

# FUNCTIONAL SPECIFICATION

LANU FULL AUTO PROCESS

SUPRIVISOR:

Joseph Kehoe

(Student) - Andrew Bashorum

C00238900

## Contents

Project Description.....	2
User Interaction .....	2
System Use Case Diagram.....	4
Use Cases .....	5
Application (FURPS+) .....	9
Functionality .....	9
Usability .....	9
Reliability.....	9
Performance .....	10
Supportability.....	10
Metrics .....	10
Testing.....	10
Bibliography .....	10

## Project Description

This project will be a useful tool in automating the creation of an accurate model for a house given only its address as input. This will also be able to give feedback to the user on the extensions they may be able to build, or the extensions already built onto the inputted house.

It will be helpful for standard users who want to discover and visualize the possible extensions their house can get legally. Governments who have needs such as surveying houses in bulk or plotting their next move to solve the housing crisis. Councils such as Brent and Harlow use prop-tec tools like this one often. It is also useful for other companies such as real estate agents looking to buy houses which can be greatly expanded before selling them. Nothing like this proposed application currently exists.

## User Interaction

Dan is a developer. Dan has a portfolio of houses all based in London. Dan realizes that adding a rear dormer to a house of his will add 20% to its value. Dan uploads his portfolio to the Auto Lanu application. Within 20 minutes dan is given a report. The report tells Dan that half of his portfolio can legally add a rear dormer to their attic. Each house has an individual report detailing the size of each possible extension, the added square meter, the approximate cost of each extension, and the added value of each extension. An example of such output can be seen in Figure 1.

### Extensions




3D Model	Description	Add. Area	Total Area	Approx Cost	Add. Value
	<b>Max Option 1</b> Ground floor rear (PN), Ground floor side, Full width rear dormer <a href="#">Show more</a> <a href="#">Next Steps</a>	762 sq. ft.	1,740 sq. ft.	£142,000	£225,000
	<b>Max Option 2</b> Ground floor rear (PN), Ground floor side, Full width rear dormer <a href="#">Show more</a> <a href="#">Next Steps</a>	666 sq. ft.	1,644 sq. ft.	£124,000	£153,000
	<b>Ground Floor Option 1</b> Ground floor rear (PN), Ground floor side <a href="#">Show more</a> <a href="#">Next Steps</a>	437 sq. ft.	1,416 sq. ft.	£81,000	£109,000

Figure 1, Final Output of project

Sharron and Paul have a three-bedroom house. They have one child and one more on the way. They won't have space for their new-born but can't afford to buy a larger home. Paul quickly uploads his

address to the Lanu auto system, ticks the dormer box, and is told within 30 seconds if he can add a dormer to his attic. Paul is also given the same output seen in Figure 1. Paul can now contact a builder and give them the plans he received from the Lanu system. Fast-tracking the solution to his problem.

Borris works for Brent council (London Borough). The council committee has recently decided on a new housing scheme. They need to house another ten thousand people. They do not have the budget to build the new houses required. Instead, Borris decides that it would be more economical to add a large side extension onto existing council houses than to build brand new houses on newly bought land. For this side extension to be built a minimum 3-meter gap must exist between two houses. To check which council houses are suitable for this, Borris uploads a list of the other council houses in Brent to the Lanu system. Borris ticks the side extension box and submits. Within 6 hours Borris is emailed a report detailing the houses for which this is possible, each house containing a report like the one seen in Figure 1.

## System Use Case Diagram

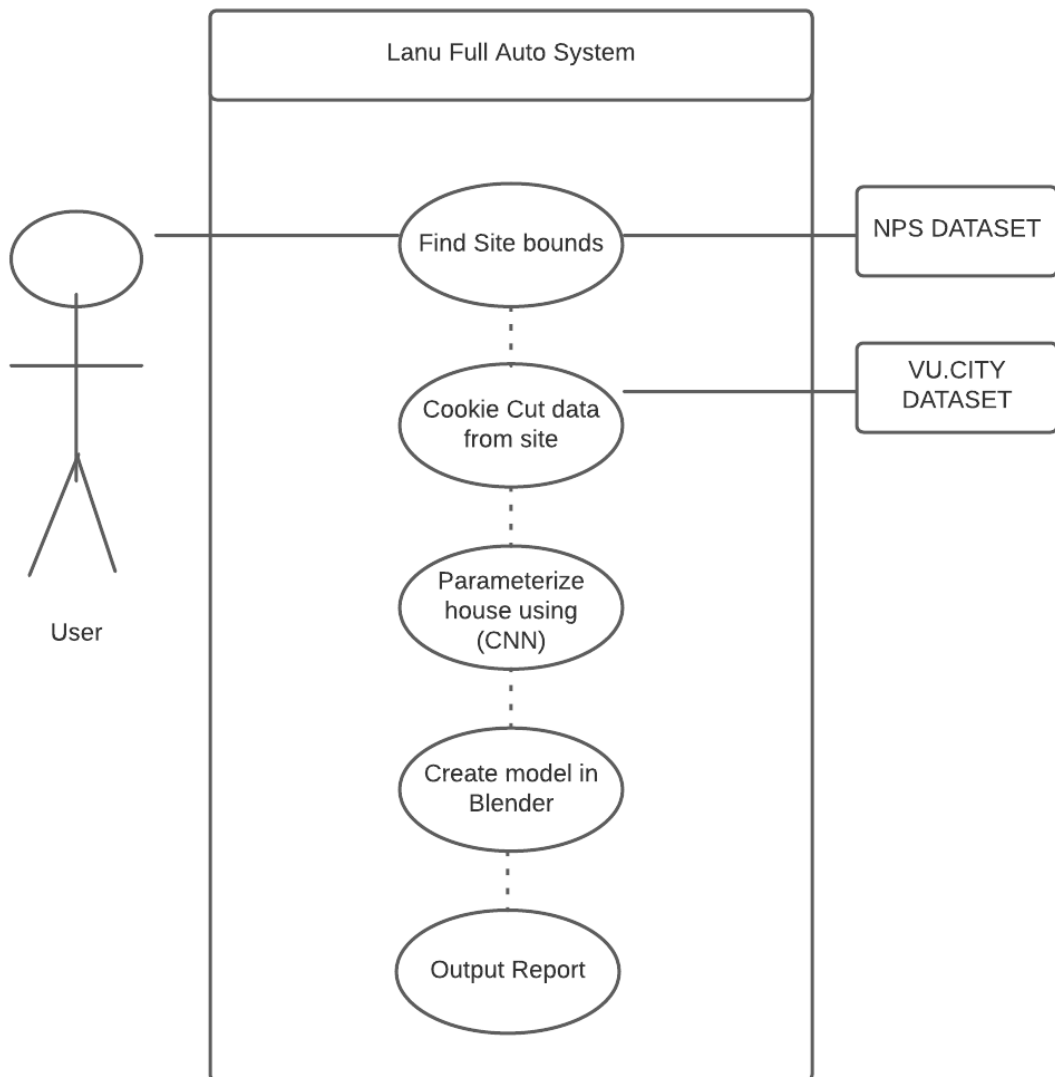


Figure 2, System Use Case Diagram

## Use Cases

<i>Name:</i> Lanu Full auto system	
<i>Actors</i>	User, NPS Dataset, VU.CITY dataset, CNN
<i>Description</i>	Outputs a report detailing possible extensions, 3D house models a full address
<i>Preconditions</i>	<ul style="list-style-type: none"> <li>Valid full address (including zip code) must in input</li> <li>Address must be in the UK</li> <li>Address must be within 1km squared of Lynmouth drive</li> <li>NPS Dataset must be initialized to Postgres database</li> </ul>
<i>Main success scenario</i>	<ol style="list-style-type: none"> <li>User inputs address.</li> <li>Site bounds is extracted from NPS dataset</li> <li>Raster RGB image created from VU.CITY dataset</li> <li>Convolutional neural network used on image to determine and parameterize the features of the RGB image.</li> <li>Model is created in Blender.</li> <li>Report is generated</li> </ol>
<i>Post Conditions</i>	<ul style="list-style-type: none"> <li>Multiple 3D models have been generated</li> <li>Report has been generated</li> </ul>
<i>Alternative</i>	<p>Not a valid address</p> <ol style="list-style-type: none"> <li>System outputs "Invalid address".</li> <li>System requests new address.</li> </ol> <p>Site from outside specific square kilometre of VU.CITY data</p> <ol style="list-style-type: none"> <li>System outputs "Invalid address".</li> <li>System requests new address.</li> </ol> <p>Features of image is not recognized machine learning algorithm.</p> <ol style="list-style-type: none"> <li>System saves incorrect values in house and site dictionaries.</li> <li>System generates incorrect model of house</li> </ol>

<i>Name:</i> Find site bounds	
<i>Actors</i>	User, NPS Dataset
<i>Description</i>	Outputs site bounds polygon given a full address
<i>Preconditions</i>	<ul style="list-style-type: none"> <li>Valid full address (including zip code) must in input</li> <li>Address must be in the UK</li> </ul>
<i>Main success scenario</i>	<ol style="list-style-type: none"> <li>User inputs address.</li> <li>Address is geocoded.</li> <li>NPS database is queried using value from step 2.</li> <li>Site bounds polygon stored in pickle file</li> </ol>
<i>Post Conditions</i>	<ul style="list-style-type: none"> <li>A pickle file with the site bounds relating to the input address has been stored.</li> <li>A site finder object also stored. This contains various variables such as the sites possible neighbouring sites.</li> </ul>
<i>Alternative</i>	<p>Not a valid address</p> <ol style="list-style-type: none"> <li>System outputs "Invalid address".</li> <li>System requests new address.</li> </ol> <p>Site Not found</p> <ol style="list-style-type: none"> <li>System outputs "Site not found".</li> <li>System requests new address.</li> </ol>

<i>Name:</i> Cookie cut data from site bounds	
<i>Actors</i>	VU.CITY dataset
<i>Description</i>	Uses site bounds to extract vectors from VU.CITY dataset
<i>Preconditions</i>	<ul style="list-style-type: none"> <li>An accurate site bounds has been found for the house</li> <li>Site is within square kilometre which the VU.CITY dataset covers.</li> </ul>

<i>Main success scenario</i>	<ol style="list-style-type: none"> <li>1. Site bounds pickle file is read by cookie cutter.</li> <li>2. VU.CITY data is queried using site bounds.</li> <li>3. Raster data is extracted from dataset.</li> <li>4. Raster data is transformed into RGB image</li> <li>5. RGB image is saved</li> <li>6. A link to the RGB image is saved into siteFinder object which is then saved into a pickle file.</li> </ol>
<i>Post Conditions</i>	<ul style="list-style-type: none"> <li>• An RGB image of the house is saved. With each colour detailing a direction.</li> <li>• SiteFinder object is updated with this new houses image.</li> </ul>
<i>Alternative</i>	<p>Site from outside specific square kilometre</p> <ol style="list-style-type: none"> <li>5. System outputs "Invalid address".</li> <li>6. System requests new address.</li> </ol>

*Name:* Parameterize house using (CNN)

<i>Actors</i>	User, RGB image dataset
<i>Description</i>	Takes RGB dataset as input. Using convolutional neural network to determine and parameterize the features of the raster data.
<i>Preconditions</i>	<ul style="list-style-type: none"> <li>• A large dataset of accurate RGB image of the raster data must have been created and stored.</li> <li>• System must be trained correctly on this dataset.</li> <li>• A new accurate RGB image must have been made correctly pertaining to the house in question</li> </ul>
<i>Main success scenario</i>	<ol style="list-style-type: none"> <li>1. The RGB image will input into the CNN system</li> <li>2. A decision tree method which asks specific questions about the image will occur.</li> <li>3. Each feature of the image will be classified and stripped.</li> <li>4. Now the RGB image will remain except without the last features which were found</li> <li>5. When a feature such (dormer or chimney) is discovered, its co-ordinates will be noted and saved into the site and house dictionaries.</li> </ol>
<i>Post Conditions</i>	The house dictionary will be updated with the values of the features found. This will allow for the generation of the Blender model.
<i>Alternative</i>	<p>Features of image is not recognized my machine learning algorithm.</p> <ol style="list-style-type: none"> <li>1. System saves incorrect values in house and site dictionaries.</li> <li>2. System generates incorrect model of house</li> </ol>



<i>Name:</i> <i>Blender model generation</i>	
<i>Actors</i>	House dictionary, Site dictionary, User
<i>Description</i>	Creates 3D blender model of the house using house, site dictionaries and blenders python scripting feature.
<i>Preconditions</i>	<ul style="list-style-type: none"> <li>Accurate values need to be saved into the dictionaries. This means the alternative of the Parameterize house using (CNN) use case cannot take place.</li> </ul>
<i>Main success scenario</i>	<ol style="list-style-type: none"> <li>Building blocks pertaining to the dimensions and coordinates of each, feature and main blocks of the house are extracted from the house dictionary.</li> <li>Blender's python scripting is utilized to automatically generate the model.</li> <li>The model is output in multiple formats. Such as an AR model, png, jpeg, a rotational gif.</li> </ol>
<i>Post Conditions</i>	The multiple forms of output will be saved and stored locally. Ready for to be used in the report.
<i>Alternative</i>	If inaccurate values were saved an inaccurate model will be created and ultimately displayed to the user.

<i>Name:</i> <i>Create Report</i>	
<i>Actors</i>	House dictionary, Site dictionary, Blender models, User
<i>Description</i>	Creates and outputs report
<i>Preconditions</i>	<ul style="list-style-type: none"> <li>Blender models generated</li> <li>Address must be in the UK</li> </ul>
<i>Main success scenario</i>	<ol style="list-style-type: none"> <li>Build cost and added cost calculated using blender model and postcode.</li> <li>Report Built using these values and Blender models.</li> <li>Data uploaded to Lanus Amazon S3 system</li> <li>Display S3 link to user</li> </ol>
<i>Post Conditions</i>	<ul style="list-style-type: none"> <li>Report generated and displayed to user.</li> <li>A site finder object also stored. This contains various variables such as the sites possible neighbouring sites.</li> </ul>
<i>Alternative</i>	<p>Not a valid address</p> <ol style="list-style-type: none"> <li>System outputs "Invalid address".</li> <li>System requests new address.</li> </ol> <p>Site Not found</p> <ol style="list-style-type: none"> <li>System outputs "Site not found".</li> <li>System requests new address.</li> </ol>

## Application (FURPS+)

“FURPS is a technique to validate the prioritised requirements after an understanding with client’s needs and necessities. The acronym FURPS is Functionality, Usability, Reliability, Performance, and Supportability, over a period of time and grave need raised to see the solution from more dimensions gave to emergence of FURPS+. This FURPS+ technique made the requirements classification to stress on understanding the different types of non-functional requirements more.” (Coepd, 2014)

### Functionality

The main features of this project are outlined below.

It should be noted that due to the VU.CITY dataset being limited to a square kilometre area, the functionality will only work for houses within that area. All the features require a user to input an address string that is suitable for instant geocoding. Meaning that it is already in googles street address format.

Main Features:

1. Automatic Generation of accurate house model.
2. Automatically determine if an inputted house already has a rear dormer.
3. Automatically determine if an inputted house can legally and realistically fit and build a rear dormer.
4. Automatically determine if there is enough side space to build a granny flat extension (Large side extension)

Functionality is the most import part of this project as the functionality seen in this project is completely original and hasn’t been completed before. This makes it fair important than the reliability or usability of the project. What makes the functionality of this project unique is the automation. No one has made an automatic model generation system quite like this before.

### Usability

Usability refers to the users experience such as experience and accessibility when using the user interface. This application should simply allow users tick a small collection of tick boxes and input a string detailing an address. If the address is in the format an error message should display. This system should work on a Mac OS system.

### Reliability

Reliability determines the accuracy and dependability of the application. If the inputted address is within the square kilometre outlined by the VU.CITY dataset.

- A model which is accurate to about 80% should be generated 99% of the time.
- The accuracy of the dormer determination should be 95%.
- The determination of the side gap should be 98%.
- The determination of a new dormer given the boroughs ruleset should be around 50%

## Performance

Performance refers to the systems start up time, response time, recovery time and the length of time it takes to return a result. Performance is somewhat important to the application, but it should be noted that the application is only a prototype and delays are expected. When an address is submitted, a result detailing the existence of a dormer should be shown within 5 minutes 99% of the time. An accurate model should be generated within 15 minutes 99% of the time. This longer time is due to possible time delays when using scripting in Blender to generate the model. As experience has shown some of these operations can delay a process.

## Supportability

Supportability is determined by testability and maintainability. This project which will only be a prototype will not be able to be installed on any device other than the developers. This is due to the heavy datasets needed to be stored locally on any machine which runs the application. This will change in the future when these datasets will be hosted on the cloud.

## Metrics

The success of this project will be gauged on the following parameters:

1. The application should take an address as input
2. The application should return a link to a form like the one seen in Figure 1.
3. The application should be able to determine if a dormer is in the attic of the inputted house 75% of the time.
4. The application should generate a model of the house 60% of the time

## Testing

All parameters can only be tested on Andrew's Mac Mini. Using addresses from the houses within the square kilometre of VU.CITY data. To test each parameter the application will be run with different inputs. These following inputs should be tested.

- A semi-detached house with a dormer.
- A terraced house without a dormer.
- A detached house with a very large side gap. (4+ meters)
- A detached house without a very large side gap.

Each outcome will be tested at least 20 times to get accurate results and percentages to test against the projects metrics.

## Bibliography

Coepd, B. A. T. i. H. -, 2014. s.l.: s.n.