**MEMBANGUN KONTEN LMS VR INTERACTIVE 180 VIDEO MENGGUNAKAN A-FRAME DAN SCORM**

**TEXTBOOK**

**JANUARI 2020**

**DAFTAR ISI**

**Cover1**

**Daftar Isi2**

**Pendahuluan3**

**Bab 1. Konversi Video 360 Menjadi Video 1804**

1. Stitching 360 Video4
2. Konversi ke 180 Video Menggunakan Software Adobe Premiere Pro5

**Bab 2. Membangun Konten LMS VR Interactive 180 Video8**

1. XAMPP Localhost8
2. Setup Project Baru8
3. Membuat Interaksi Tanya Jawab (Question and Answers/Option)11
4. Input Score ke LMS Menggunakan SCORM18
5. SCORM Packaging20
6. Publish SCORM Package ke LMS22

**Kesimpulan25**

**Glossary26**

**PENDAHULUAN**

A-Frame adalah sebuah web development framework yang khusus ditujukan untuk pengembangan WebVR dengan syntax mengikuti tag-tag HTML. Komponen-komponen framework ini juga dapat dimodifikasi sesuai kebutuhan melalui perintah-perintah kode Javascript. SCORM adalah standar pendistribusian paket e-Learning yang dapat digunakan untuk menampung berbagai spesifikasi dan standar untuk konten e-Learning berbasis web.

Dengan menggunakan A-Frame dan SCORM (Shareable Content Object Reference Model), kita dapat membuat konten WebVR yang dapat berinteraksi dengan LMS (Learning Management System), pada textbook ini kita akan membahas konten VR Interactive 180 Video. Pada dasarnya, kita akan membangun user interface; seperti button, text, dan 360 video viewer; menggunakan framework A-Frame. Kemudian untuk interaksi dengan LMS, kita menulis kode Javascript untuk men-setup value-value LMS menggunakan wrapper SCORM.

**Besertaan dengan textbook ini, disediakan beberapa file**, yaitu Adobe Premiere Pro 360 video ke 180 video template project, Interactive 180 Video WebVR template project, Interactive 180 Video WebVR example project, dan sample SCORM package. File-file tersebut juga dapat didownload melalui link berikut <https://github.com/NauvalNC/lms_interactive_180_video_vr>.

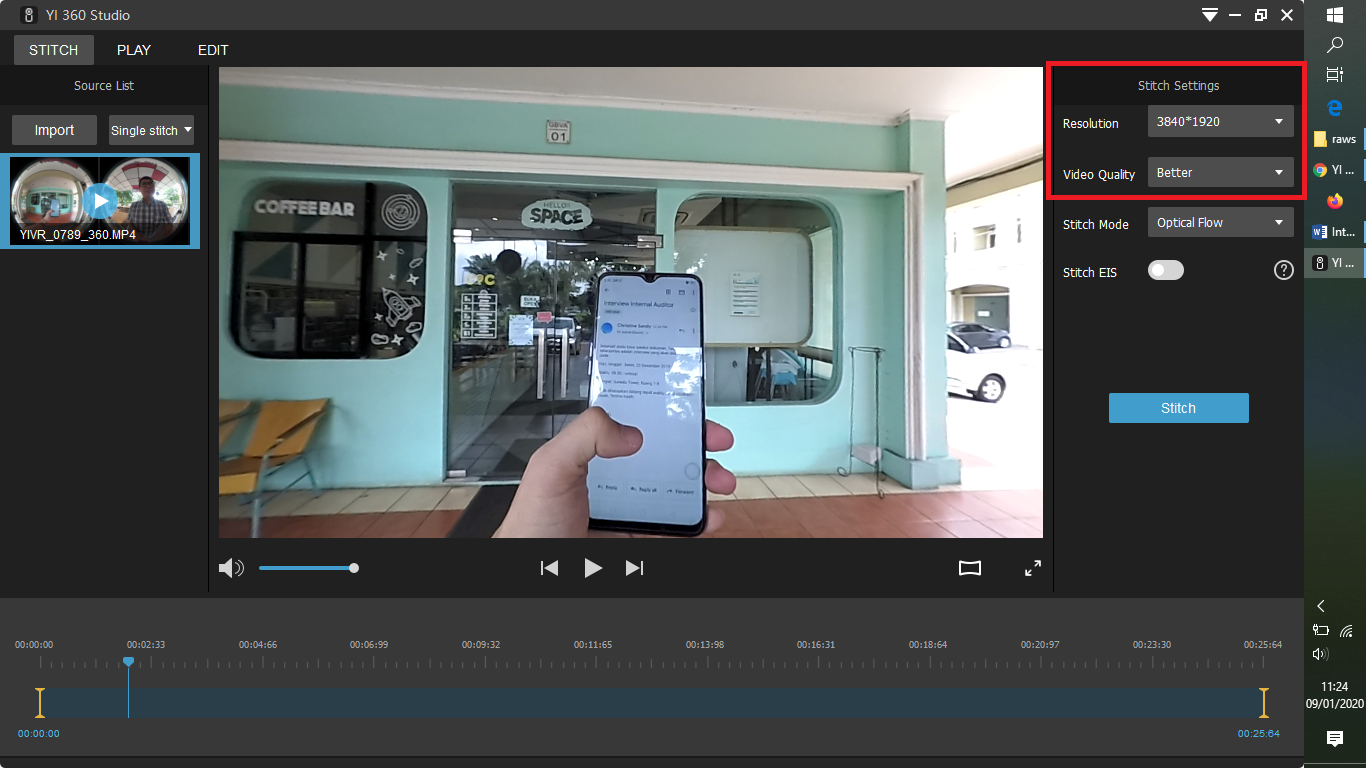
**BAB 1. KONVERSI VIDEO 360 MENJADI VIDEO 180**

Apabila kita menggunakan 360 camera tanpa fitur merekam video 180 derajat saja, maka kita perlu mengkonversinya. Berikut adalah workflow untuk proses konversi.

1. **Stitching 360 Video**

Ketika selesai merekam video 360, biasanya kita akan mendapat hasil video rekaman dimana ada 2 buah sphere serta video kita tampak seperti efek fish-eye. Ini karena tidak semua 360 camera memiliki fitur stitching otomatis. Stitching adalah proses dimana hasil video 360 yang berupa 2 sphere dikonvers menjadi video datar seperti pada umumnya. Produsen 360 camera, biasanya menyediakan software untuk proses stitching khusus untuk produk kamera mereka. Walaupun begitu, kita dapat menggunakan alternative software. Pada textbook ini, kita akan menggunakan software *YI 360 Studio*. Berikut langkah-langkahnya.

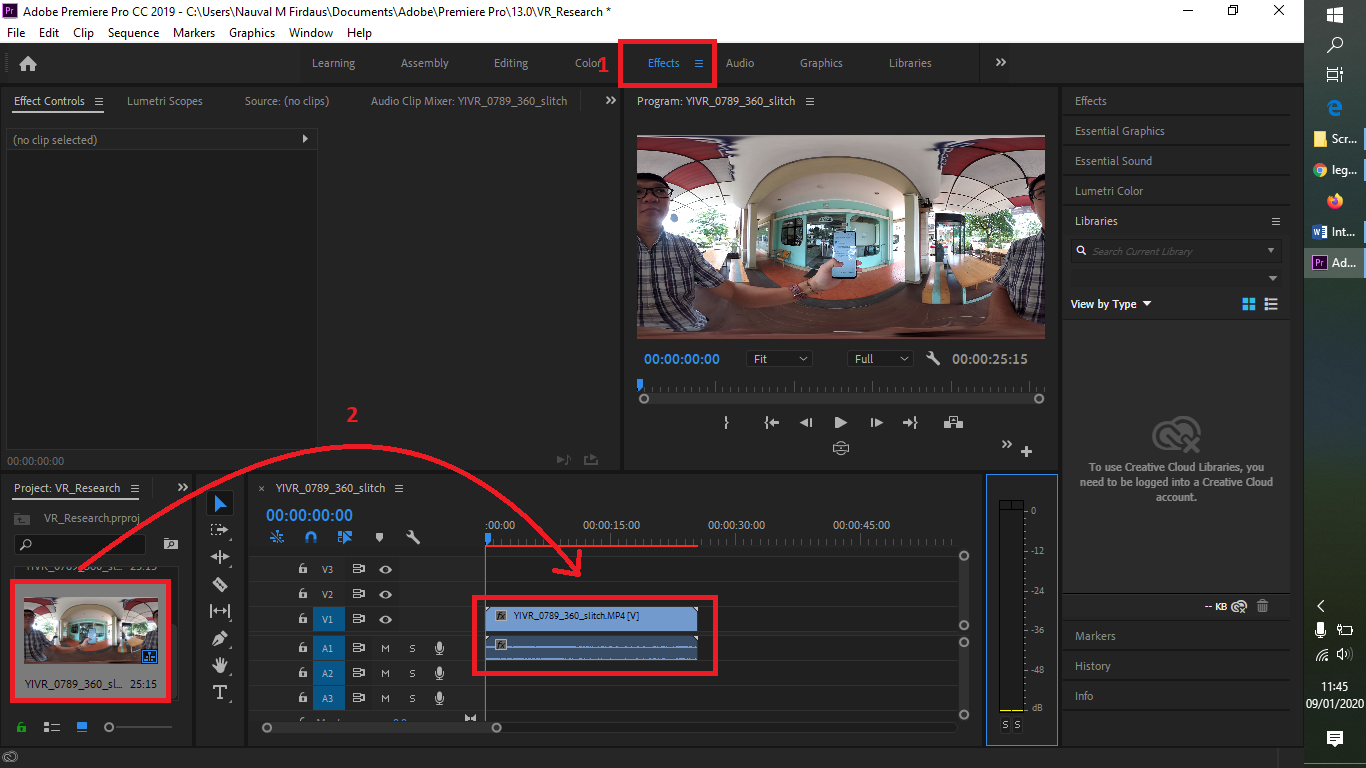
1. Download software *YI 360 Studio* online, atau menggunakan link berikut <https://www.yitechnology.com/yi-360-vr-camera-desktopsoftware>
2. Install dan jalankan softwarenya, kemudian import video yang diinginkan.
3. Lalu, pilih resolusi serta kualitas export yang diinginkan.



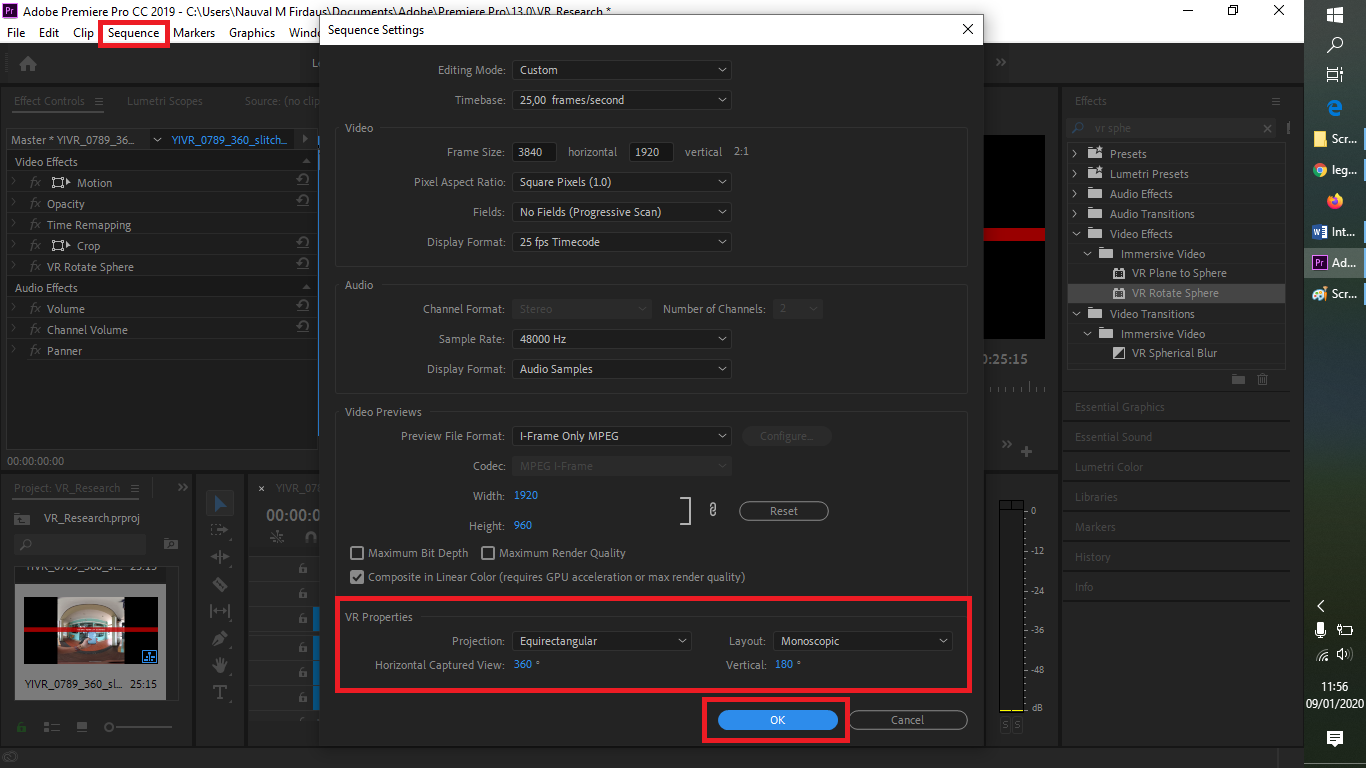
1. Kemudian, klik tombol “Stitch” untuk memulai proses stitching.
2. **Konversi ke 180 Video Menggunakan Software Adobe Premiere Pro**

Jika diperlukan, besertaan dengan textbook ini juga disediakan template project Adobe Premiere Pro. Berikut langkah-langkah konversinya.

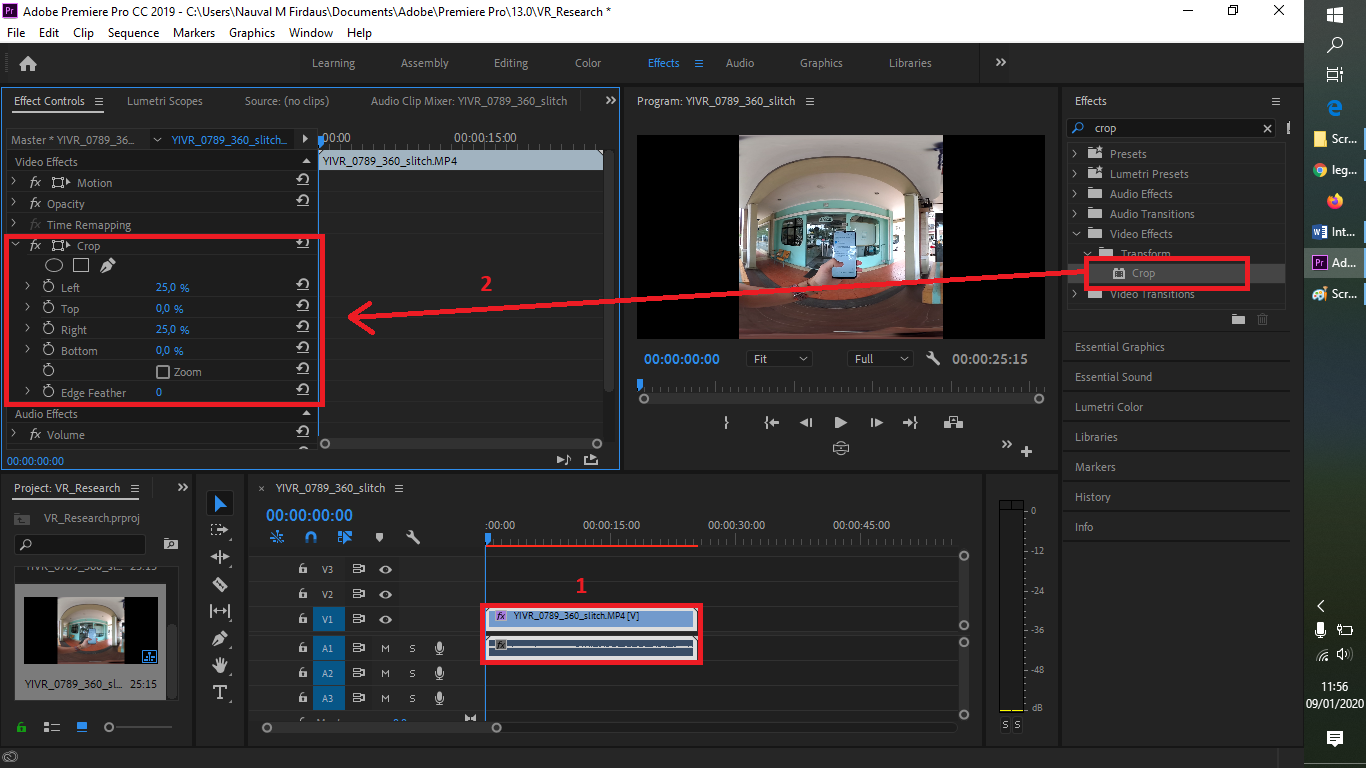
1. Buat project baru serta import video yang akan dikonvert. Ganti layout workflow menjadi layout Effect. Kemudian, drag and drop video ke panel Timeline untuk membuat Sequence baru.



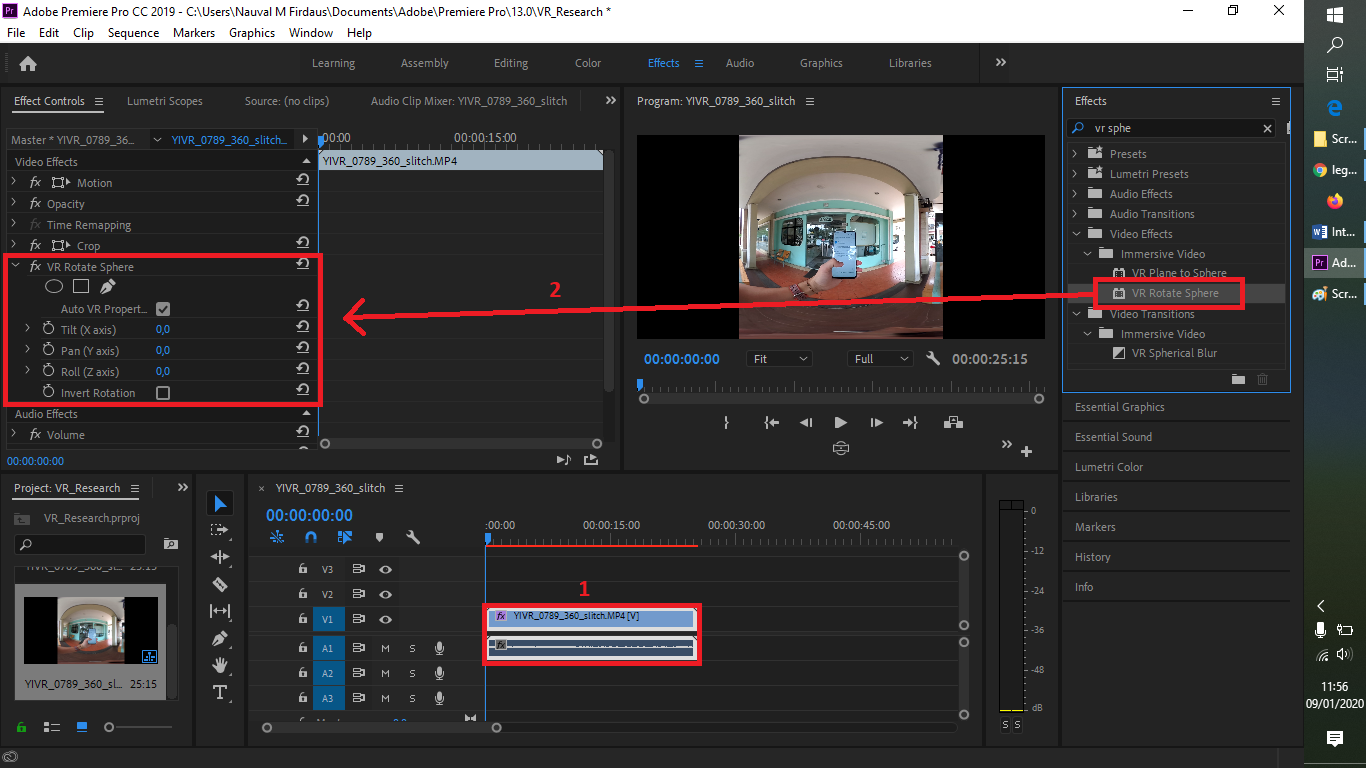
1. Pada toolbar, klik Sequence, kemudian Sequence Settings. Pada bagian VR Properties, karena kita menggunakan video 360 datar, set projection menjadi Equierectangular. Kemudian set property lainnya seperti gambar berikut.



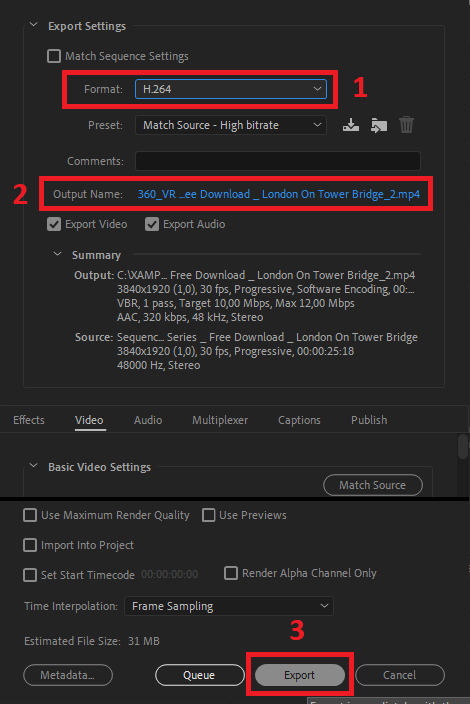
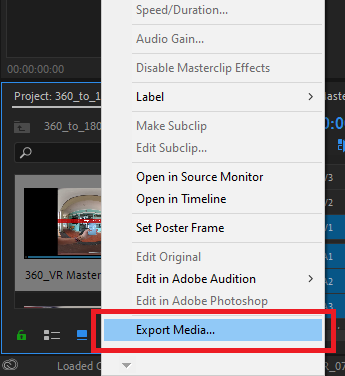
1. Pilih sequence video pada panel Timeline, kemudian pada panel Effects, cari efek *Crop*, kemudian drag and drop ke panel Effect Controls untuk menambahkan efek. Ubah property value Left dan Right menjadi 25%.



1. Pilih sequence video pada panel Timeline, kemudian pada panel Effects, cari efek *VR Rotate Sphere*, kemudian drag and drop ke panel Effect Controls untuk menambahkan efek. Jika kita mengubah value dari property Tilt, Pan, dan Roll, kita dapat memposisikan angle video yang kita inginkan.



1. Untuk mengexport video, pilih sequence yang telah kita buat, kemudian klik kanan, pilih “Export Media…”. Kemudian pilih format “H.264” dan ganti Output Name sesuai nama dan direktori yang diinginkan. Kemudian klik “Export”.



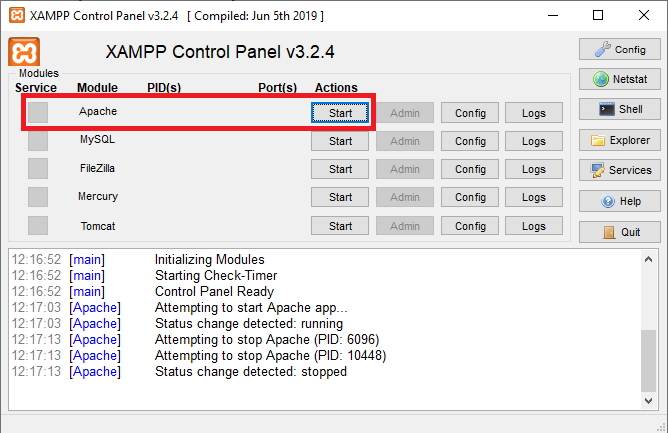
**BAB 2. MEMBANGUN KONTEN LMS VR INTERACTIVE 180 VIDEO**

Pada bagian ini, kita akan membangun konten interactive 180 video, dengan interaksi utama yaitu tanya jawab (question and answers). Perlu diperhatikan, pada textbook ini kita tidak memulai semuanya dari scratch, melainkan kita akan menggunakan template yang disediakan besertaan dengan textbook.

1. **XAMPP Localhost**

Untuk pengembangan WebVR menggunakan A-Frame, kita perlu men-setup hosting terlebih dahulu. Pada textbook ini, kita akan menggunakan localhost XAMPP. Untuk membuat localhost, lakukan langkah berikut.

1. Download XAMPP online, atau gunakan link berikut <https://www.apachefriends.org/index.html>
2. Install dan jalankan softwarenya, kemudian “Start” Apache module.



1. **Setup Project Baru**
2. Pada dasarnya, konten yang akan kita kembangkan adalah WebVR. Karena kita menggunakan A-Frame dan localhost, kita perlu membuat folder baru pada direktori XAMPP htdocs, secara default direktori terletak pada “C:\Program Files\XAMPP\htdocs\”. Pada direktori tersebut, buatlah folder baru sesuai nama project yang diinginkan. Kemudian, copy paste semua file dalam folder “Template Project” yang disediakan ke dalam folder project baru kita.
3. Kemudian, kita bisa memasukkan video kita pada folder “videos”. Pada textbook ini, video yang dimasukkan bernama “beach\_180.mp4” dan “london\_180.mp4”.
4. Buka file index.html pada text editor, pada textbook ini kita menggunakan Visual Studio Code. File ini adalah halaman web yang akan ditampilkan pada localhost kita dan course di LMS nanti. Pada file ini, kita akan menemukan tag <a-scene>, didalam tag inilah semua komponen user interface berada. Dalam tag <a-scene> terdapat tag <a-assets> dimana semua assets seperti video, kita hubungkan dengan index.html. Untuk menghubungkan video kita, dibawah komentar “Video Sky Sources” ikuti syntax line pertama berikut.

<!-- Video Sky Sources -->

<video id="id\_video" src="videos/nama\_video.mp4"></video>

<video id="beach\_180" src="videos/beach\_180.mp4"></video>

<video id="london\_180" src="videos/london\_180.mp4"></video>

* **id** : id video yang digunakan sebagai pengganti source/alamat asli video.
* **src** : source/alamat video kita, karena kita memasukkan video dalam folder videos, maka format alamatnya adalah “videos/nama\_video.mp4”.

1. Pilih id video yang ingin pertama ditampilkan, kemudian didalam tag <a-scene> dibawah komen “A Frame Video Sphere”, ganti value src menjadi id video yang dipilih dengan format “#id\_video”. Pada contoh berikut, id yang dipilih adalah “london\_180”.

<!-- A Frame Video Sphere -->

<a-videosphere src="#london\_180" rotation="0 -90 0" …… >

</a-videosphere>

* **src** : source/alamat id video kita, pada dasarnya id tersebut akan merujuk pada alamat video asli yang telah di-set pada bagian <a-assets> (step nomor 3).

1. Di area line kode bawa pada file index.html, terdapat tag <script> dibawah komentar “Behaviour Scripts goes here”, kita bisa mengganti deskripsi konten kita melalui fungsi ***initDescription***. Deskripsi tidak boleh lebih dari 145 karakter.

//Course Description

var des = initDescription("In this course, you need to guess where do you think

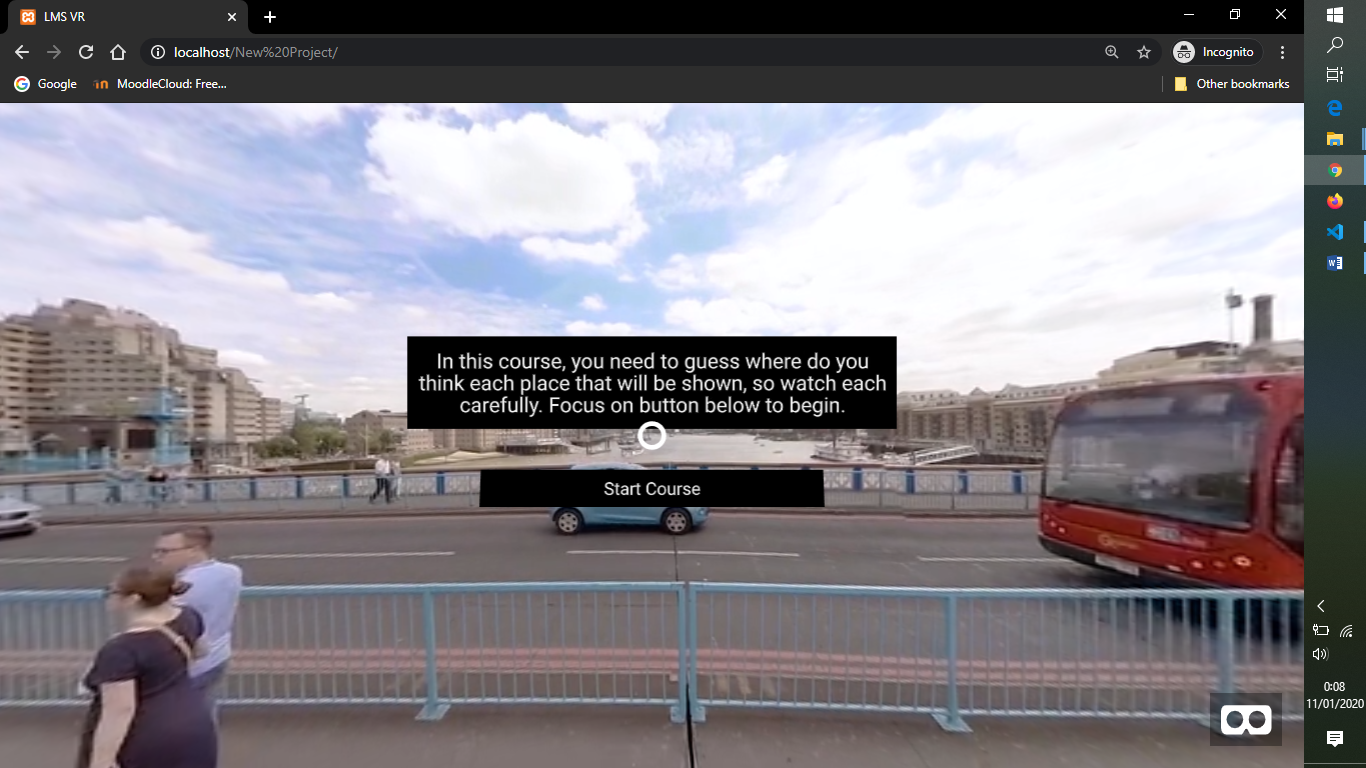
each place that will be shown, so watch each carefully. Focus on button below to

begin.");

Jika kita tidak memerlukan deskripsi untuk konten kita, non-aktifkan/komen kode pada line fungsi ***initDescription*** tersebut dengan menuliskan “//” pada awal line.

//var des = initDescription("Course description goes here.”);

1. Untuk men-debug, save file index.html. Kemudian, buka browser lalu pada address bar masukkan alamat “localhost/nama\_folder\_project/”. Dibagian bawah kiri, terdapat icon untuk masuk ke VR mode. Karena kita debugging di desktop, kita dapat melakukan mouse panning untuk beralih angle. Jika kita fokus pada tombol “Start Course”, maka video akan di-play.



1. **Membuat Interaksi Tanya Jawab (Question and Answers/Option)**

Kita akan menulis perintah dan behaviour/perilaku interaksi konten pada bahasa Javascript. Selain membuat fungsi sendiri, kita juga akan menggunakan pre-defined function (fungsi yang sudah jadi) yang disediakan oleh script “***lms-vr-functions-pack.js***” dalam yang terletak dalam folder “js”. Kita akan memunculkan interaksi question and answers/options pada detik tertentu setelah video di-play oleh tombol “Start Course”. Berikut langkah membuat interaksi question and answers/options.

1. Di area line kode bawa pada file index.html, terdapat tag <script> dibawah komentar “Behaviour Scripts goes here”, didalam tag inilah dimana disarankan kode Javascript yang mengatur behaviour interaksi dituliskan. Ada dua variable general dalam script tersebut, yaitu ***sceneEl*** dan ***startCourseBtn***. ***sceneEl*** adalah variable yang menampung referensi ke tag <a-scene> yaitu scene utama. Sedangkan ***startCourseBtn*** adalah variable yang menampung referensi ke button “Start Course”. Dibawah variable general tersebut, dibawah komentar “When Start Course button clicked” adalah fungsi yang akan dipanggil jika tombol “Start Course” dieksekusi.
2. Kita akan membuat fungsi baru bernama “***Interaction1***”. Fungsi ini akan mem-pause video. Kita akan memanggil fungsi ini pada saat tombol “Start Course” dieksekusi.

//When Start Course button clicked

startCoursBtn.addEventListener('click', function()

{

skyVideoTimeOutRequest(sceneEl, 15, "Interaction1");

});

//User defined behaviour goes below here

function Interaction1()

{

pauseSkyVideo();

}

Fungsi ***skyVideoTimeOutRequest*** dan ***pauseSkyVideo*** adalah fungsi pre-defined di “lms-vr-functions-pack.js”. ***skyVideoTimeOutRequest*** akan memanggil fungsi yang diberikan (dalam kasus ini “***Interaction1***”) pada detik tertentu pada video (dalam kasus ini, fungsi akan dipanggil ketika video pada detik ke 15). Fungsi ini membutuhkan 3 parameter, namun hanya 2 paramater terakhir yang perlu diganti value-nya tergantung kebutuhan, yaitu parameter detik ke (15), dan fungsi yang akan dipanggil (“***Interaction1***”). Sedangkan ***pauseSkyVideo*** adalah fungsi untuk mem-pause video.

1. Selanjutnya, kita akan membuat user interface berupa text panel pertanyaan (question) saat fungsi ***Interaction1*** dipanggil.

function Interaction1()

{

pauseSkyVideo();

var question = initTextPanel(sceneEl, "#txt-panel", "Where are you?");

}

Fungsi ***initTextPanel*** adalah fungsi pre-defined di “lms-vr-functions-pack.js”. Fungsi ini akan menginisialisasi text panel. Fungsi ini membutuhkan 3 parameter, namun hanya 1 paramater terakhir yang perlu diganti value-nya tergantung kebutuhan, yaitu text yang akan ditampilkan, text harus tidak lebih dari 92 karakter. (pada kasus ini value-nya “Where are you?"). Fungsi ini juga akan mengembalikan/return referensi komponen text panel yang telah diinisialisasikan, pada kasus ini kita tamping pada variable ***question***.

1. Selanjutnya, kita akan membuat user interface berupa button option (jawaban) saat fungsi ***Interaction1*** dipanggil.

function Interaction1()

{

pauseSkyVideo();

var question = initTextPanel(sceneEl, "#txt-panel", "Where are you?");

var options =

[

{text: "Mount Everest", method: function()

{

setVideoSky("#beach\_180", true);

}},

{text: "London Tower Bridge", method: function()

{

setVideoSky("#beach\_180", true);

}}

];

initOptions(sceneEl, "#opt", options , function()

{

clearTextPanel(question);

});

}

Untuk membuat **option** (jawaban), pertama kita harus membuat **array of object, object tersebut terdiri dari 2 property** yaitu text yang akan ditampilkan (harus tidak lebih dari 42 karakter), serta method (fungsi) yang akan dipanggil ketika object/option dipilih. Kita dapat membuat object option sebanyak mungkin, namun disarankan tidak melebihi 4 option untuk menjaga konsistensi estetis layout.

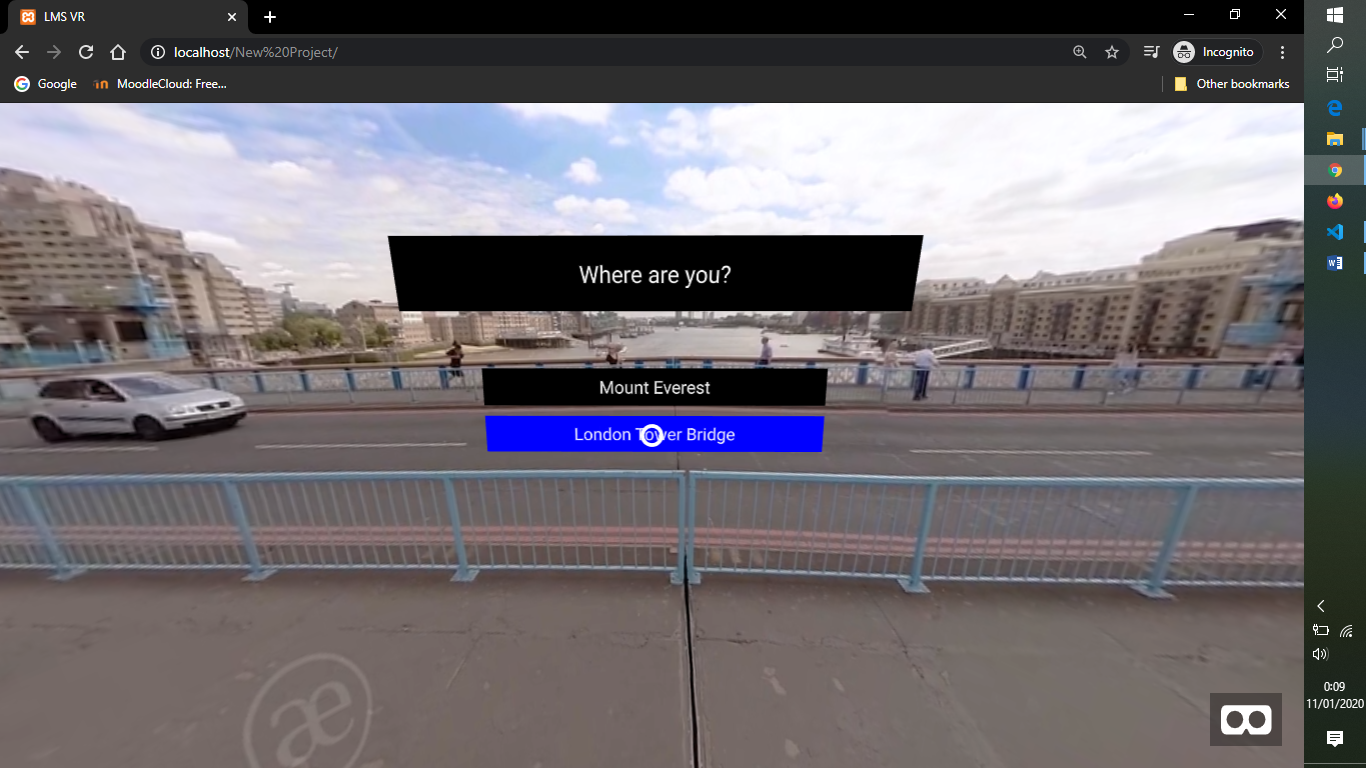
Pada kode diatas, array of object-nya ditampung oleh variable bernama ***options***, yang memiliki 2 object option, yaitu dengan text “Mount Everest” dan “London Tower Bridge”. Tiap object option tersebut, sama-sama akan memanggil fungsi ***setVideoSky*** dengan parameter yang sama.

***setVideoSky*** adalah fungsi pre-defined di “lms-vr-functions-pack.js” yang akan men-set/mengganti video 180/360 yang akan ditampilkan. Fungsi ini membutuhkan 2 parameter, yaitu id video (dalam format “#id\_video” yang akan ditampilkan (dalam kasus ini “#beach\_180”), dan autoplay video dimana true jika autoplay, false jika tidak (dalam kasus ini true, yang berarti video akan di-play tepat setelah video di-set).

***initOptions*** adalah fungsi pre-defined di “lms-vr-functions-pack.js” yang akan menginisialisasi button panel ber-text. Fungsi ini memerlukan 4 parameter, namun hanya 2 parameter terakhir yang perlu diganti sesuai kebutuhan, yaitu parameter array of option object (dalam kasus ini variable options), serta parameter fungsi yang akan dipanggil setelah salah satu option dipilih (dalam kasus ini ***clearTextPanel***).

***clearTextPanel*** adalah fungsi pre-defined di “lms-vr-functions-pack.js” yang akan menghapus/destroy komponen text panel yang telah terinisialisasi. Fungsi ini memerlukan 1 parameter, yaitu komponen yang akan dihapus (pada kasus ini variable ***question***).

1. Save index.html dan debug di browser. Saat kita fokus pada tombol “Start Course”, maka video di-play, ketika video pada detik ke-15, user interface pertanyaan serta opsi akan ditampilkan. Jika kita memilih salah satu opsi, maka video akan dialihkan ke video ber-id “#beach\_180”.



1. Setelah memahami step 1-4, kita dapat membuat interaksi tanya jawab dengan cara yang sama. Berikut fungsi bernama “***Interaction2***” yang mengatur behaviour interaksi selanjutnya, yang dipanggil setelah ketika detik ke-10 video. (*Jangan bingung dengan syntax “……”, itu hanya tanda bahwa syntax yang sebelumnya disembunyikan, agar snipset menampilkan kode yang penting saja*).

function Interaction1()

{

……

initOptions(sceneEl, "#opt", options, function()

{

clearTextPanel(question);

timeoutFunctionRequest(1000, function()

{

skyVideoTimeOutRequest(sceneEl, 10, "Interaction2")

});

});

}

function Interaction2()

{

pauseSkyVideo();

var question = initTextPanel(sceneEl, "#txt-panel", "Where are you again?");

var options =

[

{text: "Peble Beach ", method: null},

{text: "Bavarian Alps", method: null}

];

initOptions(sceneEl, "#opt", options , function()

{

clearTextPanel(question);

});

}

Ada sedikit perubahan saat memanggil fungsi ***skyVideoTimeOutRequest*** pada fungsi ***Interaction1***, dimana fungsi ini diletakkan didalam fungsi ***timeoutFunctionRequest***. ***timeoutFunctionRequest*** adalah fungsi pre-defined di “lms-vr-functions-pack.js” yang akan memanggil fungsi parameter setelah millisecond tertentu. Fungsi ini membutuhkan 2 parameter, yaitu millisecond (pada kasus ini 1000 millisecond atau 1 detik) dan fungsi yang akan dipanggil (pada kasus ini ***skyVideoTimeOutRequest***).

Mengapa pada kasus ini kita perlu menjeda selama 1 detik? Karena apabila kita perhatikan, ketika video diganti/transisi antara video dengan id “#london\_180” dan “#beach\_180”, terdapat animasi fading, kemudian baru menampilkan dan memulai video. Animasi fading tersebut membutuhkan waktu 1 detik. Apabila kita panggil fungsi ***skyVideoTimeOutRequest*** tanpa jeda 1 detik, pada kasus ini maka request waktu video yang dituju oleh ***skyVideoTimeOutRequest*** bukan video baru yang di-set, melainkan video sebelumnya, sehingga kemungkinan besar terjadi forever loop. Oleh karena itu, apabila kita ingin mengganti video sekaligus request fungsi ***skyVideoTimeOutRequest***, maka gunakan jeda 1 detik (1000 millisecond) menggunakan fungsi ***timeoutFunctionRequest***.

Pada fungsi ***Interaction2***, objek-objek variable ***options*** memiliki nilai value method ***null*** (berarti kosong/tidak ada), yang berarti ketika dipilih opsi tersebut, maka tidak memanggil fungsi apapun.

1. Interaksi pada fungsi ***Interaction2*** akan kita jadikan interaksi terakhir, sehingga setelah interaksi tanya jawab pada fungsi ini, kita akan mengakhiri konten. Caranya sebagai berikut.

function Interaction2()

{

……

initOptions(sceneEl, "#opt", options , function()

{

clearTextPanel(question);

finishCourse(function()

{

initTextPanel(sceneEl, "#txt-panel", "Course Finished");

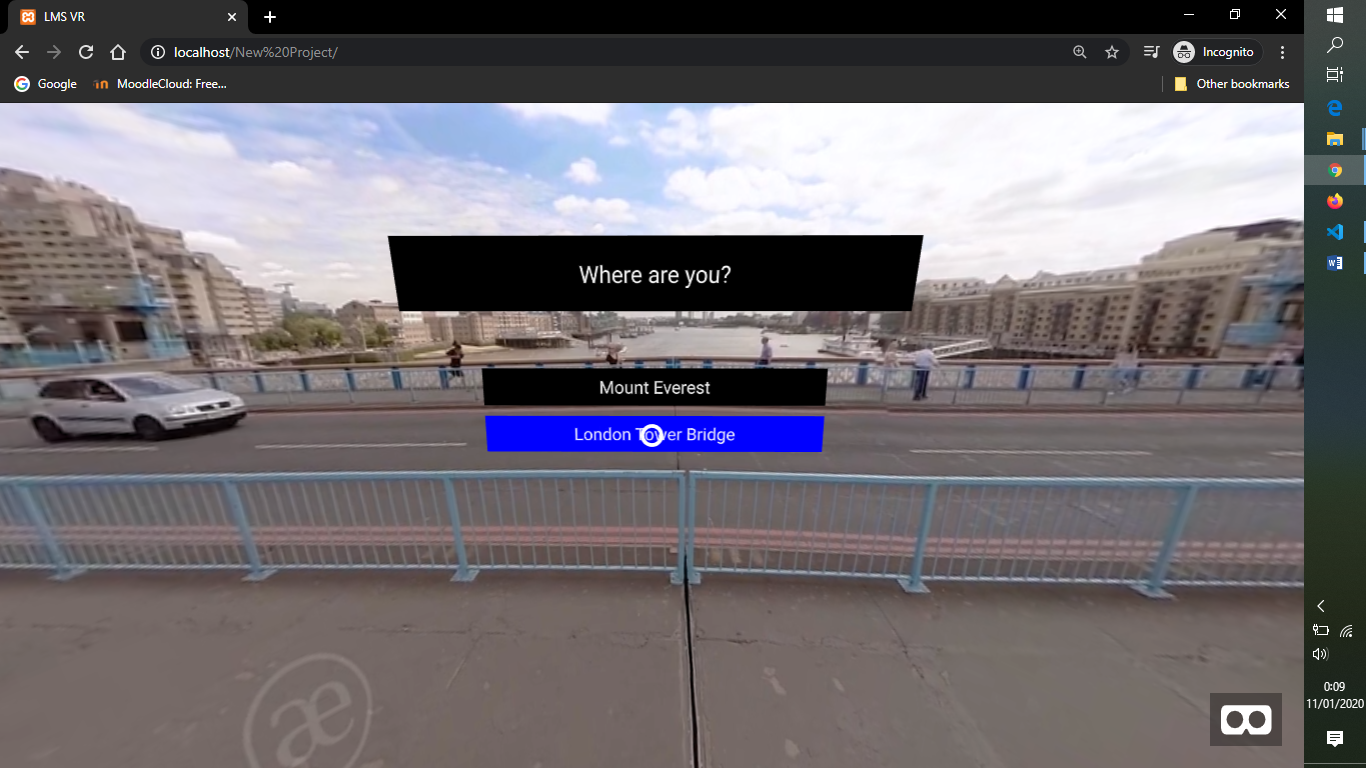
});

});

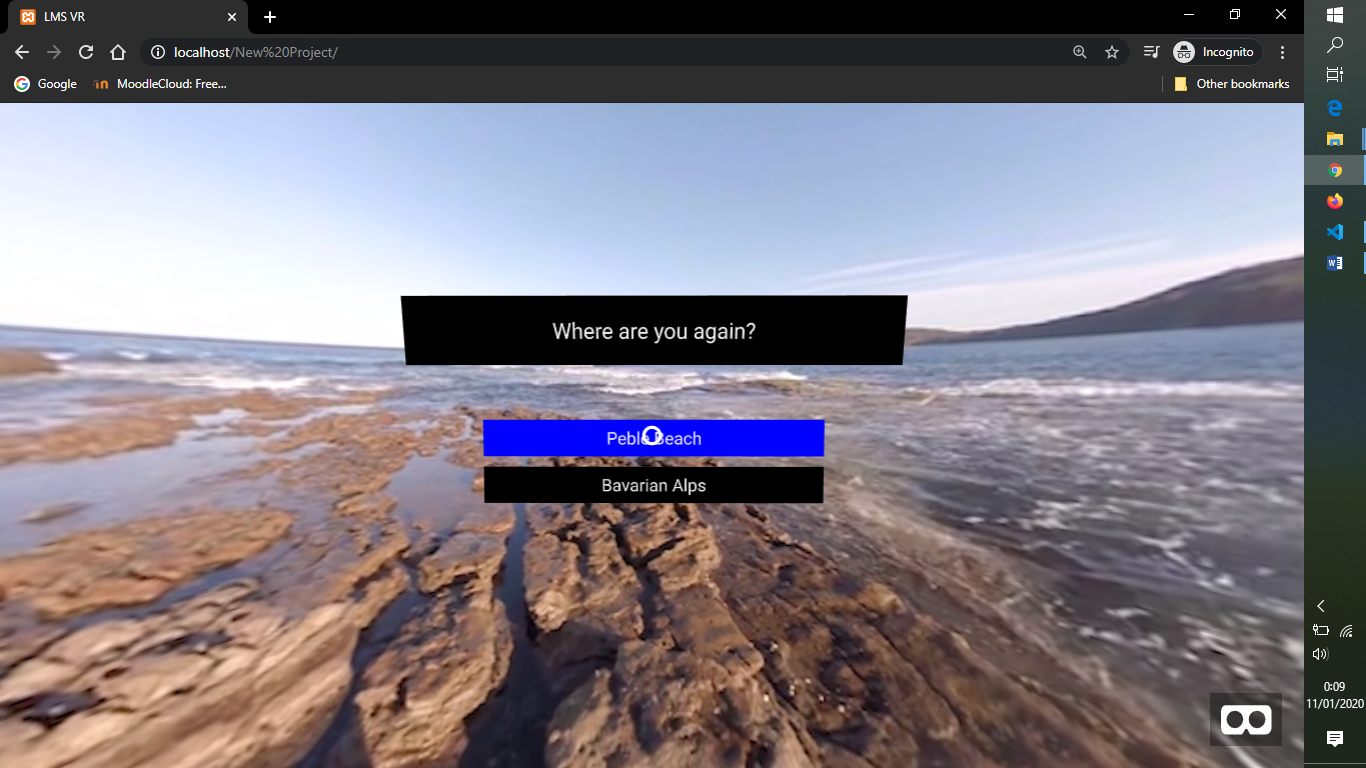
}

***finishCourse*** adalah fungsi pre-defined di “lms-vr-functions-pack.js” yang akan mengakhiri konten. Ketika fungsi ini dipanggil, video akan fade out hitam. Fungsi ini membutuhkan 1 parameter, yaitu fungsi yang akan dipanggil ketika konten berhasil diakhiri. Pada kasus ini, ketika konten diakhiri, kita akan memunculkan text panel bertuliskan “Course Finished” menggunakan fungsi ***initTextPanel***.

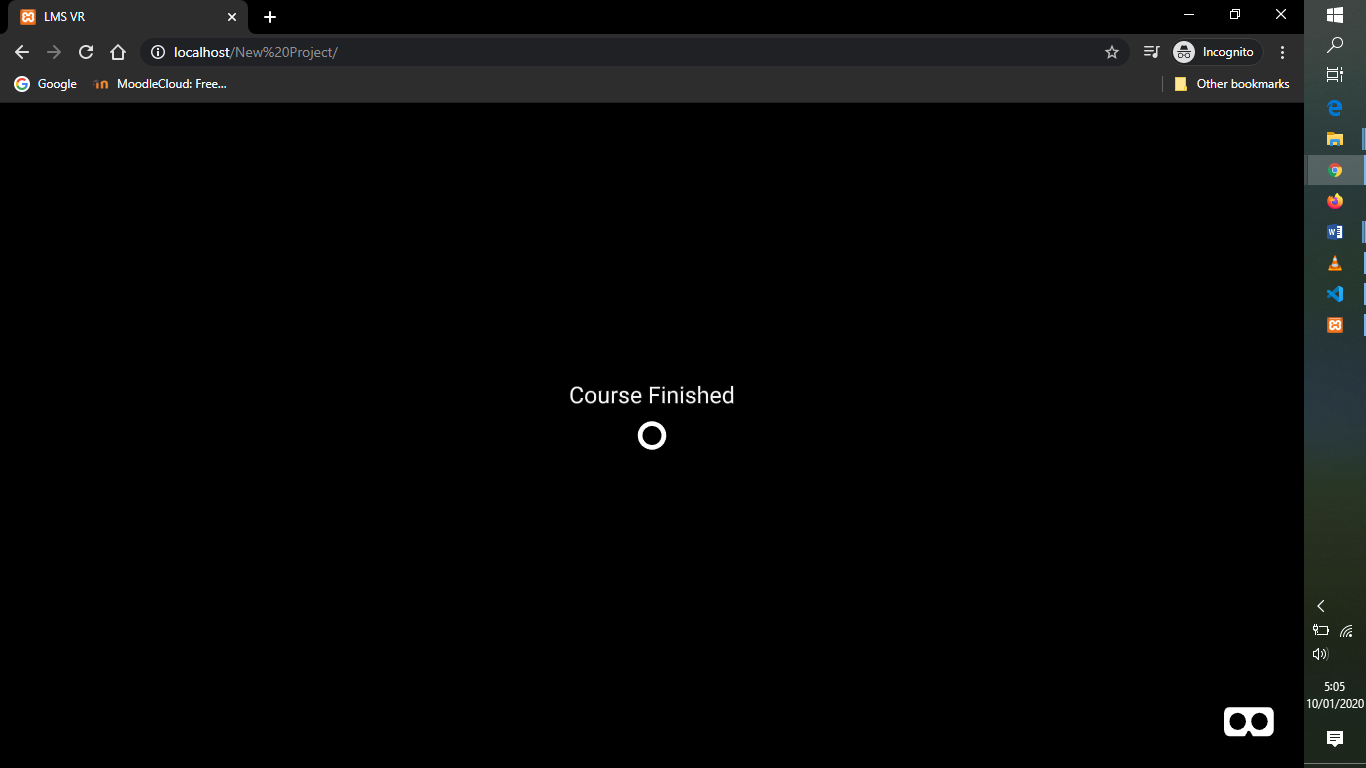
1. Save index.html dan debug di browser. Saat kita fokus pada tombol “Start Course”, maka video di-play, ketika video pada detik ke-15, user interface pertanyaan serta opsi akan ditampilkan.



Jika kita memilih salah satu opsi, maka video akan dialihkan ke video ber-id “#beach\_180”. Ketika video pada detik ke-10, user interface pertanyaan serta opsi akan ditampilkan.



Jika kita memilih salah satu opsi, maka video akan fade out dan ditampilkan text panel “Course Finished”.



1. **Input Score ke LMS Menggunakan SCORM**

Karena konten yang kita berbasis web, untuk berinteraksi dengan LMS kita bisa menggunakan script Javascript. Sebelum itu, ada beberapa hal yang perlu dipahami. Pertama, kita akan mem-package project kita menjadi SCORM package. Web dan LMS dapat berinteraksi menggunakan API SCORM yang disediakan oleh LMS, yaitu service dimana kita meng-upload package SCORM kita. Kita akan menggunakan fungsi yang disediakan oleh file “scorm12.js” dalam folder “scorm”.

Siklus interaksi web dengan LMS menggunakan SCORM memiliki 4 state. State-state tersebut telah diatur oleh fungsi dalam script “scorm12.js” yaitu,

|  |  |
| --- | --- |
| **Nama Fungsi** | **Behaviour** |
| LMSInitialize() | Menginisialisasi/membuka koneksi antara web dengan LMS. |
| LMSGetValue(data\_model) | Mendapatkan/return value dari LMS. Fungsi ini membutuhkan 1 parameter yaitu string data model SCORM. Contoh, LMSGetValue(“com.core.student\_name”); akan mengambil nama student yang mengakses konten. |
| LMSetValue(data\_model, value) | Men-set value ke LMS. Fungsi ini membutuhkan 2 parameter yaitu string data model SCORM, dan value yang akan di-set. Contoh, LMSSetValue(“cmi.core.score.raw”, 80); akan men-set value score 80 ke database student yang mengakes. |
| LMSCommit() | Mensubmit value yang telah di-set oleh LMSSetValue ke LMS. |
| LMSFinish() | Menutup koneksi antara web dengan LMS. |

Pada textbook ini, kita tidak melakukan interaksi yang kompleks. Kita hanya akan men-set score student sesuai dengan nilai pada sesi interaksi tanya jawab, sebagai berikut.

1. Untuk proses scoring, kita dapat menggunakan fungsi ***addScore*** yang terdapat dalam file “lms-vr-functions-pack.js”. ***addScore*** akan mengakumulasi/menambahkan score, dan fungsi ini membutuhkan 1 parameter yaitu nilai yang akan di-set. Nilai akumulasi harus pada range 0 <= x <= 100 (nol sampai dengan 100). Pada ***bagian c*** sebelumnya ada 2 sesi tanya jawab, kita akan menambahkan score 50 untuk jawaban yang benar, sehingga jika total score jika semua jawaban yang benar adalah 100.

function Interaction1()

{

……

var options =

[

……

{text: "London Tower Bridge", method: function()

{

setVideoSky("#beach\_180", true);

addScore(50);

}}

];

……

}

function Interaction2()

{

……

var options =

[

{text: "Peble Beach", method: function()

{

addScore(50);

}},

……

];

……

}

1. Save index.html. Untuk men-debug fungsi scoring, kita harus mem-package project kita ke SCORM package dahulu, kemudian meng-upload-nya ke LMS, baru kita bisa men-debug langsung di LMS. Jika kita men-debug langsung dari localhost, kita akan mendapatkan error karena API SCORM tidak ditemukan (karena API SCORM disediakan oleh LMS).
2. **SCORM Packaging**

Pada dasarnya, untuk mem-package ke SCORM package sangatlah sederhana. Ada dua komponen utama, yaitu imsmanifest.xml dan index.html kita (nama file harus seperti itu). Kemudian, semua file di project dicompress menjadi file zip. Kita akan menggunakan file imsmanifest.xml yang disediakan oleh template sebelumnya dengan sedikit modifikasi. Berikut proses packaging ke SCORM package.

1. Buka file imsmanifest.xml. Didalam tag <organization> dibawah komentar “Course Informations” adalah course title dan sub-course title. Kita bisa menggantinya sesuai kebutuhan.

<organization identifier="B0">

<!-- Course Information-->

<title>My Course Title</title>

<item identifier="i1" identifierref="r1" isvisible="true">

<title>My Sub-Course Title</title>

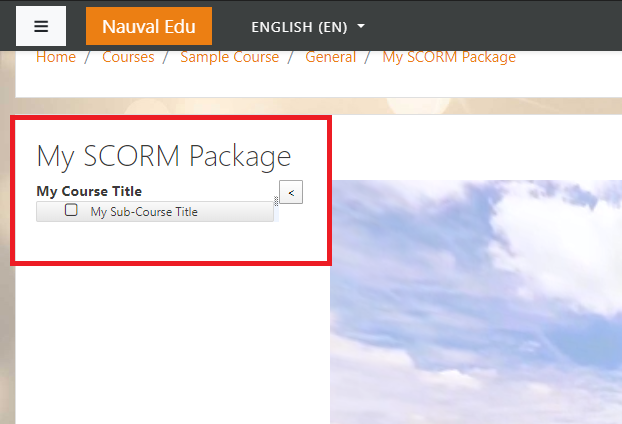
</item>

</organization>

Sebenarnya, kita bisa membuat multiple sub-course, namun pada textbook ini kita hanya akan membuat 1 sub-course yaitu konten interactive 180 video kita.

1. Kemudian, setelah men-set title dan sub-title, save file imsmanifest.xml. Lalu, pilih semua file di project (bukan folder-nya), kemudian compress menjadi file zip. File zip itulah SCORM package konten kita.

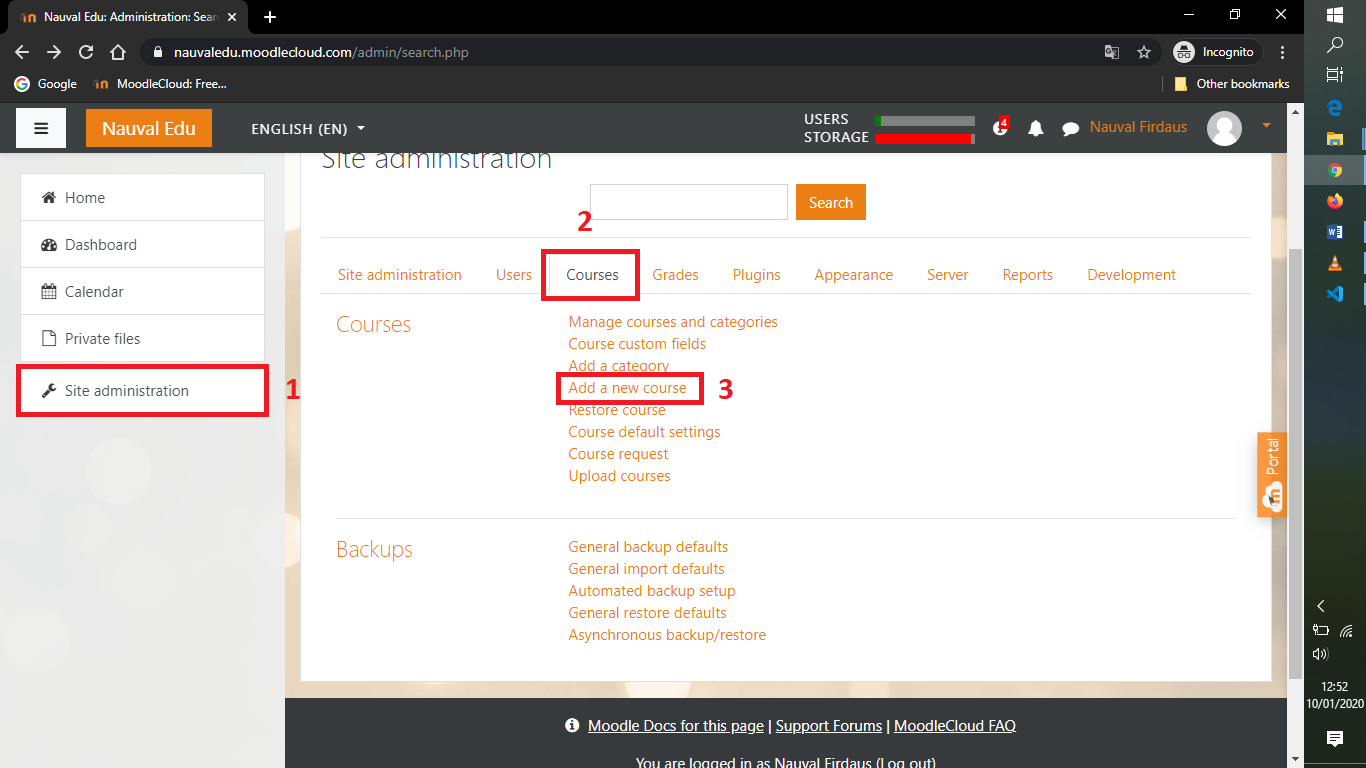
Jika package sudah kita upload, pada LMS Moodle, title dan sub-course title akan terlihat pada bagian kanan konten.



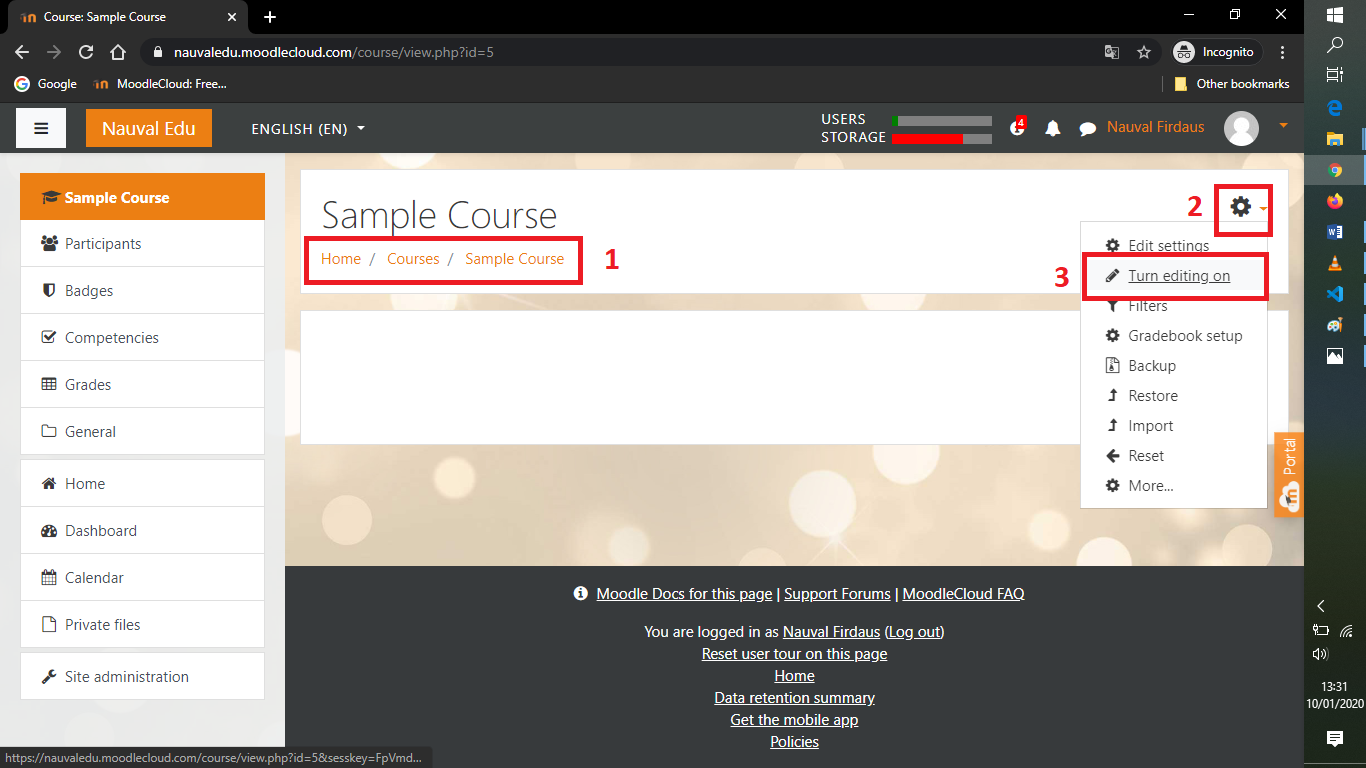
1. **Publish SCORM Package ke LMS**

Pada textbook ini, kita akan meng-upload dan mem-publish package SCORM ke LMS Moodle. Berikut langkah-langkahnya.

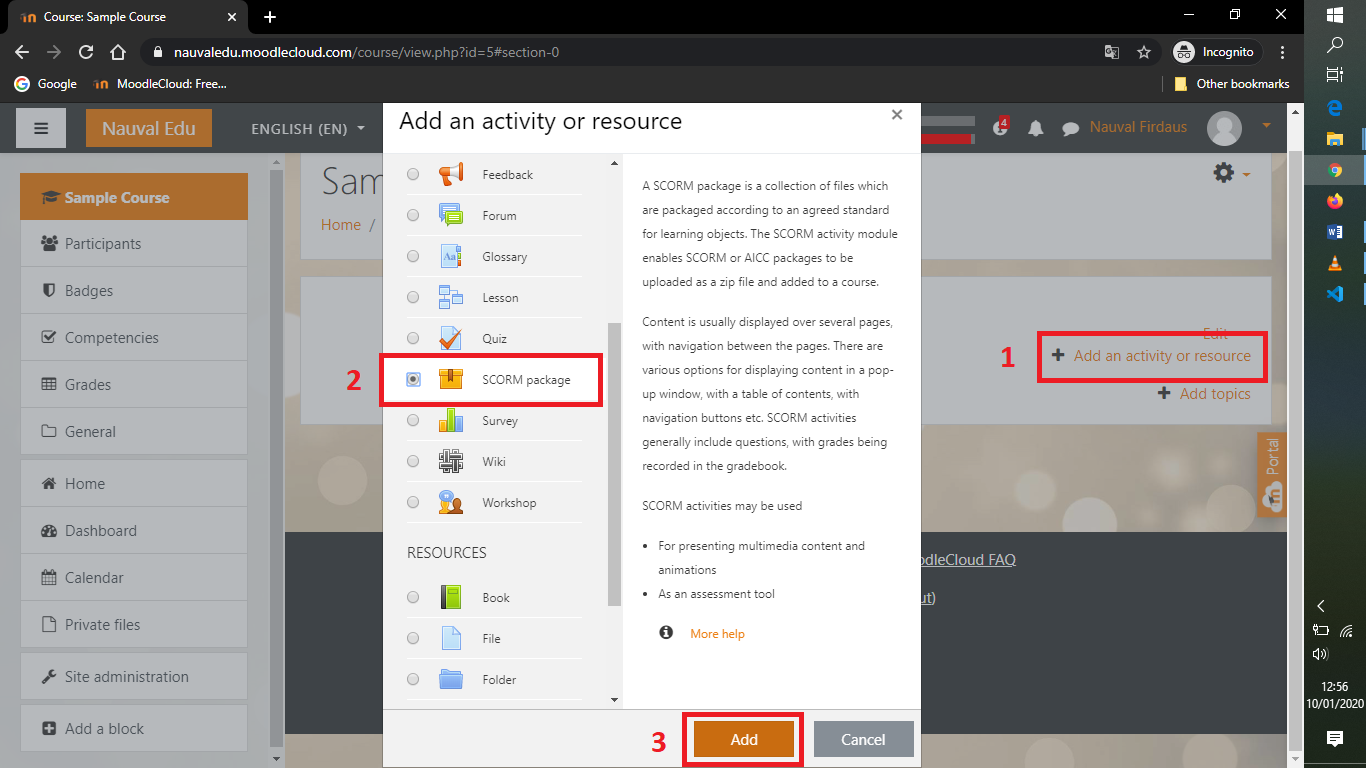
1. Pada textbook ini, kita akan membuat course baru. Seperti biasanya, buka akun Moodle Cloud, kemudian pada tab ***Site Administration*** buka tab ***Course*** lalu ***Add a new course***. Kemudian beri nama dan setup sesuai keinginan, lalu save.



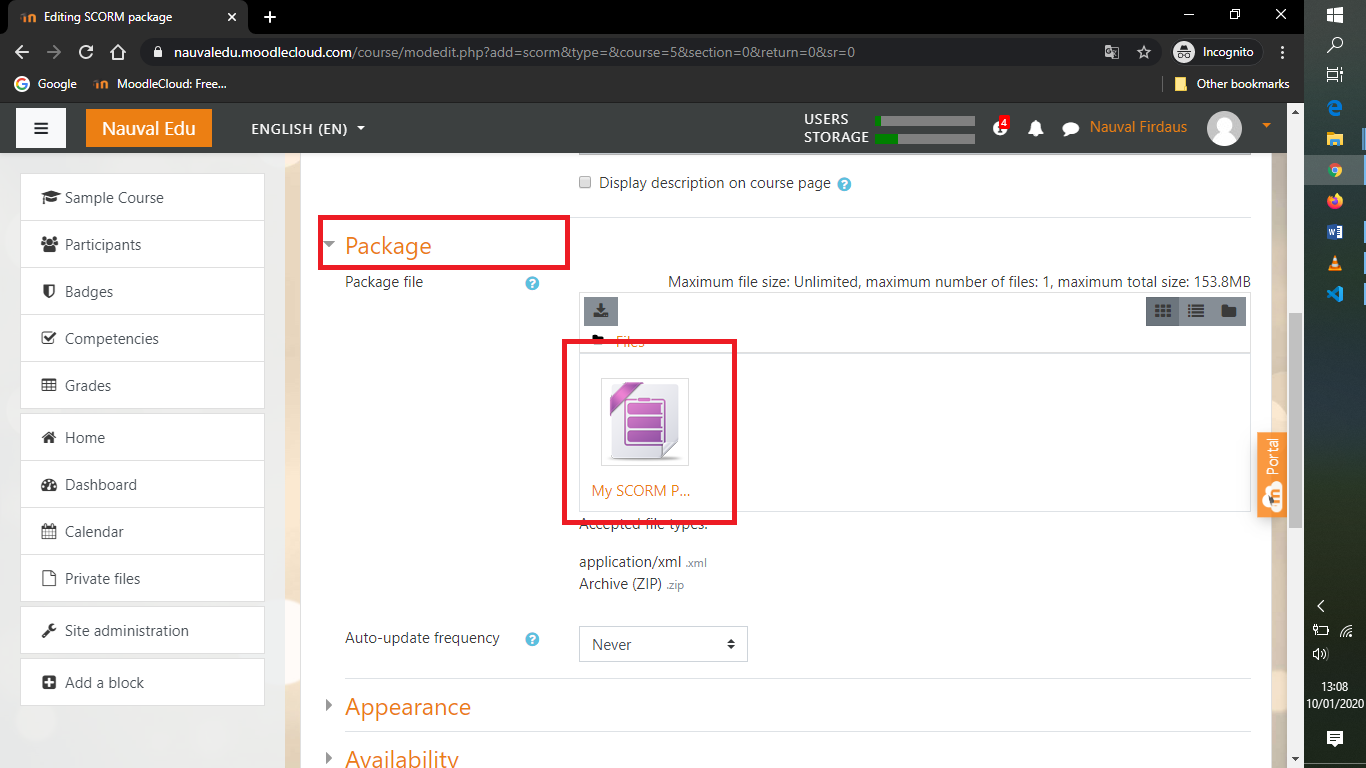
1. Kemudian, buka course yang baru dibuat. Lalu aktifkan “Turn editing on”, sehingga kita dapat menambahkan konten SCORM package baru.



Pilih “Add an activity or resource” dan pilih “SCORM Package”, kemudian klik “Add” untuk menambahkan.

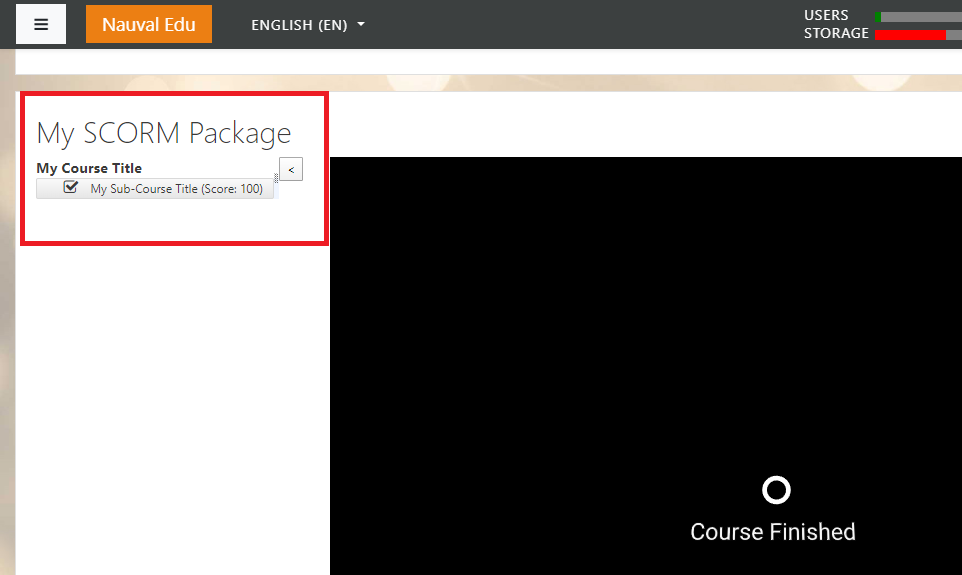


Kemudian, isi field nama dan field lainnya sesuai kebutuhan. Kemudian, pada section “Package”, drag and drop SCORM package kita untuk meng-upload. Tunggu sampai proses upload selesai, kemudian tekan “Save and Display” untuk menyimpan dan mengalihkan page ke tampilan konten.

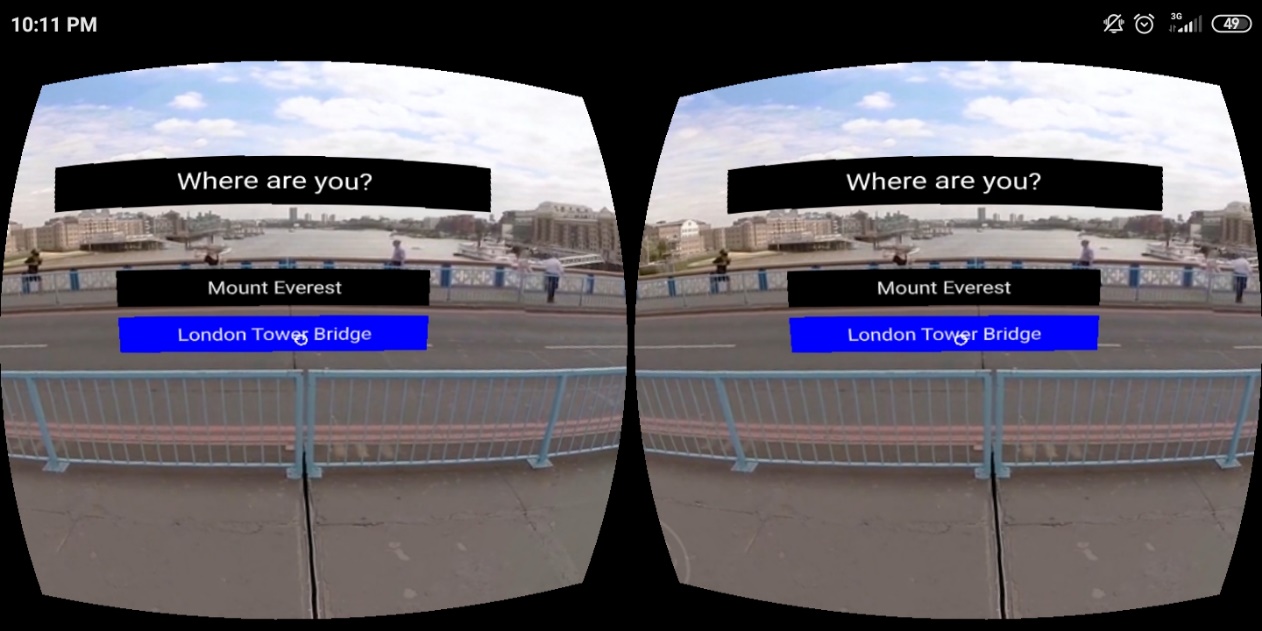


Kemudian, tekan “Enter” untuk masuk ke konten.

1. Dari sini, konten kita sudah dipubllish serta siap digunakan. Apabila kita menyelesaikan course-nya, maka pada bagian kanan page, status course akan dinyatakan selesai (tanda checkmark) dan score sudah ditetapkan.



1. Apabila kita membuka konten melalui aplikasi mobile Moodle, dan menekan tombol VR pada bagian kiri konten, kita akan masuk ke mode seperti Google Cardboard.



Baiklah, konten video 180 interaktif kita sudah selesai. Berikut link sample course <https://nauvaledu.moodlecloud.com/>, masuk sebagai username “**guest\_01**” dan password “**guest\_01**”.

**KESIMPULAN**

A-Frame adalah sebuah web development framework yang khusus ditujukan untuk pengembangan WebVR dengan syntax mengikuti tag-tag HTML. Komponen-komponen framework ini juga dapat dimodifikasi sesuai kebutuhan melalui perintah-perintah kode Javascript. SCORM adalah standar pendistribusian paket e-Learning yang dapat digunakan untuk menampung berbagai spesifikasi dan standar untuk konten e-Learning berbasis web.

Dengan menggunakan A-Frame dan SCORM, kita dapat membuat konten WebVR yang dapat berinteraksi dengan LMS (Learning Management System), pada textbook ini kita akan membahas konten VR Interactive 180 Video. Pada dasarnya, kita akan membangun user interface; seperti button, text, dan 360 video viewer; menggunakan framework A-Frame. Kemudian untuk interaksi dengan LMS, kita menulis kode Javascript untuk men-setup value-value LMS menggunakan wrapper SCORM.

**GLOSSARY**

* **A-Frame**
* **HTML**

HyperText Markup Language; Standard Bahasa markup untuk membuat halaman web.

* **Javascript**
* **LMS**
* **User Interface**
* **SCORM**
* **Fungsi/Function/Method**
* **Parameter**
* **Property/Attribute**
* **Debug**
* **WebVR**

Virtual reality yang diakses melalui web.

* **Wrapper**

Entity yang mengenkapsulasi item lain, digunakan untuk mengkonvers data menjadi format yang kompatibel atau menyembunyikan kompleksitas algoritma. Tujuannya adalah agar sebuah fungsi dapat diakses dengan mudah, yaitu hanya sekedar memanggil nama fungsi dan passing parameter.