HACETTEPE UNIVERSITY
COMPUTER ENGINEERING DEPARTMENT

UNDERGRADUATE PROJECT FINAL REPORT

Project Name	Report Date
WEB PAGE COMPLEXITY STUDY	08.01.2021

Student Number(s)	Student Name(s)
21627543 21626901 21627802	Ece OMURTAY Deniz Ece AKTAŞ Ömer Bilal YAY
Supervisor(s)	**Company Representative(s)**
Murat AYDOS	

Project Coordinator	Report Approval
Date: _____	☐ Yes ☐ No If no, rational of rejection: _____

Project Video Youtube Link
https://youtu.be/UVoamZPDMT4

A. TECHNICAL RESULTS

ABSTRACT

Classification of the visual complexity level of web pages through CNN based learning methods.

The visual appearance of web pages affects the way a user will interact with the web page contents. The layout which shapes the visual characteristics of overall appearance is composed of various visual components such as texts, images, form elements, and white spaces. Moreover, the placement and visual presentation of these components influence the perceived visual complexity. At this point, the literature suggests that the visual complexity of a web page impacts the cognitive load and effort of the users when they interact with web pages.

Keywords: CNN, Visual Complexity, Machine Learning

I. INTRODUCTION

This project investigates the use of convolution neural network-based deep learning methods for understanding the visual complexity level of the web pages. To do this, we first of gathered a base case by making a website and gathering user scores that provide an insight of how visually complex a website is from the screenshots we provided. Afterwards we sectored the base case and made a database and then we found the feature vectors of websites and optimized the vectors and finally we are working with different machine learning methods to classify the level of complexity of web pages based on dimensionally reduced vectors.

In this report we will summarize all the work we have done and the experiences we learned.

II. BACKGROUND

Our primary focus during this project was to implement the survey website and get the dataset needed and to do the necessary machine learning steps on the dataset. The reason to choose this project was mostly to learnt new information and use what we learn and gain experiences.

To run this project we only had windows as our platform Linux would have been ideal.

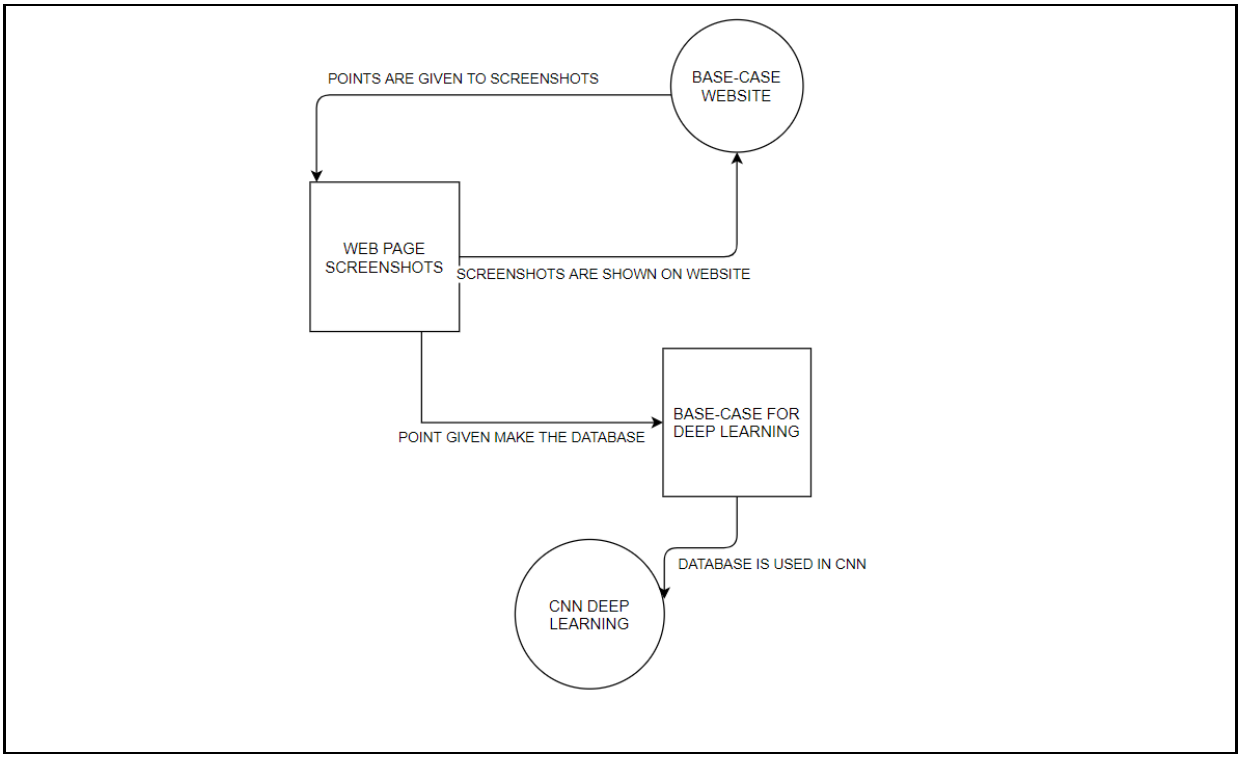
The desired outcome is to have the machine learning algorithm that can measure the complexity of a website from a screenshot.

III. RELATED WORK

There are many studies on visual complexities and aesthetics of web sites and their impacts on user's reactions to these components and these research papers and results have been used to make better websites. There are even some websites like Wix that provide aesthetic and visual choices for websites designs.

Our supervisors provided us with some research papers to read and the papers are; visual complexity and aesthetic perception of web pages[1], layout-based computation of web page similarity ranks[2], visual aesthetics of e-commerce websites an eye tracking approach[3], brand recognition of Phishing web pages via Global Image Descriptors[4].

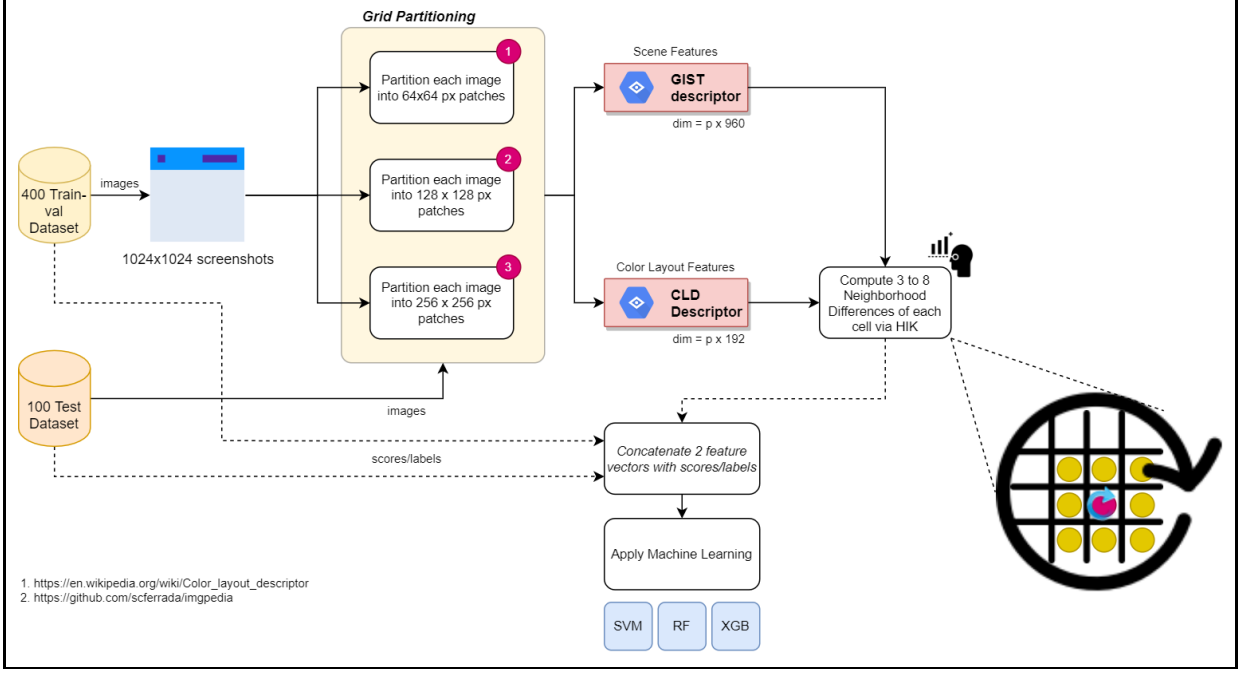
IV. METHOD



V. TECHNICAL DESIGN AND CONFIGURATION

In this project, we used screenshots of website images as input. To be precise machine learning part of the project takes 1024 x 1024 sized 400 screenshots of the websites that we constructed the survey on, for the training of machine learning. For testing the program takes 1024 x 1024 sized 100 different screenshots of the websites.

We provide both .py and .pynb files of the same code to run. Normally we would use a terminal and that would be enough to run .py file but because the programs gave problems with windows' terminal we used Jupyter Notebook to run our codes so we also used .pynb files as well.



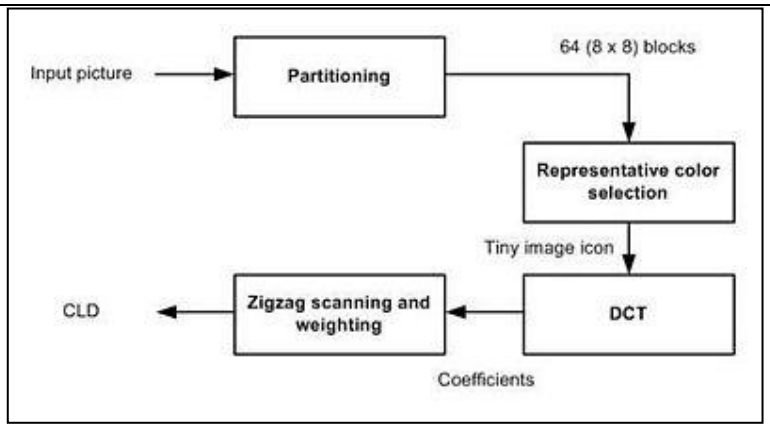
VI. PROJECT IMPLEMENTATION

Firstly, we made a website to gather user scores by using ASP.NET and transfer into database by using SQL. Then, we extracted the CLD and HOG features of each image. After that, we created 3 Machine Learning models which are Random Forest, SVM and XGBoost to train. Finally, we tested our models by using our test data.

In feature extraction, we split our images into fixed sized px grids which are 256x256, 128x128 64x64. We gave each patch separately into CLD and HOG descriptors. (The visual representation of CLD is provided in the image below.) There are 4 main steps in CLD computation. We first divide the image patch into 64 blocks (8x8 matrix) as provided in the paper. Then, we selected the representative (average) color of each block. We changed the RGB color space into YCbCr (which is luminance Y, blue and red chrominance Cb – Cr) color space to compute DCT. In final step, we did zigzag scanning to group the low frequency coefficients of 8x8 matrix. Then, we calculated the neighborhood differences of CLD features of each patch by using Histogram Intersection Kernel (HIK). In HOG descriptor, we gave each patch into descriptor and concatenate them to find the HOG feature of the image. Then also, we calculated the neighborhood differences of these HOG features via HIK. Finally, we concatenate these arrays and created the feature vector of an image. (We used 32x32 as cell size in HOG)

Since this is a classification problem, our aim is to find discrete class label output for an image. In order to classify the images, we gave labels to mark values of images. We specified range values as follows: 5 – 20 (very basic): 1, 20 – 40 (basic): 2, 40 – 60 (moderate): 3, 60 – 80 (complex): 4, 80 – 100 (very complex): 5. Thus, we can say that how complex a website is by looking at these labels.

After doing this, we gave both feature vector and ground truth into models to train.

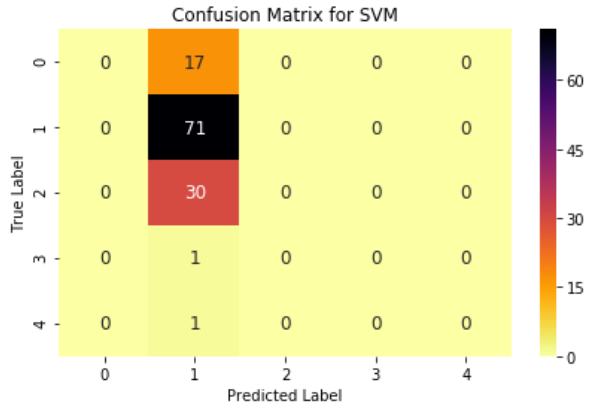


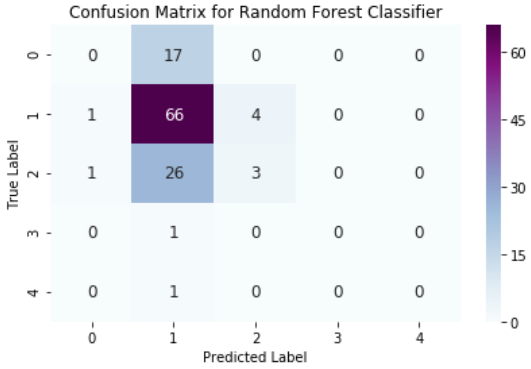
In [40]:
```
label_test = []

for path in imgs_test:

    img_name = path.split("\\")[1]
    mean_val = test_labels[test_labels["ImageName"] == img_name]["Mark"].values[0]

    if 5 < mean_val < 20:
        label_test.append(1)
    elif 20 < mean_val < 40:
        label_test.append(2)
    elif 40 < mean_val < 60:
        label_test.append(3)
    elif 60 < mean_val < 80:
        label_test.append(4)
    else:
        label_test.append(5)


test["ImageName"] = [p.split("\\")[1] for p in imgs_test]
test["Label"] = label_test

for i in range(len(imgs_test)):
    test.at[i, 'Features'] = test_data[i]
```

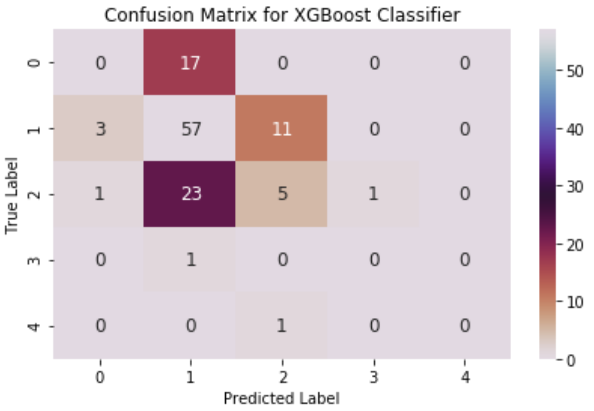
```python
def grid_partitioning(image, patch_size):
    patches = []
    w, h, _ = image.shape
    for x in range(0, h, patch_size):
        for y in range(0, w, patch_size):
            grid = image[x: x + patch_size, y:y + patch_size]
            patches.append(grid)

    return patches


# CLD
def representative_color(grid, patch_size):  # calculates the representative color of given patch(grid)

    blocks = np.zeros((8, 8, 3))
    step = int(patch_size / 8)

    for r in range(8):  # divided into 64 blocks
        for c in range(8):
            block = grid[r: r + step, c:c + step]
            avg_color = np.mean(block, axis=(0, 1))
            avg_color = np.uint8(avg_color)
            blocks[r, c, :] = avg_color

    return blocks


def compute_dct(blocks):
    im = cv2.cvtColor(np.array(blocks, dtype=np.uint8), cv2.COLOR_BGR2YCR_CB)
    y, cr, cb = cv2.split(im)
    dct_y = cv2.dct(np.float32(y))
    dct_cb = cv2.dct(np.float32(cb))
    dct_cr = cv2.dct(np.float32(cr))

    return dct_y, dct_cb, dct_cr
```

```python
def hik(array, cols):
    hiks = []
    for r in range(cols):
        for c in range(cols):
            nbrs = neighbors(r, c)  # returns indices of neighbors
            hik = 0
            for idx in nbrs:
                x, y = idx
                hik += histogram_intersection(array[r][c], array[x][y])
            hiks.append(hik / len(nbrs))
    return hiks


cld_hiks = []
hog_hiks = []

for img in imgs:

    img = cv2.imread(img)
    grids = grid_partitioning(img, 64)

    img_cld = []
    img_hog = []
    for grid in grids:              # grid == patch
        blocks = representative_color(grid, 64)
        patch = zigzag(*compute_dct(blocks))
        img_cld.append(patch)

        fd, hog_image = hog_descriptor(grid, 32)
        img_hog.append(fd)

    img_hog = np.concatenate(img_hog)
    img_hog = np.array(
        [img_hog[i:i + col_num] for i in range(0, len(img_hog), col_num)])  # makes 2d array to find neighbors
    hog_hiks.append(hik(img_hog, col_num))
    img_cld = np.array(
        [img_cld[i:i + col_num] for i in range(0, len(img_cld), col_num)])  # makes 2d array to find neighbors
    cld_hiks.append(hik(img_cld, col_num))

stacked = np.concatenate((hog_hiks, cld_hiks), axis=1)
np.savetxt('train_data64.csv', stacked, delimiter=',')
```

VALIDATION AND RESULTS







Test accuracy for SVM 64x64 : 0.54

Test accuracy for XGBoost 128x128 : 0.48

Test accuracy for Random Forest 256x256 : 0.49

Images below are the classification reports of SVM classifier for 128x128 and 256x256:

	precision	recall	f1-score	support
1	0.00	0.00	0.00	17
2	0.59	1.00	0.74	71
3	0.00	0.00	0.00	30
4	0.00	0.00	0.00	1
5	0.00	0.00	0.00	1
accuracy			0.59	120
macro avg	0.12	0.20	0.15	120
weighted avg	0.35	0.59	0.44	120

Accuracy: 0.5916666666666667

	precision	recall	f1-score	support
1	0.00	0.00	0.00	17
2	0.59	1.00	0.74	71
3	0.00	0.00	0.00	30
4	0.00	0.00	0.00	1
5	0.00	0.00	0.00	1
accuracy			0.59	120
macro avg	0.12	0.20	0.15	120
weighted avg	0.35	0.59	0.44	120

Accuracy: 0.5916666666666667

As a result, SVM is the best classifier for this project when compared with others. When we analyze the f1-scores and confusion matrices, we can see that all models predict the “very basic” class (label 1) more accurate but other classes cannot predicted by models. "very basic" (label 1) class makes up most of our data. So, we reached nearly 50% accuracy in both train and test.

CONTRIBUTION(S) TO INDUSTRY AND ECONOMY

The potential contribution to the industry and the economy of our study is that if we could find the potential complexity levels of websites, in the future while designing a website the balance of complex and simple design could be found more easily and so that the user of the website would have a better experience with the website so that it would be used more frequently. This could affect the economy especially if the result are used on shopping and retail websites.

INNOVATIVE ASPECTS

There are already works and researches done about visual complexities of websites but the difference of our study is that we will find the level of complexity of web pages specifically based on dimensionally reduced vectors using machine learning and deep learning.

REFERENCES

[1]. Michailidou, Eleni & Harper, Simon & Bechhofer, Sean. (2008). Visual complexity and aesthetic perception of web pages. Proceedings of the 26th Annual ACM International Conference on Design of Communication. 215-224. 10.1145/1456536.1456581.

[2]. Bozkir, Ahmet & Sezer, Ebru. (2017). Layout-based computation of web page similarity ranks. International Journal of Human-Computer Studies. 110. 10.1016/j.ijhcs.2017.10.008.

[3]. Pappas, Ilias & Sharma, Kshitij & Mikalef, Patrick & Giannakos, Michail. (2018). Visual Aesthetics of E-Commerce Websites: An Eye-Tracking Approach. 10.24251/HICSS.2018.035.

[4]. Eroglu, Esra & Bozkir, Ahmet & Aydos, Murat. (2019). Brand Recognition of Phishing Web Pages via Global Image Descriptors. European Journal of Science and Technology. 436-443. 10.31590/ejosat.638397.

B. PROJECT RESULTS

I. CHANGES TO PROJECT PLAN

The change we made are; we had to delay machine learning part of the project for one month because the website implementation and data collecting parts took more time than anticipated and also in machine learning part we were going to use GIST descriptor to get the features but any of the team members doesn't have a Linux device so the computers did not run GIST descriptor instead we planned to use HOG descriptor.

II. PROJECT MILESTONES AND OBJECTIVES

Milestone #	Primary Objective	Due Date	Project Deliverable (if any)	Milestone Achieved?
1.	To finish the website for collecting data for base-case.	November 2020	A working website exists	Yes
2.	To get the database we will be using from the user inputs on the website and dividing them into labels.	December 2020	Database that will be used in machine learning and deep learning.	Yes
3.	Project process evaluation and project process report delivery	December 2020	Project process reports will be submitted.	Yes
4.	To work on VGG and Resnet types of machine learnings with obtained labels in order to extract feature vectors	January 2021	Feature vector that is extracted from websites.	Yes
5.	To apply a supervised technique named UMAP to reduce the dimensions in a supervised manner	January 2021	Optimized result is acquired.	Yes
6.	Final project delivery and presentations	January 2021	Final project reports and presentations will be delivered.	Yes

III. PROJECT PRACTICES AND MEASURES

Task #	Task Description	Responsibility	Start Date	Finish Date	Success Criteria	Task Succeeded ?
1.	To get the base-case webpage development	Ece Omurtay Deniz Ece Aktaş Ömer Bilal Yay	October 2020	End of November 2020	Having a working website.	Yes
2.	Implementing the point acquired from website into database	Ömer Bilal Yay	October 2020	December 2020	Having a database that is divided into labels.	Yes
3.	Using the data base and implementing machine learning and testing.	Ece Omurtay Deniz Ece Aktaş Ömer Bilal Yay	December 2020	January 2021	Getting the optimal results from the machine learning algorithms.	Yes

Team Member	Task # Under Responsibility	Description of the Work Done
Ece Omurtay	1, 3	Did research on the topic. Worked with asp.net in Microsoft Visual Studio to implement a survey website for the purpose of getting base-case inputs for machine learning part. Spread the word about the survey and encouraged people to take the survey. Grid partitioned the dataset images and did CLD (color layer descriptor) on the partitioned datasets. Did the HOG descriptor and machine learning algorithms.
Deniz Ece Aktaş	1, 3	Did research on the topic. Worked with asp.net in Microsoft Visual Studio to implement a survey website for the purpose of getting base-case inputs for machine learning part. Spread the word about the survey and encouraged people to take the survey. Grid partitioned the dataset images. Did the learning algorithms. Wrote the reports and presentation.
Ömer Bilal Yay	1, 2, 3	Did research on the topic. Implemented the database with SQL and afterwards helped with the implementation of the website by assisting with asp.net. Spread the word about the survey and encouraged people to take the survey. Grid partitioned the dataset images and Did the learning algorithms. Made the video required.

IV. PROJECT BUDGET

We used our own computers that we have and as for software we used Visual Studio, PyCharm and SQL but we used the student packets of these software so we didn't have any expenses and also because of the COVID-19 pandemic we worked remotely so we did not have any commuting expenses. As for income we don't have any income that is related to design project.

V. PROJECT RISKS

Risk Item #	Description	Probability	Effect	Did It Happen?	How did you handle its occurrence if happened? (Plan-B)
1.	If the user points are given in the start of the project is given with errors or given not seriously.	Possible	Results of study would be wrong	Yes	We implemented a confirmation system for points by taking five backup points and comparing them to other points.
2.	The machine learning methods we will try could give not the best results	Possible	Results wouldn't be optimal	No	We are using more than one machine learning methods to see which one gives the best results.
3.	The hardware that we have (windows pc) doesn't run the programs needed.	Possible	The descriptor wouldn't work	Yes	The GIST descriptor we planned to use didn't work on windows pcs and we do not possess or have access to a Linux device so we decided to use HOG descriptor instead.

VI. SELF EVALUATION

I find my teams accomplishments successful because even if we were a little slow to finish our tasks we still did most of them correctly. The biggest problem we had was about time management and not the content itself. As a team we worked well together and did the most of the work together but because of COVID-19 we couldn't meet up so we only met up online and this brought up some issues like if someone's internet was down we had to delay our meeting. But other than these small issues that we had no way of solving, we did not experience any problems and were kind with each other.

As a team all of the material in this project was new knowledge to us, so we learnt a lot of information about a lot of fields like computer vision, machine learning, web development and database management but because we had to learn a lot of things from scratch we were a little slow during implementation.

Especially during COVID-19 we experienced a lot of issues; the issues related to work are, if our Windows computers didn't work we didn't have any other alternatives, we couldn't find any other devices and had to change one of the descriptors we used.

VII. LESSONS LEARNED

If I were to do this project from the start, I would try to do a better time management and rather than focusing on survey website implementation I would focus on machine learning.

Feedback:

BBM419 Design Project-I course caused a lot of issues to the students because at the start of the semester we expected a class where we would be informed at and because we weren't informed we didn't know how to and why to connect to teachers and select a design project we learnt these from other students not the school. Also the website was updated but it was updated some time into the semester and the information on the website hasn't really been updated again because this year BBM479 class was taken out and BBM419 class came instead of it we expected a better explanation of what changed and what we had to do. As an example because the example proposal on the website wasn't updated, after sending the proposal we had to add some section to it because apparently new sections were added but we were informed after we submitted our proposals.