industry, 3 December 2017. [Online]. Available: https://elearningindustry.com/what-is-an-lms-learning-management-system-basic-functions-features. [Accessed 24 08 2019]. |
| [20] | A. H. Omer, “The 4 Types Of Compatibility Standards In eLearning,” eLearningIndustry, 19 November 2016. [Online]. Available: https://elearningindustry.com/compatibility-standards-in-elearning-4-types. [Accessed 22 August 2019]. |
| [21] | xAPI, “xAPI Overview,” Rustici Software, 2019. [Online]. Available: https://xapi.com/overview/. [Accessed 24 08 2019]. |
| [22] | uptale, “Uptale - Discover the Immersive Learning Platform in 1 minute,” uptale, 15 October 2018. [Online]. Available: https://www.youtube.com/watch?v=kA8D9pjblhc. [Accessed 29 08 2019]. |
| [23] | eLearning Industry, “Uptale Reviews,” eLearning Industry, June 2019. [Online]. Available: https://elearningindustry.com/directory/elearning-software/uptale/reviews. [Accessed 21 08 2019]. |
| [24] | WikiPedia, “Greenfield Project,” WikiPedia, 17 September 2018. [Online]. Available: https://en.wikipedia.org/wiki/Greenfield\_project. [Accessed 28 08 2019]. |
| [25] | Agile Alliance, “Agile 101,” Agile Alliance, 2019. [Online]. Available: https://www.agilealliance.org/agile101/. [Accessed 30 08 2019]. |
| [26] | S. Lewis, “waterfall model,” Search Software Quality, February 2019. [Online]. Available: https://searchsoftwarequality.techtarget.com/definition/waterfall-model. [Accessed 30 August 2019]. |
| [27] | A. Williams, “Writing Maintainable Code,” Just Software SOlutions, 2019. [Online]. Available: https://www.justsoftwaresolutions.co.uk/articles/maintainable\_code.html. [Accessed 30 August 2019]. |
| [28] | Packt Editorial Staff, “What is a multi layered software architecture,” Packt, 17 May 2018. [Online]. Available: https://hub.packtpub.com/what-is-multi-layered-software-architecture/. [Accessed 30 August 2019]. |
| [29] | D. Murli, “What are the advantages and disadvantages of a layered architecture,” 3 June 2012. [Online]. Available: http://murlid05.blogspot.com/2012/06/what-are-advantages-and-disadvantages.html. [Accessed 30 August 2019]. |
| [30] | Microsoft, “The .NET Platform,” Microsoft, 2019. [Online]. Available: https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet. [Accessed 31 August 2019]. |
| [31] | T. C. Lethrbridge and R. Laganiere, “Architecting and Designing Software,” in *Object-Oriented Software Engineering Second Edition*, Berkshire, McGraw-Hill Education, 2005, pp. 355 - 356. |
| [32] | Dot Net Development, “Six Benefits of Using MVC Model for Effective Web Application Development,” 28 April 2016. [Online]. Available: https://www.brainvire.com/six-benefits-of-using-mvc-model-for-effective-web-application-development/. [Accessed 31 08 2019]. |
| [33] | A. Vodovnik, “Repository and Service Pattern in a Multilayered Architecture,” 26 August 2015. [Online]. Available: https://www.vodovnik.com/2015/08/26/repository-and-services-pattern-in-a-multilayered-architecture/. [Accessed 2015 August 31]. |
| [34] | Microsoft, “Validating with a Service Layer (C#),” Microsoft, 03 02 2009. [Online]. Available: https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions-1/models-data/validating-with-a-service-layer-cs. [Accessed 27 08 2019]. |
| [35] | L. McCulloch, “The Importance of Data Validation,” deltaDNA, 12 September 2016. [Online]. Available: https://deltadna.com/blog/importance-data-validation/. [Accessed 31 August 2019]. |
| [36] | S. S. Shekhawat, “CRUD operation using the generic repository pattern,” codeproject, 24 December 2014. [Online]. Available: https://www.codeproject.com/Articles/838097/CRUD-Operations-Using-the-Generic-Repository-Pat. [Accessed 02 September 2019]. |
| [37] | T. Janssen, “Design Patterns Explained – Dependency Injection with Code Examples,” Stackify, 2019 June 2018. [Online]. Available: https://stackify.com/dependency-injection/. [Accessed 30 August 2019]. |
| [38] | B. Karia, “A quick intro to Dependency Injection: what it is, and when to use it,” freecodecamp, 18 October 2018. [Online]. Available: https://www.freecodecamp.org/news/a-quick-intro-to-dependency-injection-what-it-is-and-when-to-use-it-7578c84fa88f/. [Accessed 31 August 2019]. |
| [39] | J. Oliveira, “Database or Code First Approach - Developer's Vision,” 02 September 2016. [Online]. Available: https://www.linkedin.com/pulse/database-code-first-approach-developers-vision-jader-oliveira. [Accessed 28 August 2019]. |
| [40] | R. Elmasri and S. Navathe, “Chapter 15 Basics of Functional Dependencies and Normalization for Relational Databases,” in *Fundamentals of Database System: 6th Edition*, Arlington, Perason Education, Inc., 2011, pp. 519 - 528. |
| [41] | E. R. and N. S., “Chapter 15 Basics of Functional Dependencies and Normalization for Relational Databases,” in *Fundamentals of Database Systems: 6th Edition*, Arlington, Pearson Education, Inc., 2011, pp. 507 - 509. |
| [42] | R. Dam and T. Siang, “Personas - A Simple Introduction,” Interaction Design Foundation, 15 August 2019. [Online]. Available: https://www.interaction-design.org/literature/article/personas-why-and-how-you-should-use-them. [Accessed 04 September 2019]. |
| [43] | J. Gullisken, B. Goransson, I. Boivie, S. Blomkvist, J. Persson and A. Cjander, “Key principles for user-centred systems design,” *Behaviour & Information Technology,* vol. 22, no. 6, pp. 397 - 409, 2003. |
| [44] | Eric, “Mouse Drag Rotate Cube (3D WebGL),” jsFiddle, 12 September 2016. [Online]. Available: https://jsfiddle.net/MadLittleMods/n6u6asza/. [Accessed 12 08 2019]. |
| [45] | Web Development Tutorial, “Understanding Partial View in ASP.NET MVC,” Web Development Help, 16 September 2014. [Online]. Available: http://www.webdevelopmenthelp.net/2014/09/partial-view-in-asp-net-mvc.html. [Accessed 02 September 2019]. |
| [46] | D. Esposito, “Tips and Tricks about Razor Partial Views,” Red Gate Software, 23 November 2015. [Online]. Available: https://www.red-gate.com/simple-talk/dotnet/asp-net/tips-and-tricks-about-razor-partial-views/. [Accessed 1 September 2019]. |
| [47] | Microsoft, “Code First Migrations,” Microsoft, 23 October 2016. [Online]. Available: https://docs.microsoft.com/en-us/ef/ef6/modeling/code-first/migrations/. [Accessed 02 09 2019]. |
| [48] | I. Zakrevsky, “Design of Service Layer,” 17 July 2017. [Online]. Available: https://emacsway.github.io/en/service-layer/. [Accessed 30 August 2019]. |
| [49] | P.-E. Bergman, “Repository Design Pattern,” 20 April 2017. [Online]. Available: https://medium.com/@pererikbergman/repository-design-pattern-e28c0f3e4a30. [Accessed 03 September 2019]. |
| [50] | Rustici Software's, “xAPI solved and explained,” Rustici, [Online]. Available: https://xapi.com/?\_ga=2.198603494.859206863.1567721892-1999115137.1567624975. [Accessed 4 September 2019]. |
| [51] | SmartBear, “What is Unit Testing,” SmartBear, 2019. [Online]. Available: https://smartbear.com/learn/automated-testing/what-is-unit-testing/. [Accessed 02 September 2019]. |
| [52] | J. Sauro, “MEASURING USABILITY WITH THE SYSTEM USABILITY SCALE (SUS),” MeasuringU, 2 February 2011. [Online]. Available: https://measuringu.com/sus/. [Accessed 31 August 2019]. |
| [53] | Human Performance Research Group, “NASA TASK LOAD INDEX (TLX),” Ames Research Center, California, 1986. |
| [54] | J. Hamari, J. Koivisto and H. Sarsa, “Does Gamification Work? — A Literature Review of Empirical Studies on Gamification,” in *47th Hawaii International Conference on System Services*, Hawaii, 2014. |
| [55] | H. Shah, “Why Α Learning Management System Is Useful,” eLearningIndustry, 4 June 2015. [Online]. Available: https://elearningindustry.com/learning-management-system-useful. [Accessed 22 August 2019]. |

###### User Stories

The following is the list of final user stories:

General User

1. As a user, I must be able to register my account.
2. As a user, I must be able to log in securely.
3. As a user, I must be able to log out securely.
4. As a user, I would like to be able to contact ReaLearn.
5. As a user, I would like to be able to retrieve my password if I forget it.

Standard User

1. As a standard user, I must be able to edit my own details.
2. As a standard user, I must be able to view courses that have been assigned to me.

Admin User

1. As an admin user, I must be able to add standard users.
2. As an admin user, I must be able to view all my companies’ courses.
3. As an admin user, I must be able to add courses.
4. As an admin user, I must be able to delete courses.
5. As an admin user, I must be able to delete standard users.
6. As an admin user, I must be able to edit courses.
7. As an admin user, I must be able to view all my companies’ standard users.
8. As an admin user, I must be able to edit standard users’ details.
9. As an admin user, I must be able to export courses to xAPI.

Scene Editor

1. As a scene designer, I must be able to view the existing scenes within my course.
2. As a scene designer, I must be able to view all assets edit properties in the scene.
3. As a scene designer, I must be able to add a new scene.
4. As a scene designer, I must be able to add hotspot asset.
5. As a scene designer, I must be able to add an image asset.
6. As a scene designer, I must be able to add an audio asset.
7. As a scene designer, I must be able to add a video asset.
8. As a scene designer, I must be able to add an image background.
9. As a scene designer, I must be able to add text object to scene.
10. As a scene designer, I must be able to add question.
11. As a scene designer, I must be able to edit objects rotation within the scene.
12. As a scene designer, I must be able to edit any assets position within the scene.
13. As a scene designer, I must be able to edit objects scale within the scene.
14. As a scene designer, I must be able to change background colour.
15. As a scene designer, I must be able to change font colour.
16. As a scene designer, I must be able to update text in a text object.
17. As a scene designer, I must be able to save changes to individual objects.
18. As a scene designer, I must be able to change video and image object colour.
19. As a scene designer, I must be able to change a hotspot’s linked object.
20. As a scene designer, I must be able to change a video in a video object.
21. As a scene designer, I must be able to change an image in an image object.
22. As a scene designer, I must be able to change audio volume.
23. As a scene designer, I must be able to change background image.
24. As a scene designer, I must be able to change the audio in an audio object.
25. As a scene designer, I must be able to change the event of a hotspot.
26. As a scene designer, I must be able to edit the question and responses of a question object.
27. As a scene designer, I must be able to delete any object asset.
28. As a scene designer, I must be able to delete a scene.
29. As a scene designer, I would like to be able to filter the visible objects in the scene.
30. As a scene designer, I must be able to preview a scene.
31. As a scene designer, I must be able to export my course from within the scene editor.
32. As a scene designer, I would like to be able to upload a background video.
33. As a scene designer, I would like to be able to save all of my changes.
34. As a scene designer, I would like to be able to discard my changes.
35. As a scene designer, I would like to be able to display questions at specific times during videos.
36. As a scene designer, I would like to be able to duplicate objects.
37. As a scene designer, I would like to be able to duplicate objects.
38. As a scene designer, I would like to be able to insert a scene at any place in the course.
39. As a scene designer, I would like to be able to preview a full course.
40. As a scene designer, I would like to be able to reorder scenes.
41. As a scene designer, I would like to be able to upload images from a URL.
42. As a scene designer, I would like to be able to upload videos from a URL.

Content Viewers

1. As a content viewer, I must be able to stop media by interacting with a hotspot.
2. As a content viewer, I must be able to change scene by interacting with a hotspot.
3. As a content viewer, I must be able to hide an object after interacting with a hotspot.
4. As a content viewer, I must be able to play media by interacting with a hotspot.
5. As a content viewer, I must be able to see an object after interacting with a hotspot.

System

1. As the system, I must be able to export the selected course along with all necessary files as xAPI/SCORM format.

###### - MoSCoW Statements

Must Have

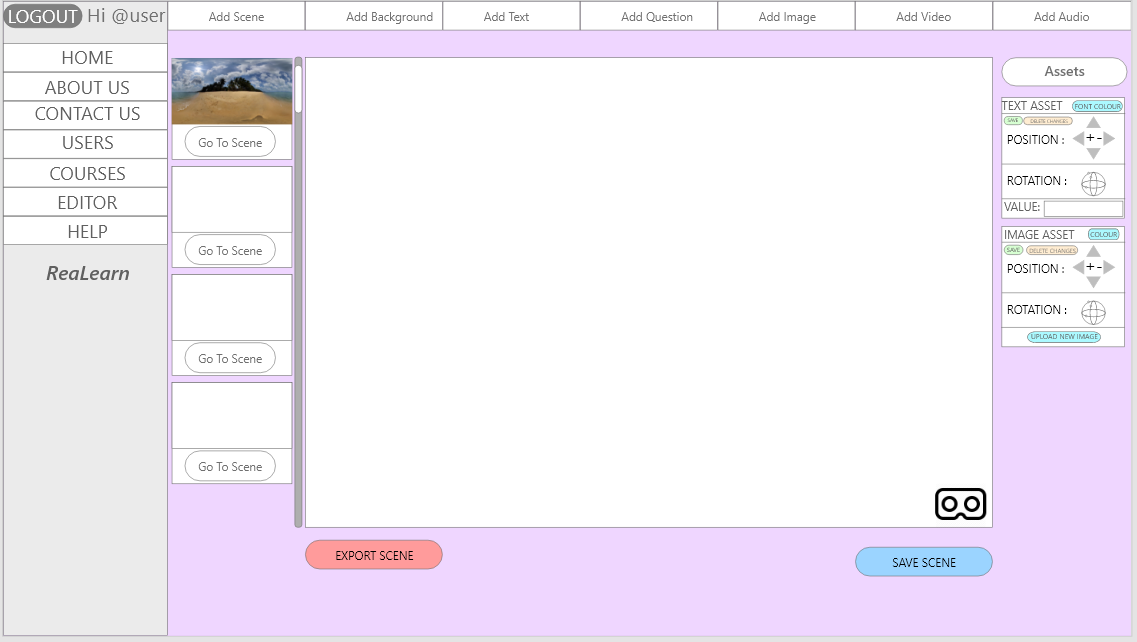
1. As a user, I must be able to register my account.
2. As a user, I must be able to log in securely.
3. As a user, I must be able to log out securely.
4. As a standard user, I must be able to edit my own details.
5. As a standard user, I must be able to view courses that have been assigned to me.
6. As an admin user, I must be able to add standard users.
7. As an admin user, I must be able to view all my companies’ courses.
8. As an admin user, I must be able to add courses.
9. As an admin user, I must be able to delete courses.
10. As an admin user, I must be able to delete standard users.
11. As an admin user, I must be able to edit courses.
12. As an admin user, I must be able to view all my companies’ standard users.
13. As an admin user, I must be able to edit standard users’ details.
14. As an admin user, I must be able to export courses to xAPI.
15. As a scene designer, I must be able to view the existing scenes within my course.
16. As a scene designer, I must be able to view all assets edit properties in the scene.
17. As a scene designer, I must be able to add a new scene.
18. As a scene designer, I must be able to add hotspot asset.
19. As a scene designer, I must be able to add an image asset.
20. As a scene designer, I must be able to add an audio asset.
21. As a scene designer, I must be able to add a video asset.
22. As a scene designer, I must be able to add an image background.
23. As a scene designer, I must be able to add text object to scene.
24. As a scene designer, I must be able to add question.
25. As a scene designer, I must be able to edit objects rotation within the scene.
26. As a scene designer, I must be able to edit any assets position within the scene.
27. As a scene designer, I must be able to edit objects scale within the scene.
28. As a scene designer, I must be able to change background colour.
29. As a scene designer, I must be able to change font colour.
30. As a scene designer, I must be able to change an assets size.
31. As a scene designer, I must be able to update text in a text object.
32. As a scene designer, I must be able to save changes to individual objects.
33. As a scene designer, I must be able to change video and image object colour.
34. As a scene designer, I must be able to change a hotspot’s linked object.
35. As a scene designer, I must be able to change a video in a video object.
36. As a scene designer, I must be able to change an image in an image object.
37. As a scene designer, I must be able to change audio volume.
38. As a scene designer, I must be able to change background image.
39. As a scene designer, I must be able to change the audio in an audio object.
40. As a scene designer, I must be able to change the event of a hotspot.
41. As a scene designer, I must be able to edit the question and responses of a question object.
42. As a scene designer, I must be able to delete any object asset.
43. As a scene designer, I must be able to delete a scene.
44. As a scene designer, I must be able to preview a scene.
45. As a scene designer, I must be able to export my course from within the scene editor.
46. As a content viewer, I must be able to stop media by interacting with a hotspot.
47. As a content viewer, I must be able to change scene by interacting with a hotspot.
48. As a content viewer, I must be able to hide an object after interacting with a hotspot.
49. As a content viewer, I must be able to interact with a hotspot by clicking my controller.
50. As a content viewer, I must be able to interact with a hotspot by viewing it.
51. As a content viewer, I must be able to play media by interacting with a hotspot.
52. As a content viewer, I must be able to see an object after interacting with a hotspot.
53. As the system, I must be able to export the selected course along with all necessary files as xAPI/SCORM format.

Would Have

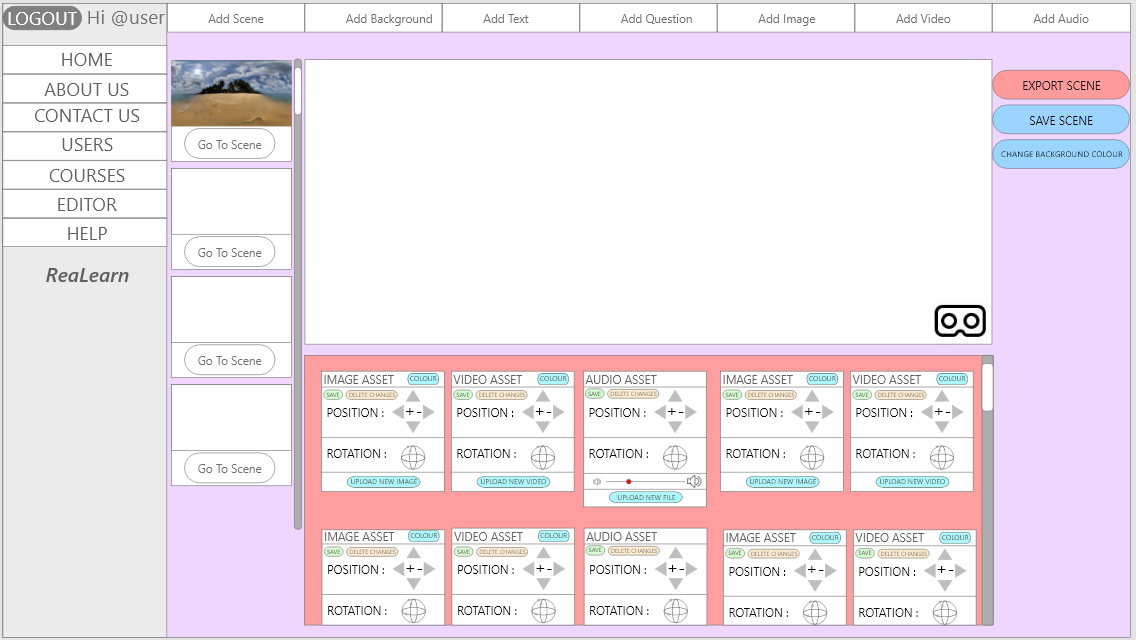
1. As a user, I would like to be able to contact ReaLearn.
2. As a user, I would like to be able to retrieve my password if I forget it.
3. As a scene designer, I would like to be able to filter the visible objects in the scene.
4. As a scene designer, I would like to be able to save all my changes.
5. As a scene designer, I would like to be able to discard my changes.
6. As a scene designer, I would like to be able to display questions at specific times during videos.
7. As a scene designer, I would like to be able to duplicate objects.
8. As a scene designer, I would like to be able to duplicate objects.
9. As a scene designer, I would like to be able to insert a scene at any place in the course.
10. As a scene designer, I would like to be able to preview a full course.
11. As a scene designer, I would like to be able to reorder scenes.
12. As a scene designer, I would like to be able to upload images from a URL.
13. As a scene designer, I would like to be able to upload videos from a URL.
14. As a scene designer, I would like to be able to upload a background video.

###### - Wire Frame Development

First Draft



Second Draft



Final Draft



###### - Mock Scene for Usability Testing

