Software Requirements Specification

for

OpenSim Marketplace

Version 1.0 approved

Prepared by Ryan Wickett, Ryan Smith, Jonas Wojtas, Jordan Williams

Team 12

10/11/2020

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Initial	10/11/2020	Initial version	1.0

1. Product Overview

1.1 Problem Overview

Users of OpenSim require a streamlined method of acquiring and sharing assets used in the OpenSim client. Assets are currently shared between users through use of direct requests which works but can lead to inconsistencies and is generally inefficient.

1.2 Project Scope

The OpenSim Marketplace is a web interface that will allow users of OpenSim to search, download and upload assets used in the OpenSim client. This marketplace will provide an easy to use and efficient method to share assets between users and download shared assets directly into their OpenSim inventory, ready for use. Users will be able to login via their OpenSim Account and search across the asset database to find the assets they need.

2. Overall Description

2.1 Product Features

The system will provide the user with a web page interface that can be accessed via the users OpenSim account in which the user will be able to search for, download, and upload OpenSim assets. The assets will be able to accessed via OpenSim Inventory.

2.2 User Classes and Characteristics

Listed in order of highest to lowest priority:

NUWC Newport Virtual Worlds:

Virtual Worlds developers at NUWC Newport will have direct access to the marketplace and its backend, and will be responsible for maintaining the system and administrative functions.

All Virtual Worlds Users:

The entire set of users (including other base locations, e.g. Hawaii) who have access to the marketplace in order to upload/download assets for their local Virtual Worlds projects.

2.3 Operating Environment

The hardware platform is any Windows 10 machine. The software will be interacting and integrated with the OpenSim client meaning the software must coexist with said client.

2.4 Design and Implementation Constraints

Some issues or items that may limit the development options are the integration with OpenSim, since the created technology must be integrated into the OpenSim Client. There are no other technology limitations outside of that, aside from a MySQL database being required. There is also a policy regarding any used libraries or other possibly copyrighted material used in the program must be a MIT license, meaning that the used libraries and software is open source.

2.5 User Documentation

A User-manual will be provided with the software outlining its functionality as well as how to use said functionality.

2.6 Assumptions and Dependencies

The marketplace can only function via interaction with a preexisting OpenSim server and accessed through the Firestorm client. NUWC Virtual Worlds teams will already have knowledge of how to access NUWC OpenSim servers via Firestorm and servers will already exist.

This project works under these assumptions and will not be providing instructions on how to setup an OpenSim server/Firestorm client, as there are many well-documented resources online for this purpose.

3. Requirements Specifications

3.1 Features -- Functional Requirements

3.1.1 Start/Stop website:

3.1.1.1. Description

NUWC Newport Virtual Worlds developers will be able to start and stop the website service via the command prompt. This will be important for website integration, maintenance and upgrades.

3.1.1.2. Requirement

This is a basic function of the website that must exist for any developers working on it. Crashes will be detailed in the log for maintenance and debugging purposes. Integration with Diva Wifi: TBD.

3.1.2 User account integration:

3.1.2.1. Description

The marketplace will associate assets and asset ownership with OpenSim users, the database(s) for which already exist on NUWC OpenSim servers.

3.1.2.2. Requirement

This feature will require interaction with the OpenSim user database through MySQL. Details will become more clear after further research into Diva Wifi websites.

3.1.3 Download assets from the marketplace website to the user's machine:

3.1.3.1. Description

Users will have the ability to download assets directly onto their machine through the website.

3.1.3.2. Requirement

This feature will require interaction with the OpenSim asset database through MySQL. Asset files (.txt, scripts, models, etc.) will be pulled from the database and saved to their machine via a file explorer window. A download button will be required on each asset's webpage.

3.1.4 Upload assets to the marketplace website from the user's machine:

3.1.4.1. Description

Users will have the ability to upload assets onto the marketplace website from their machine.

3.1.4.2. Requirement

This feature will require interaction with the OpenSim asset database through MySQL. Asset files (.txt, scripts, models, etc.) will be selected from the user's machine via a file explorer window, validated, and then uploaded to the asset database. An upload button will be required on each asset's webpage.

3.1.5 Download assets from the marketplace website to the user's OpenSim inventory:

3.1.5.1. Description

Users will have the ability to download assets directly into their OpenSim inventory through the website.

3.1.5.2. Requirement

This feature will require interaction with the OpenSim asset database through MySQL. Asset files (.txt, scripts, models, etc.) will be pulled from the database and saved to their OpenSim inventory. A download button will be required on each asset's webpage.

3.1.6 Upload assets to the marketplace website from the user's OpenSim inventory:

3.1.6.1. Description

Users will have the ability to upload assets onto the marketplace website from their OpenSim inventory.

3.1.6.2. Requirement

This feature will require interaction with the OpenSim asset database through MySQL. Asset files (.txt, scripts, models, etc.) will be selected from the user's OpenSim inventory via Diva Wifi (details TBD), validated, and then uploaded to the asset database. An upload button will be required on each asset's webpage.

3.1.7 Individual web pages dedicated to each asset:

3.1.7.1. Description

Every asset uploaded to/available for download from the marketplace website will have its own webpage with associated details about the asset.

3.1.7.2. Requirement

React will be used to automatically generate the asset webpage when it is loaded by a user. The web page will contain a description, asset preview, a download button, etc. (TBD).

3.1.8 Previews for each asset on the marketplace website:

3.1.8.1. Description

At the bare minimum each asset will have an icon or image representing the type of asset it is on the marketplace (a cog icon for scripts, cube for model, etc.). Ideally, textures and models would have an image preview on the website that displays the actual asset.

3.1.8.2. Requirement

For basic image per asset type display, the asset's file type will need to be read upon marketplace rendering for the browser to select the correct icon. Textures will likely be displayed just by displaying the entire or a portion of the 2D texture file. Model preview details are TBD.

3.1.9 Sort assets by property:

3.1.9.1. Description

Users will have the ability to sort the list of assets by a few reasonable properties, including date uploaded, asset name, popularity, etc. (TBD).

3.1.9.2. Requirement

This will require a stored procedure in the asset database that has an optional parameter which sorts the result set (list of assets) when called.

3.2 Nonfunctional Requirements

3.2.1 Performance Requirements

Viewing the list of assets and downloading/uploading individual assets must be efficient. An acceptable execution time is under 1 second (TBD).

3.2.2 Safety Requirements

The safety of the data (assets, user information, etc.) is the major concern when it comes to safety in regards to the OpenSim Marketplace. To prevent the loss of said data, safeguards such as data backups as well as proper security measures will be provided.

3.2.3 Security Requirements

The security of the server that hosts the database and website as well as the user who uses said website are the major concern in regards to security. Login information provided by the user must be secure and access to the server and database must be restricted to OpenSim developers and other permitted 3rd parties only to prevent server/data breaches. Integration with OpenSim will provide some security through shared login information for both the OpenSim client and marketplace.

3.2.4 Software Quality Attributes

Maintainability:

Our product will provide easy to understand and use code and documentation which will help future developers keep the product up to date as well as ensure users who will have to deal with as little downtime as possible. Code will be written with readability as well as modularity in mind as code that is split up and organized based on functionality will be easier to read and therefore to maintain. Documentation such as a user-manual will provide a complete description of how our product works which will help future developers understand our code and ultimately maintain the product.

Reliability:

Our product will provide error checking to ensure that no bad data or queries are being used. This comes in the forms of checks to ensure correct file formats and uncorrupted data is being uploaded and downloaded to and from the database, checks to ensure that users are connected and secure while using the marketplace, and checks to ensure data searched and displayed on the site is accurate, to name a few.

4. External Interface Requirements

4.1 User Interfaces

There most likely will be a keyboard shortcut or multiple to open the marketplace in the Opensim client. Any GUI standards are still TBD, but there are some major details that are definitely important such as a search bar for items, sorting by name, date etc for the items in the search, as well as buttons to download the items directly from the search menu.

4.2 Hardware Interfaces

Keyboard - Will enter in any needed text input.

Mouse - Will allow traversing through clicking buttons and scrolling

4.3 Software Interfaces

Nodejs <12.18.4> - A backend service that gives javascript access to data outside of a browser. This is used to communicate to other servers and databases.

Express <4.17.1> - Backend web framework that gives an easy to use API that interacts with a frontend. Any interaction that a user wants to have with persistent data must use that framework as a medium.

MySQL (Nodejs) <2.18.1> - A library for Nodejs that allows the backend to send SQL queries to a MySQL database.

React <16.13.1> - A library allowing dynamic components for a web interface.

Axios <0.20.0> - Interacts between React and Nodejs and sends endpoints for data queries.

React Bootstrap <1.3.0> - A library for making premade UI components.

4.4 Communications Interfaces

We have multiple communication functions within our program thus far; Axios is one of them, which is responsible for connecting our GUI with the back end server by sending requests from the front end interface to the Nodejs server. We also use Express functions as a buffer to handle any data transactions from the front end, especially to and from the database. Finally, MySQL (Nodejs) handles all SQL queries to retrieve items from the database requested by the backend.

5. Other Requirements

N/A

Appendix A: Glossary

Assets	Models, scripts, textures, or meshes that are used and shared by Virtual Worlds users.
Diva Wifi	An add-on module for OpenSim that is involved in creating user accounts as well as inventory and profile management (Wifi is the abbreviation of "Web Interface for I").
Firestorm Client	An open source 3D virtual world software for Mac Windows and Linux created by the Phoenix Project
JavaScript	A programming language used commonly for web-development. Used for React and Nodejs.
MySQL	An open source database management system currently utilised by NUWC's Virtual Worlds team.
NodeJS	An open source back-end framework for javascript.
NUWC	Naval Undersea Warfare Center (Division Newport)
OpenSim	An open source multi-platform, multi-user 3D application server. It can be used to create a virtual environment (or world) which can be accessed through a variety of clients.
OpenSim Client	The application end-users use to interact with the OpenSim Server (i.e. Virtual Worlds). Responsible for rendering the world, taking user input, etc.
OpenSim Client Inventory	A space within the OpenSim Client where assets the user has created/downloaded are stored.
OpenSim Server	Hosts a 3D multiplayer world which can allow users to interact with one another, connecting individual OpenSim clients
React	Open Source Javascript Library for building a front-end user interface.
Robust	"Redesigned OpenSimulator Basic Universal Server Technology"
Second Life	The main commercial application which uses OpenSim..
Virtual World	Computer simulated 3D environment

Appendix B: Analysis Models

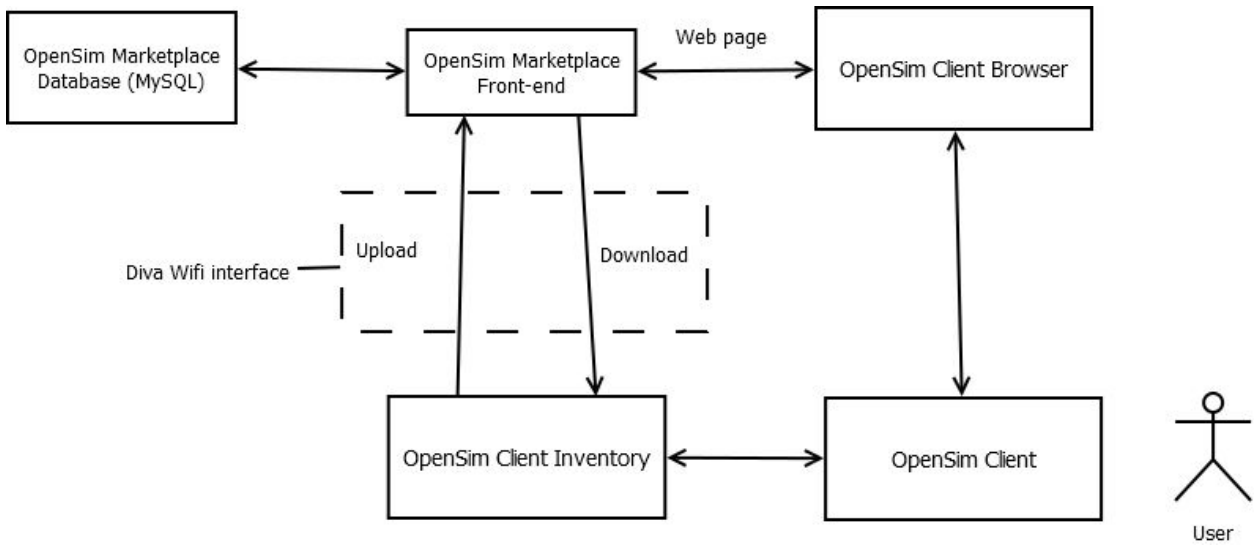

Figure 1: Sample interaction model between the OpenSim client, inventory, marketplace, and database.

Appendix C: Issues List

Issues

- **OM-34**: Create sample website with Diva Wifi
- **OM-20:** Ability to organize items based on certain fields
- **OM-19:** Ability to click an item to inspect its name, description, etc.
- **OM-23**: Ability to download data from database
- **OM-21**: Ability to upload data to database
- **OM-30**: Look into ways of querying using variables from the backend
- **OM-35**: Familiarize with CSS
- **OM-36**: Create the home page layout
- **OM-37:** Create the item page layout
- **OM-39**: Create search page layout