The University of Edinburgh

INFR11172

Software Development

Planning and Design of MyUniCompass

Group 11
Authors:
B118857 B128883
B136013 B140222
B143852

Course Organiser:
Dr Grant Alistair

February 13, 2019

THE UNIVERSITY *of* EDINBURGH

Contents

1 Introduction

This assignment requires us to create a research engine, MyUniCompass, described as plan 1 in the given problem (Grant 2019, Software Development Coursework), that offers prospective students an avenue to search a range of universities through a wide-ranging set of criteria. This set of criteria will include degree programmes on offer, graduation rates, employment chances, quality of life measures and teaching excellence.

The following report will show how we break the whole project into its main components and discuss them in details. The components consist of the project plan, the requirement engineering, system and user interface designs, the data management plan and the risk analysis report. The final section will summarise all considerable future development of MyUniCompass.

2 Project Planning

2.1 Team Structure

The team consists of five postgraduate students pursuing their MSc in High Performance Computing at the University of Edinburgh. The team members and their main roles are stated in Table 1 in the form of a RACI Matrix (Kennedy 2019, week 4, p.10). This is a simple way to visualize the roles and degree of contribution for each member of the team. The involvement is described by four letters which are interpreted as follow: A (Accountable), R (Responsible), C (Consulted) and I (Informed).

Role	Nikolaos Xenakis	Le Yang	Alexey Riepenhausen	Yeow Tong Yeo	Xiaoyan Ma
Project Manager	A, R	I	I	I	C
System Designer	R	I	C	I	A, R
UI Designer	A, R	C	R	C	C
Business Analyst	I	R	I	R	A, R
Tester	C	A, R	C	R	R
Documentation Specialist	R	R	R	A, R	R
Risk Officer	C	C	A, R	C	C
Software Developer	A, R	C	R	C	C
User Liaison	C	R	C	R	A, R

Table 1: RACI Matrix

2.2 Work Assignment Process

The assignment of each task has been made with respect to the background expertise, job skills and preferences of each group member (Kennedy 2019, week 4, p.3). For example, Xenakis Nikolaos worked as a Software Project Manager, so the project planning and team management parts are more suitable for him. Alexey Riepenhausen, have been involved in UI design projects, so he is responsible for the UI Design of the application. Le Yang and Yeow Tong Yeo are interested in the data management domain, investigating and gathering publicly available datasets for the need of the software. Yeow Tong Yeo is also experienced in LaTeX and will manage and overlook the documentation. Xiaoyan Ma is experienced in business analysis so requirements engineering would be a perfect fit for her background. It is worth mentioning that all of the workloads have been shared by the team members as equal as possible. For example, every team member contributes to each task by developing, designing or providing useful comments through the meeting discussions.

2.3 Communication Process

In order to design and develop the system according to the plan, intra-team communication is absolutely essential. Arranging fixed meetings throughout the week and defining the means of the communication process is vital. Also, this process contains a lot of obstacles and unexpected events. This is why developers invested a lot of time trying to find the most suitable time for our meetings. As a result, a decision is made to meet twice per week in fixed meetings when all of the team members are available in order to discuss the project tasks. These tasks include what has been done during the last week, what have to be done and what are the issues that need be resolved. This proved to be very useful because a lot of questions and argues are solved through that process.

Other than the arranged meetings, the team members use Slack communication platform as a mean for sharing ideas and files in the team. Emails, as a more formal way of communication. Person to person meetings in the Bayes Centre for urgent issues that need immediate handling.

2.4 Development Model

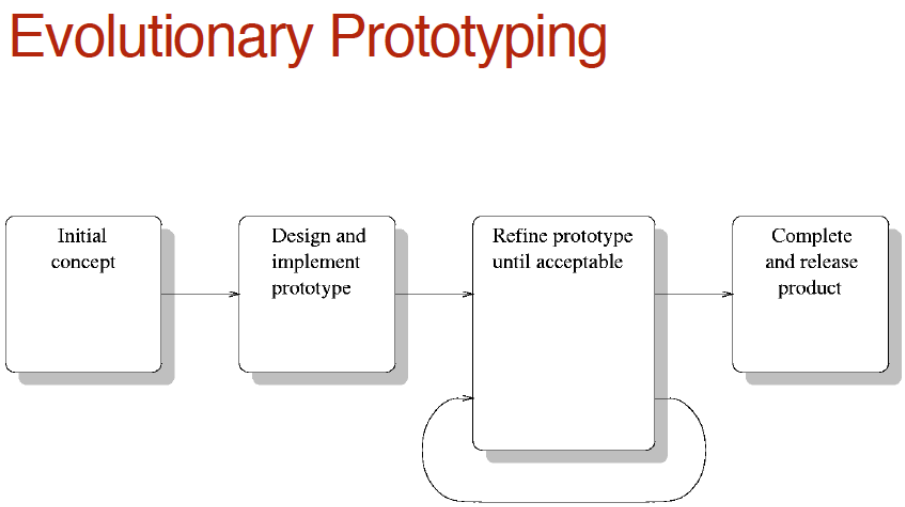


Figure 1: Evolutionary Prototyping

In terms of the development model, the team chose to follow the Evolutionary Prototyping (Kavoussanakis 2019, week 3, p.25) which is illustrated in Figure 1. This approach helps us improve our prototype by constantly resolving the project requirements. Step by step features have been added in order to meet our final design. This decision made based on the fact that the project description was a simple and abstract statement. The team members were constantly discussing and analyzing the requirements of the application with the course organizer in order to have a better understanding of the problem that the team was trying to solve. Each time, they received feedback on the current design, they constantly were modifying the prototype, until it meets the specifications. As a result, the evolutionary prototyping model suits our needs in order to end up to our final product.

2.5 Work Plan

A Work Plan is needed in order to identify the time and resources cost for each task, in order to achieve accurate project planning. This section contains the Tasks Decomposition for the main objectives that the project includes. Overall, the team is able to work on the project from 21 January to 29 March 2019, which are 10 weeks. Assuming that they work only on weekdays, 5 days per week, each one has 50 days available. Of course, there are many limitations. In general, each student attends to about 5 courses this semester, so he has 10 days to work on this project. Taking into account the available time in the day, an estimation says that he can invest 2 hours per day for coursework. As a result, each member can invest 20 hours for this project, this means 100 hours in total, for all of the members.

These hours equal roughly to the estimation time for all of the tasks. In Table 2 each important project task is presented in separate rows, including all the necessary information from the management's perspective. The time estimation for a task to complete has been evaluated based on the complexity, the potential risks that may occur and severity of the tasks as well as the experience of the team members on similar problems that they have faced in the past. The tasks that have not completed until this point, they have not actual time so they are measured as TBC (to be confirmed).

Task	Estimated Time (h)	Actual Time (h)	Resources (Members)	Description
Communication Tools	1	1	1	Setting the inter-team communication methods
Team Organization and Roles	3	4	5	Setting the roles and responsibilities of each team member
Concept Discussion	4	3	1	Discussing about the problem
Technology Investigation	4	4	1	Finding the appropriate technologies, languages, frameworks and platforms as the development environment
Requirements Engineering	11	10	2	Gathering and analysing the requirements of the project
Risk Analysis	6	7	1	Identifying and quantifying the risks
Set up Repository	1	1	1	Set up the project repository
System Design	7	8	2	Analysis and Design of the system
UI Design	6	7	2	Design of the User Interface
Design and Discover Data Sets	7	8	2	Design the format of the Data and discover the available Data Sets
Documentation Part 1	6	7	1	Write the Documentation Part 1
Build Development Environment	3	TBC	1	Build the platform that the prototype will be implemented
Software Template	5	TBC	1	Build the main components and the outline of the prototype
Prototype Implementation	13	TBC	2	Development of the working prototype
Usability Evaluation Planing	4	TBC	2	Plan how clients use the product
Prototype Evaluation	5	TBC	3	Testing and evaluation to confirm that the product work as it is supposed to
Usability Analysis	4	TBC	2	Analyze the user need to improve the usability experience
Documentation part 2	5	TBC	3	Write the Documentation Part 2
Total	96	TBC	5	

Table 2: Task Decomposition Table

Gantt Chart A Gantt chart has been generated in order to visualize the project plan and provide a better understanding of the dependencies of the tasks, as shown in Figure 2. The chart describes the time effort for all of the basic project tasks. Each task contains specific information about the team members that are involved and the estimated time that is needed. Based on the fact that most of the tasks cannot be done concurrently there are arrows that represent the dependencies across the tasks of the workflow. The chart has been designed thanks to `https://prod.teamgantt.com` web application. It worth to mention that some tasks may require for example 4 hours to complete but allocate 2 days in the Gantt chart. This happens because these tasks have been distributed through different days summing up to 4 work hours.

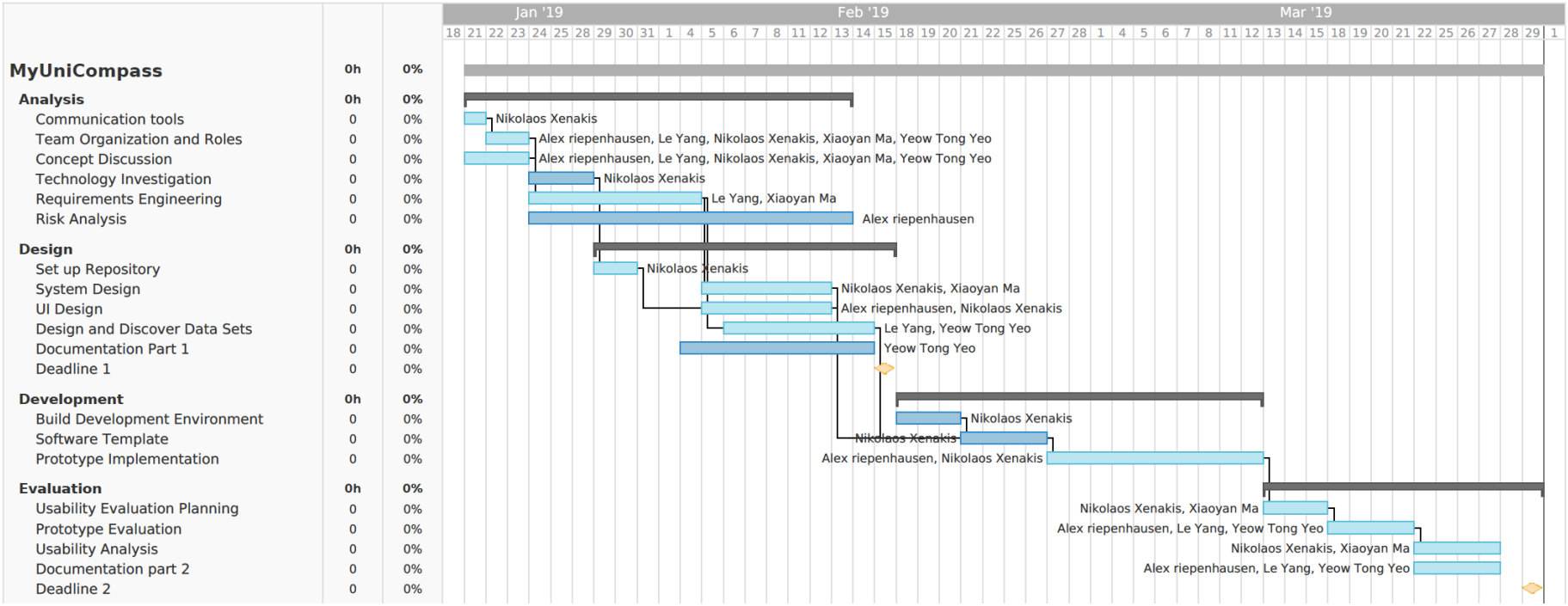


Figure 2: Gantt chart of Project Plan

3 Requirement Engineering

3.1 Requirement Gathering Strategy

In order to obtain the sufficient system requirements, the requirements are gathered via the following method: interview the domain expert and end users (in this case, they are Mr Alistair Grant), research the existing documentation (coursework requirement) and research the existing systems.

3.2 Environment

Domain: Given the information provided by the coursework, this is a search engine that focuses on providing university information based on the different search criteria.

Target Users: The primary target users are people who are wishing to attend a university, the supplemental groups would be advisors, teachers, and parents.

Competing system: There are similar online applications existing. But those have either very limited searching keyword or too much irrelevant information. `TOPUNIVERSITIES` (https://www.topuniversities.com/) is one of the most commonly used websites for searching for higher education opportunities. Only three options – Study level, Subject of interest and Study destination can be found in the child page of "Compare universities worldwide". Another application is `UCAS` (https://www.ucas.com/), which provides numerous and jumbled information that can cause distraction and confusion while browsing through it. Our research application will focus on providing detailed information only about finding a suitable university that fit in users' criteria.

3.3 System Requirements

Actors: in large there are two main actors interacting with this system: end users, prospect undergraduates and postgraduates, parents, school advisors, and the general public, and server where all the historical and current information stored at. There are three main requirements: search for universities' information based on the single search criterion, search for universities' information based on the multiple search criterion (advanced search), compare the universities' performance based on the keywords or selections.

A list of system requirements are presented in the below tables, all the requirements are grouped using MoSCoW method:

Priority	Functional Requirements Description
M	[FR1] Connect to internal servers where historical university information saved.
M	[FR2] End users can input the search criteria.
M	[FR3] Search the university information throughout the database based on the search criterion.
M	[FR4] Display the search result on screen.
M	[FR5] Connect to an external network.
M	[FR6] Display the search result on screen.
M	[FR7] Connect to an external network.
S	[FR8] Display no information found and provide guidance to users for obtaining better search results.
S	[FR9] Rank the search results by different criteria.
S	[FR10] Filter the search results by different criteria.
S	[FR11] Click on a search result to view the university full information.
W	[FR12] Print out the research results (a consolidation of print function throughout the system)
W	[FR13] Email the search results (a consolidation of email function throughout the system)
W	[FR14] Save the research finding in various file formats in a designated location (a consolidation of save function throughout the system)
W	[FR15] Support disabled end user to access the app.
C	[FR16] End users could have their personal account.

Table 3: List of functional requirements (FR)

Priority	Non-functional Requirements Description
M	[NR1] The user interface must be a graphic user interface.
M	[NR2] The system must run on a number of different types of platforms: laptop, desktop, and tablet.
M	[NR3] The system must be accessible via a website.
M	[NR4] Must be available at all time.
M	[NR5] Speed - waiting time no more than 30-45 second per command.
M	[NR6] The system need to have essential security in place to protect the database.
M	[NR7] Sufficient data storage.
M	[NR8] If there are end user personal account, user information must be secured.
C	[NR9] The system could be accessed via an application.
C	[NR10] Warning message for typing error.

Table 4: List of non-functional requirements (NR)

Please note that where M stands for must have, S stands for should have, W stands for would like to have, C stands for could have.

4 System Design

4.1 High Level System Architecture

In this section the high level system architecture is presented, given that this is a database driven web application, the system is designed as shown in the diagram below:

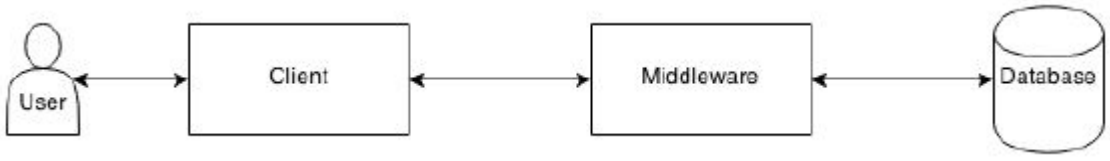


Figure 3: High Level System Architecture

The above high level system diagram shows the end users would only interact with the Client device, the system front end. In the case of this project, it would be a web browser. The Client device then interacts with the Database, the system back end through Middleware. In the Middleware, a number of servers including Domain Name Servers, Load Balancing Servers, Web Servers, Application Servers, and Caching Servers, etc., would be housed in here. For our application, we would predominately focus on the developing of the application server. A database server would be employed to handle the data management side of the application. An Entity Relationship diagram will be used to present how database tables are defined and populated.

4.2 System Architecture Design

To implement this system, a Model-View-Controller design pattern is used. The architecture is shown below:

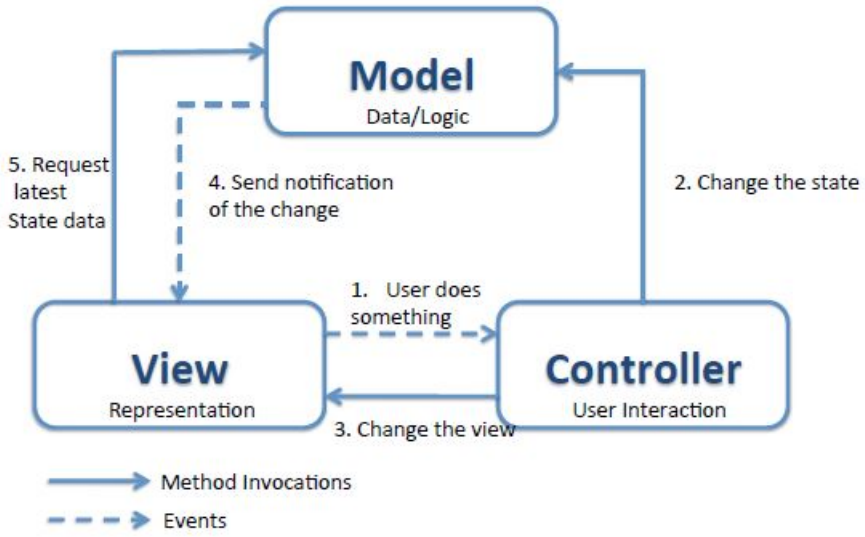


Figure 4: Model-View-Controller Architecture

As shown in Figure 4, the Model layer represents application and domain logic, it notifies View layer when it changes and enables View layer to query the model, it also allows the Controller layer to access the application functionalities encapsulated by the model.

View layer is tasked to visually represent the model and acts as a presentation filter, it specifies how the model data should be presented. The View layer is attached to the Model layer and gets the data necessary for the presentation from the model by asking questions, i.e., sending requests. Meanwhile, when the model changes, the view must update its presentation. This layer is also responsible for forwarding user request/gestures to the controller.

Controller layer defines the application behaviour, controller layer is the link between a user and the system, it interprets users requests/gestures and maps them into actions for the Model layer to perform and arranges for the relevant views to present themselves in appropriate places on the screen.

4.3 Implementation Path

To implement the Model-View-Controller architecture, we present our implementation path in terms of System Components and Technology Breakdown.

4.3.1 Component Details

At the design stage, it plans to have the following components as shown in the diagram:

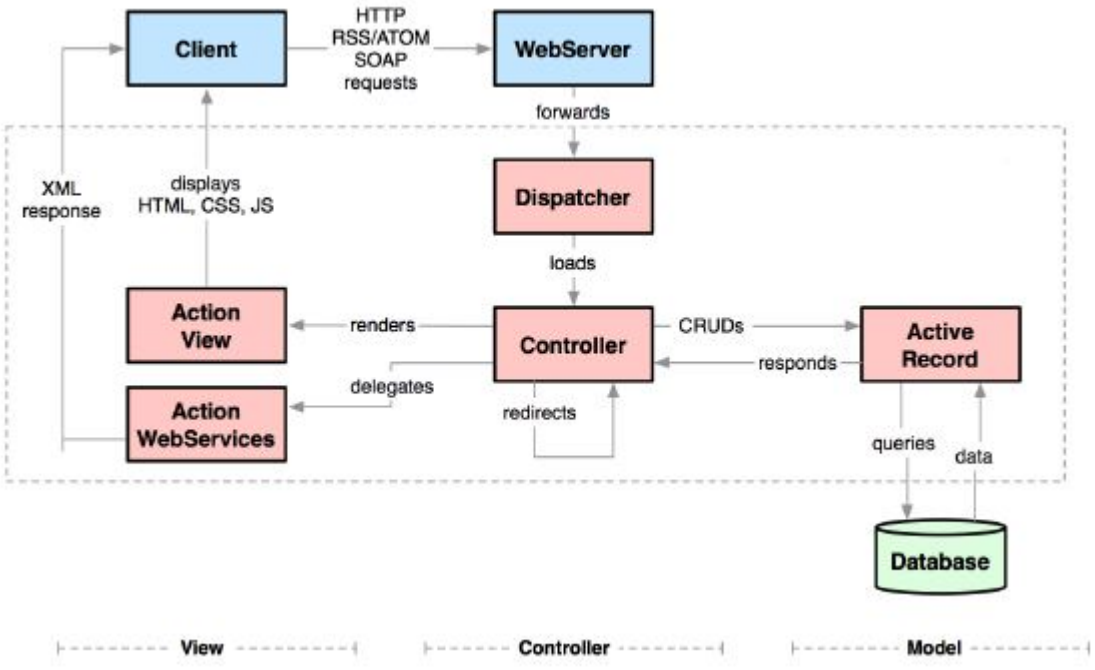


Figure 5: System Components

The above Figure 5 shows the components that are designed to work together, according to the system architecture. Note that both Client and WebServer belong to the front-end system, the components that the Middleware layer contains are coloured in pink, and the database is located in the system back end.

In the model layer, the Database would be used to store the historical university information, Active Record would be tasked to interact with Controller and the Database. The

Model layer contains the codes that operate on the application data, for example, any actions that wanted to be executed on the raw data must go through this layer.

The Controller layer acts as the orchestrator of our application. It controls the flow of the program as seen in Figure 5. It receives user commands, processes it and then contacts the Model layer, and finally instructs the View to display appropriately to the user. In detail, the WebServer is tasked to receive the requests from Client devices then forward them to Dispatcher which then process and load the requests to the Controller. Controller will act accordingly based on the requests.

The View layer is the presentation layer, it defines how our web page should look like to the end users, how the application presents data, or how a user can submit certain instructions to be executed by the application. In the view layer, Client devices interact with the end users, it sends requests to the WebServer and displays the returned data styled and formatted by Action View and Action WebServices to the end users.

4.3.2 Technology Breakdown

The project is a data-driven information provider platform. This means that the key decisions are to be accessible from a variety of machines without extra effort by the user. Having this in mind building a website seems like the best approach.

Front-end Every computer, smartphone, and tablet has pre-installed a browser and most of the time the device is connected to the internet via WiFi or cellular. This led us to the use of web development technologies like HTML, CSS3, and JavaScript to build the user interface. Despite these choices we could use also some newer technologies for conviniency reason. Some of these are: SCSS instead of CSS3, Angular framework for a scalable web application development and Typescript instead of Javascript to exploit objet-oriented programming.

Middle-ware The Middle-ware layer will contain a number of Apache web servers that will provide the necessary API to query the underlying data and serve the requests from the front-end. This service will be possibly hosted in an Amazon Web Service (AWS) changing dynamically the cloud resources based on the traffic of the website.

Back-end For now, a simple JSON that contains all of the University data will be fetched when the user requests the website from the browser. In future development, the Back-end of the project will contain a database to store the required data sets. The data will be stored in a My-SQL database.

5 Data Management Plan

5.1 Data Collection

We obtain our data from the Higher Education Statistics Agency (HESA). They are the designated data body for England and they collect, process and publish data about higher education in the United Kingdom. The dataset obtained through HESA is the Unistats Dataset.

The Unistats Dataset is reproduced with the permission of the Office for Students (OfS), HESA and their licensors. The Unistats Dataset may be accessed in its original form here: `www.hesa.ac.uk/support/tools-and-downloads/unistats`. All copyright and other intellectual property rights in the Unistats Dataset are owned by OfS, HESA and their licensors. The license to our usage of the dataset can be found here: `www.hesa.ac.uk/support/tools-and-downloads/unistats/terms-conditions`.

As we aim to develop the application to include universities outside of the United Kingdom, we will need to look for a universal dataset that offers a similar range of criteria. One possible data provider we can research into is the Times Higher Education (THE). As a source of analysis and insight on higher education for almost five decades, it is a reliable source of data for our application.

5.2 Data Processing

We will not be using the raw dataset as our own database, as the format of tables in the raw dataset might change with each annual update and there are redundant data in the dataset. Therefore, the Extract Transform Load (ETL) procedure will be performed when the all the raw data are transferred to our data warehouse. The processed data will be stored in Comma-Separated Values (CSV) files.

In our data warehouse, we will create three tables: DOMAINS, UNIVERSITY_DETAILS and PROGRAMME. The table DOMAINS contains all the domains available in the raw dataset. The table PROGRAMME contains the programme's name, tuition fees for both EU and international students and the level of degree. The table UNIVERSITY_DETAILS contains the information of all criteria, Provider Reference Number (PRN) and the university name.

5.3 Database Structure

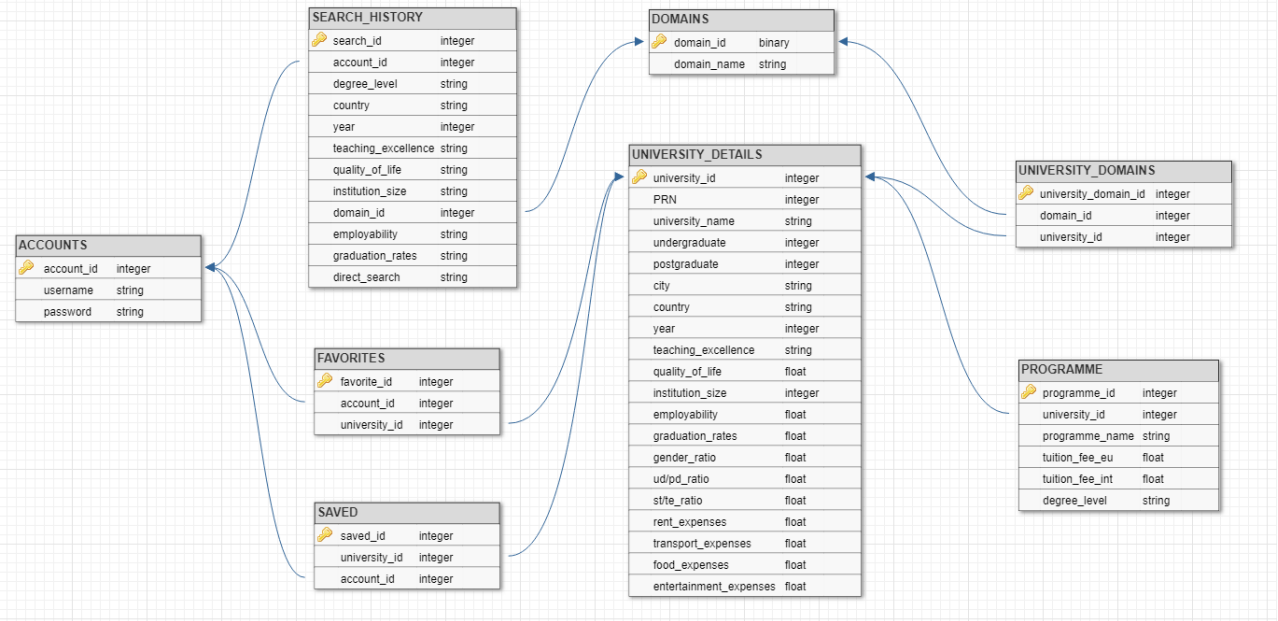


Figure 6: Database Structure

For us to integrate our database to the user interface, we need to include more tables to support the interface's features. As we are allowing users to hold an account, save their results and to register their favourite results, we need new tables to hold these pieces of information.

Therefore, Figure 6 displays the database structure required for our application. The database is built as such to allow easy maintenance of information. Each table is in the third normal form to greatly reduce information redundancy. In addition, as the table ACCOUNTS hold sensitive information of the users, Transparent Data Encryption (TDE) is used to perform real-time I/O encryption and decryption of the ACCOUNTS, FAVORITES, SAVED and SEARCH_HISTORY tables.

5.4 Updates

For both HESA and THE, the higher education data for the current year are published in September of that year. Since most universities' academic year begins in September, prospective students would have started their research at least six months before the matriculated academic year. As we need ample time to clean and update our database after each update of the data providers, we aim to update our application by the end of December to accommodate their research and usage of our application.

6 User Interface Design

This section contains the visualisation and justification for the user interface components of each page (view) of the website. These pages are the only way for the user to interact with the application and access the data model. This is why, the user interface has been designed carefully, to avoid misuse of the platform.

6.1 University List Page

The University List Page is the Index view of the application. Figure 7 show the decomposition of this page into 4 user interface components. These components are analysed in the following paragraphs.

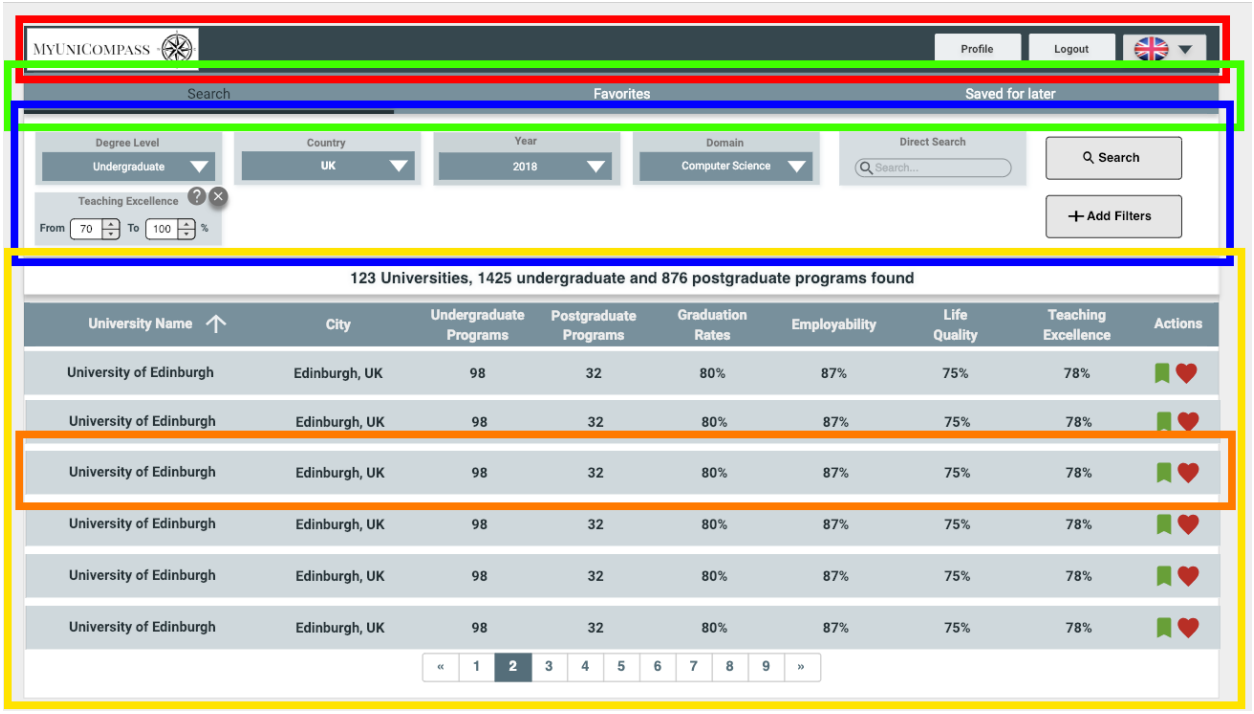


Figure 7: University List Page

Navigation Bar Component This component (Figure 7 Red Rectangle) is meant to be replicated in each page of the website. It contains the logo of the application, which redirects the user to the index page. In addition, it provides options to register, log in as a user, log out, visit your profile page or change the displayed language.

Tab Bar Component Tab bar (Figure 7 Green Rectangle) contains provides 3 different views of the same page these are:

- The Search Platform
- The Favorites Universities section
- The Saved for later Universities section

The Search Platform contains the Search Component which is the core functionality of the application and it will be explained in the following paragraph. The Favorites and Saved for later Universities section contain a list of the favourite and saved universities accordingly. These components will be included in future development.

Search Component The search component (Figure 7 Blue Rectangle) contains the primary (left side) and secondary (right side) criteria that the user defines in order to narrow down the results. The criteria in the search component are:

- Degree level
- Country
- Year
- Domain
- Direct Search
- Teaching excellence
- Quality of life
- Institution Size
- Employability
- Graduation rates

These filters are set from the provided user interface elements and can be combined based on the user's demands. The search component is responsive, this means that whenever the user presses the search button, the results are displayed in a matter of seconds. The primary criteria are by default in the search bar. The user has the option to modify the criteria at will, by adding or removing them. Clicking "add criteria" button will show an additional section that contains more criteria, as shown in Figure 8. The user will be able to add one or learn more about it if he or she moves the mouse over a criterion. The secondary filters will be possibly added in future development.

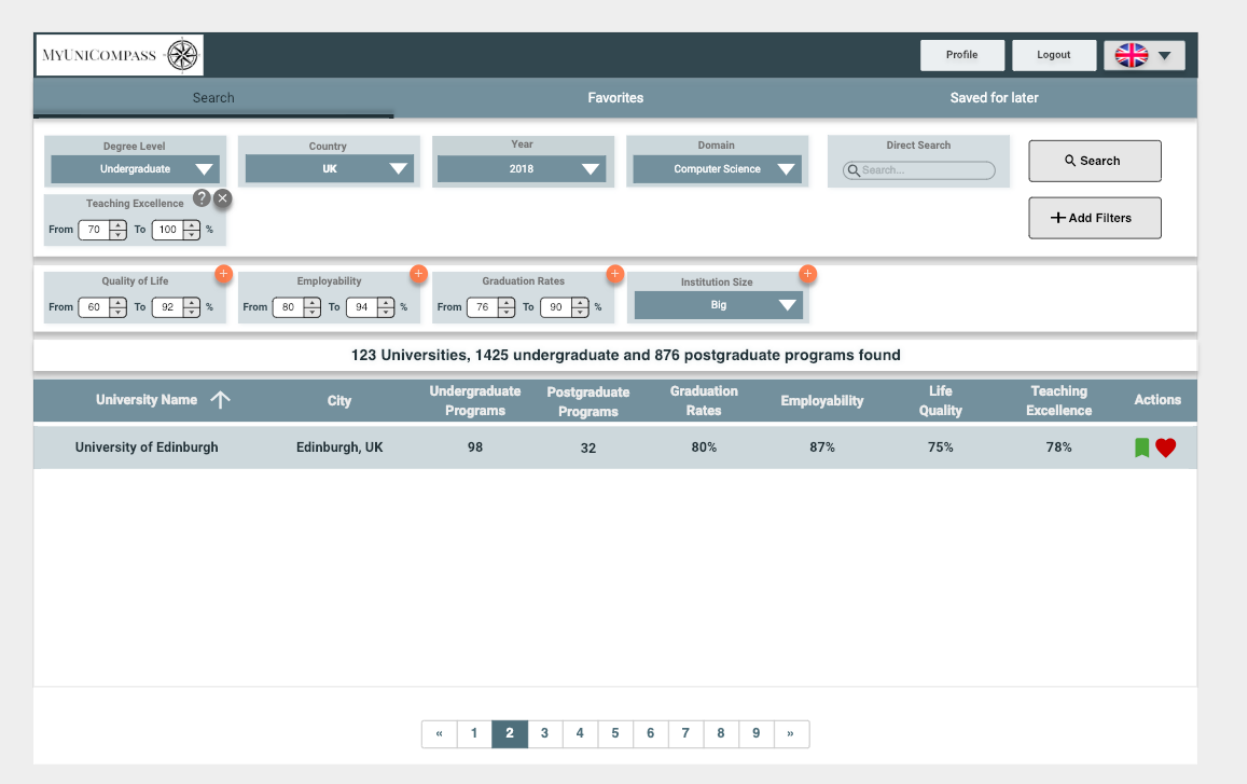


Figure 8: University List Page Expanded Search Bar

University List Component The university list component (Figure 7 Yellow Rectangle) is a list of the universities that fulfil the search criteria and provide basic information about the universities. The University List is displayed in pages and the user can navigate through the pagination bar in the bottom of the list. Each page contains 6 university records. In the top of the results, there is a small information section which displays the number of universities, the undergraduate and postgraduate programs that found based on the applied filters. If there are no results, an appropriate message is displayed.

In future development, the user will have the option to see a comparison of the filtered universities based on a certain criterion. This comparison will be possibly displayed in a bar graph containing the universities in the x-axis and the value of the specific criterion in the y-axis.

University Record Component The university record component (Figure 7 Orange Rectangle) contains the basic information for a university that the user can see while applying the criteria. Each university card contains:

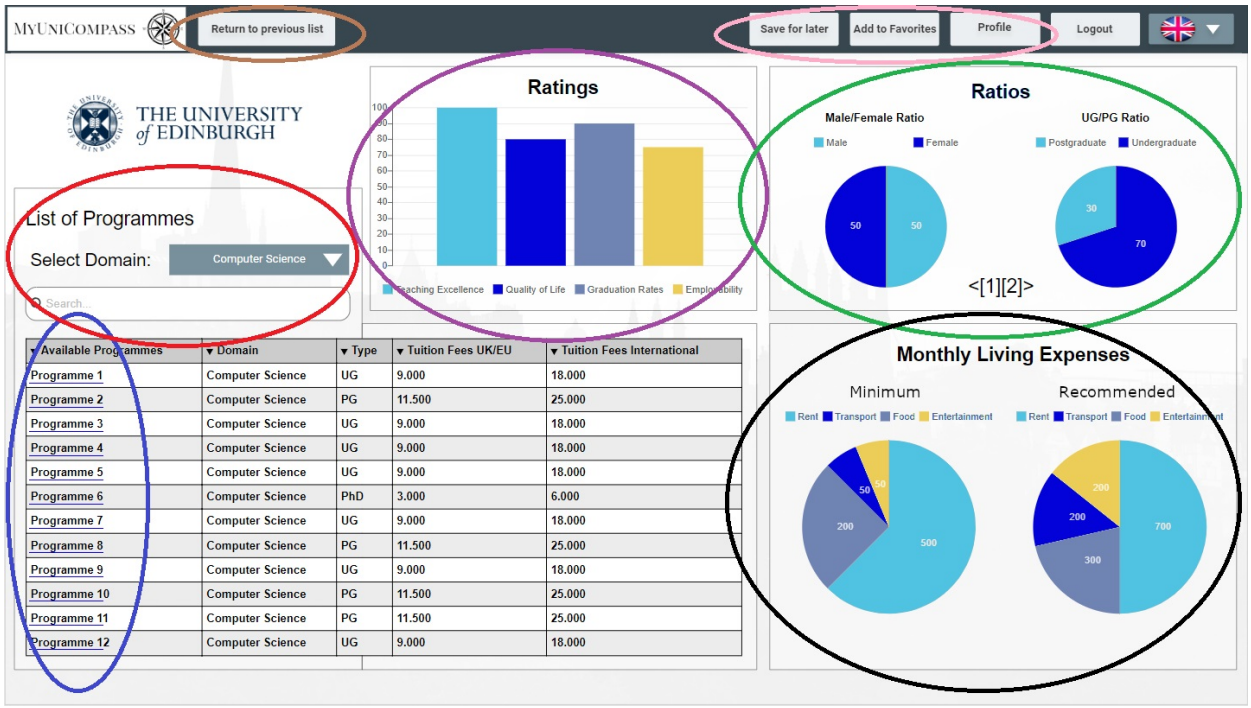
- University Name
- City
- No. of Undergraduate programs
- No. of Postgraduate programs

- Graduation Rates percentage
- Employability percentage
- Life Quality percentage
- Teaching Excellence percentage

Clicking anywhere on a record it will automatically redirect the user to the University Details Page which is explained in the following section.

6.2 University Details Page

Upon selecting a university from the search results (Figure 7), the user will be redirected to the university details page (Figure 9). This page includes and expands upon the information previously displayed in the search results (Figure 7). All of the information on the university details page is specific to the university itself, not a specific course. The individual components of the details page will be discussed in the paragraphs down below.



Available Programmes	Domain	Type	Tuition Fees UK/EU	Tuition Fees International
Programme 1	Computer Science	UG	9.000	18.000
Programme 2	Computer Science	PG	11.500	25.000
Programme 3	Computer Science	UG	9.000	18.000
Programme 4	Computer Science	UG	9.000	18.000
Programme 5	Computer Science	UG	9.000	18.000
Programme 6	Computer Science	PhD	3.000	6.000
Programme 7	Computer Science	UG	9.000	18.000
Programme 8	Computer Science	PG	11.500	25.000
Programme 9	Computer Science	UG	9.000	18.000
Programme 10	Computer Science	PG	11.500	25.000
Programme 11	Computer Science	PG	11.500	25.000
Programme 12	Computer Science	UG	9.000	18.000

Figure 9: details page

List Selection Component The list selection (Figure 9 Red Circle) allows the user to select a specific study domain. Alternatively, the user can search for a specific keyword in the search bar provided below the domain selection button. The search results themselves will be shown in a list on the left, lower corner of the website (Figure 9 Blue Circle).

List of Programmes Once the search results are specified in the list section (Figure 9 Red Circle), a list of available programmes - along with other categories, such as type and tuition fees for both UK/EU and international students - will be shown (Figure 9 Blue Circle). Each programme in the available programmes column (Figure 9 Blue Circle) is clickable and redirects the user to the university's home page, where further information concerning the specific course (e.g. entry requirements for each country) will be provided.

Ratings Chart This chart (Figure 9 Purple Circle) gives a graphical representation of the overview provided by the university record component (Figure 7 Orange Rectangle). The graphcal overview considers the following categories:

- Graduation Rates percentage
- Employability percentage
- Life Quality percentage
- Teaching Excellence percentage

If need be, these categories can be expanded to include other ratings such as research, citations, industry income and other types of information.

Ratio Chart Some statistical information, e.g. male-to-female and undergraduate-to-postgraduate ratios, cannot be displayed in a bar diagram (Figure 9 Purple Circle) and will be moved to this section (Figure 9 Green Circle). Given that there is a wide variety of possible ratios, only two of them will be displayed on the page simultaneously. In order to view other available ratios, the user can click on the bar at the bottom of the section (Figure 9 Green Circle) to slide forward to the next pair of ratios. A selection of possible ratios could be:

- Male/Female ratio
- Undergraduate/Postgraduate ratio
- Student/Staff ratio
- International/UK students ratio

Living Expenses Diagram In this section, the user is given two pie charts displaying the minimum and recommended living expenses per month (Figure 9 Black Circle). Currently, the categories are rent, transport, food and entertainment. Naturally, more categories such as medical expenses can be added to the pie chart if the data is available.

Navigation Bar Component The navigation bar is very similar to the one on the list page (Figure 7 Red Rectangle). However, there are three additional buttons for adding the university details to favourites (Figure 9 Pink Circle, Add to Favorites), returning to the previous search results (Figure 9 Brown Circle) and saving the page for later (Figure 9 Pink Circle, Save for Later). Also, visiting the profile page via the profile button (Figure 9 Pink Circle) allows the user to adjust the colour palette of the application. This functionality not only allows further customisation, but it also increases the overall accessibility of the application. For instance, users that are colour blind or have other forms of visual impairment will be able to navigate the application more easily.

6.3 User Interface Responsiveness

The current application targets multiple devices from Desktops to smartphones. This is a key factor in how the user interface has to be designed. In future development, detailed design for the most popular platforms will be introduced. For now, the design and prototype for the application will be for standard Desktop/Laptop and Smartphone resolutions which are 1366×768 and 375×667 respectively. For each new resolution, the same user interface components have been used with slight modifications. Figure 10 depicts the User Interface Design for the university list page on iPhone 6 screen dimensions, as an example of responsive design.

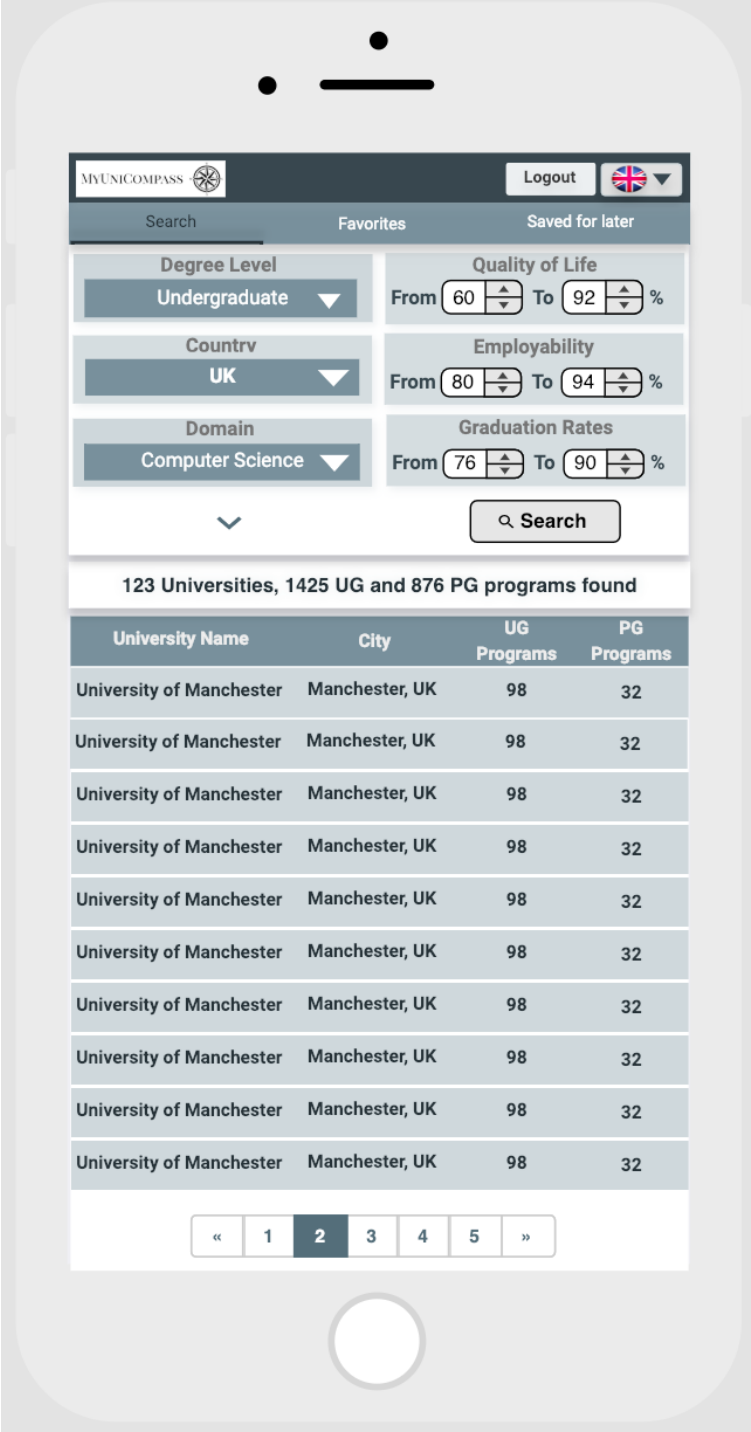


Figure 10: University List Page on IPhone 6 specifications

7 Risk Analysis

7.1 Categories of Risk

Organisational This category encompasses risks related to the organising the work required to complete the project, e.g. scheduling and time management (Figure 11, Number 1-7). Further, this category deals with different work styles, differences in timetables, the increasing amount of coursework over time and a potential reassignment of team members as the project moves along.

Requirements Gathering requirements is a crucial step in setting up the project, which comes with its own sets of risks as well (Figure 11, Number 8-11). Requirements can be ambiguous and unclear, additional requirements may appear further down the line or may be altered completely. Another problem is the potential lack of communication between developers and the end users, leading to misunderstandings and inadequate implementation.

Implementation Implementation is concerned with building the software itself. This process is challenging because it requires an adequate translation of the gathered requirements into a working piece of software/prototype. A selection of risks relevant to the given project can be seen in the table provided (Figure 11, Number 12-18).

Human Resources Named after a standard department in most companies, this category is concerned with risks involving human interaction (Figure 11, Number 19-24). The latter can be between team members or between parts of the team, end users and domain experts. Conflicts, misunderstandings, loss of people due to sick leave or other reasons, lack of experience and different backgrounds are the risks associated with this category.

Data Management Albeit a category that could be considered as part of the implementation, data management is so vital to the functioning of the application that it deserves its own category (Figure 11, Number 25-28). Data has to be gathered, cleaned and somehow processed so that it can be displayed in the final product. Depending on the complexity of the application, those steps can be very difficult to carry out. It is also worthwhile mentioning that there are also potential legal risks when dealing with data, even though those will be of relatively little concern to a coursework project. If applicable, data storage and retrieval also carry a certain security risk, especially if sensitive user data is involved.

Number	Category	Risk
1	Organisational	More coursework in the upcoming weeks
2	Organisational	Different work styles
3	Organisational	Different time tables
4	Organisational	Time management issues
5	Organisational	Reassigning team mates
6	Organisational	Missing the deadline
7	Organisational	Optimistic scheduling
8	Requirements	Requirements Addition
9	Requirements	Requirements Modification
10	Requirements	Inadequate Requirement Gathering
11	Requirements	Lack of clarity
12	Implementation	Cross Platform Compatibility Issues
13	Implementation	Design Deviation
14	Implementation	Inadequate Design
15	Implementation	Increasing project complexity
16	Implementation	Technology becoming outdated
17	Implementation	Overengineering certain features
18	Implementation	Unexpected Bugs
19	Human Resources	Conflicts within group
20	Human Resources	Different backgrounds
21	Human Resources	Inexperienced members
22	Human Resources	Sick leave
23	Human Resources	Lack of communication
24	Human Resources	Loss of teammates
25	Data Management	Difficulty of finding suitable data
26	Data Management	Integration issues
27	Data Management	Public data links breaking
28	Data Management	Data security

Figure 11: Table of Risks

7.2 Methods of Risk Analysis

Quantitative Risk Assessment This method aims to create a priority list of risks based on two variables: the probability of the risk occurring and the amount of time needed to deal with the risk (Grant 2019, Week 1, p.10). The major problem with this approach is its assumption of clearly defined risks and a clearly defined frame of time to deal with them. For smaller, less complex projects this approach may be sufficient.

Qualitative Risk Assessment Risks do not have clearly defined boundaries. Some of them are technical and can only appear at certain stages, whereas other risks (e.g. conflicts with team members) can persist for the entirety of the project. Thus, grouping risks into a set of categories rather than numbers (Grant 2019, Week 1, p.12) pays tribute to the fact that risks are inherently elusive (Grant 2019, Week 1, p.11). A different set of categories is chosen for both the probability and the impact. With those categories in mind, a graphical representation of risks can be produced in the form of a diagram (Figure 13 - 17), where the y-axis represents the probability and the x-axis impact (Grant 2019, Week 1, p.13).

Chosen Method The method chosen is a slightly modified version of the Qualitative Risk Assessment. Firstly, the probability and impact of specific risk changes over time. Furthermore, risks previously not taken into consideration may appear on the horizon while other risks may become irrelevant. For this reason, the probability, impact, priority and risk handling methods will be adjusted on a weekly basis while keeping records of the old estimates as well. Thus, decisions made in the past regarding risk management can be justified by pointing to previous risk estimates. Moreover, the progressive refinement of risk estimation is a learning experience in itself with potential benefits towards future projects in regards to planning. Secondly, a scale from one to ten is used in conjunction with the aforementioned variables (Figure 12). This level of granularity was chosen for practical reasons since it allows to create a range of impact-probability charts (Figure 13 - 17) without cramming too much data into a tight set of categories. It also leaves enough granularity for a meaningful distinction between risks that may be similar but not equivalent.

Probability	Impact	Priority	Code
Absolutely Certain	Failure	Critial	10
Highly likely	Could threaten entire project	Significant	9
Very likely	Major issue	Very Important	8
Likely	Substantial extra work for more than one team member	Important	7
Probable	Requires involvement from more than one team member	Requires attention	6
May or may not happen	Additional work required, but still manageable by one person	Normal	5
unlikely	Issue that can be dealt with by one person within a short time frame	Worth some attention	4
Very unlikely	Minor issue	Low	3
Highly Unlikely	Negligible	Very Low	2
Theoretically possible	None	None	1

Figure 12: Numerical Codes for Probability, Impact and Priority

7.3 Impact of Identified Risks

Overview As previously outlined in section 7.1, similar risks were grouped in the same category. Given the method of choice in section 7.2, every risk was assigned a probability and a potential impact value between one and ten. This exercise was repeated on a weekly basis to ensure that the current outlook on potential risks always stays up-to-date. A comprehensive list containing all weekly risk estimates in the categories probability, impact and priority are available from the team upon request.

Risk Data The risk data obtained was then visualized using a probability-to-impact scatter plot (Figure 13 - 17) as outlined in the risks and change lecture (Grant 2019, Week 1, p.13). Due to a large amount of risks (Figure 11) and the weekly updates, only a snapshot taken at the beginning of week four will be presented.

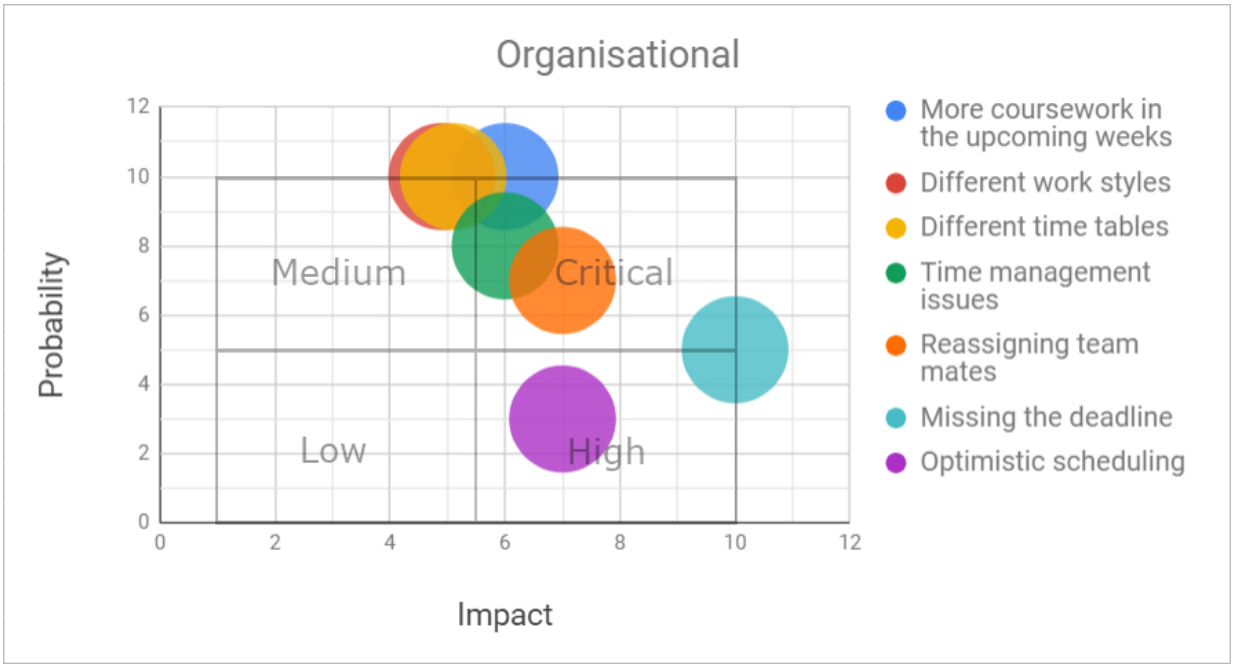


Figure 13: organisational

Organisational Risk Estimations As of week four, the most pressing issues concerning organisational risks were time management, different timetables and work styles (Figure 13). For example, one of the team members worked remotely from China, which also involved dealing with time differences. With the apparent progress of the project this week, the probability and potential impact of all risks in this category will be reduced. The only exception is "Reassigning teammates" since we expect the teams to be arbitrarily shuffled at some point in the future to simulate real-world conditions.

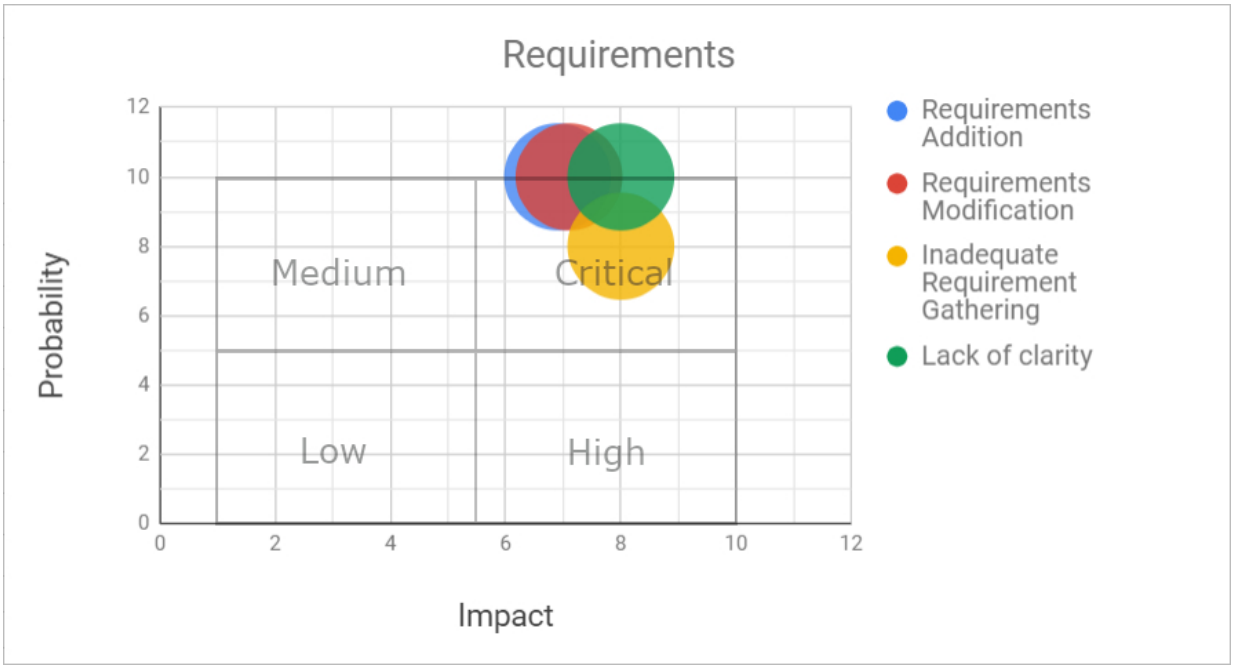


Figure 14: requirements

Requirements Risk Estimations Having a clear understanding of the requirements is essential for the project's success. In terms of risk, it is also a huge breaking point for the entire endeavour if not done properly. Therefore, all risks in this category are located in the critical section for the time being (Figure 14). Overall, those risks are not expected to move away from the critical section easily, because requirements can be added or modified by the course organiser to simulate real-world conditions.

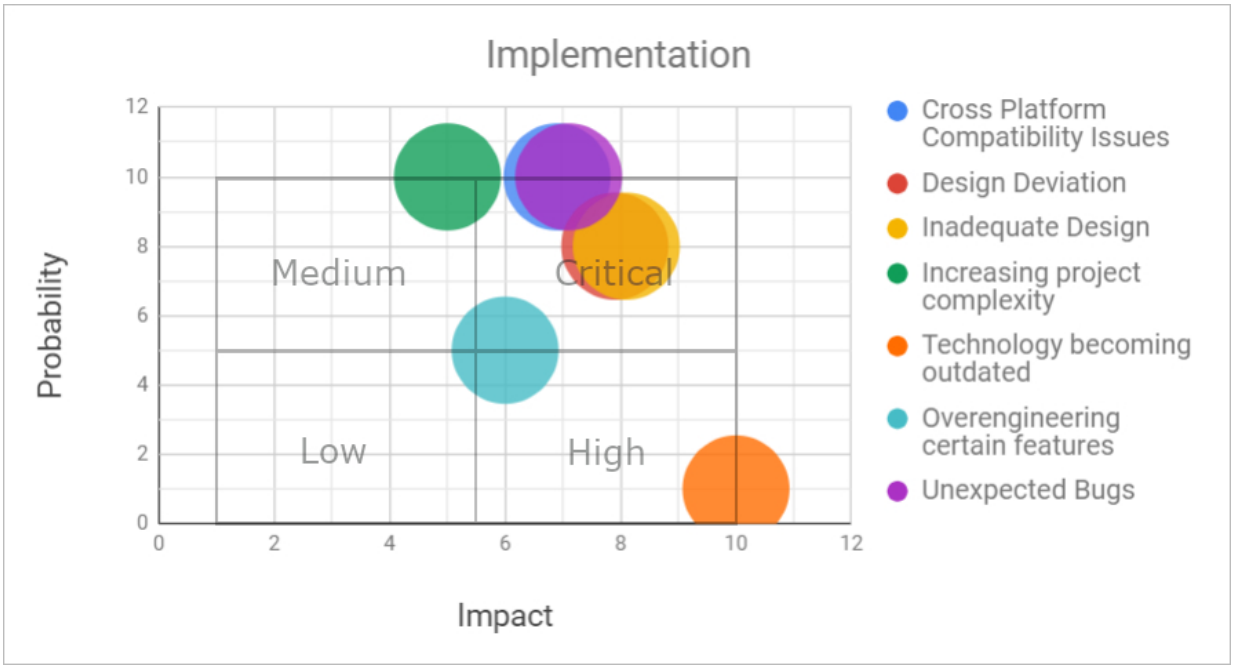


Figure 15: implementation

Implementation Risk Estimations Implementation of the project has not yet begun, but an estimate of the most probable and impactful risks has already been carried out (Figure 15). Cross-platform implementation will certainly play a role due to it being an essential requirement. Bug fixes are unavoidable in all software projects. However, the most dangerous risks in this category are inadequate design and design deviation. Here, inadequate could potentially mean inflexible, leading to enormous difficulties in implementation when requirements suddenly change. Design deviation could lead to the implementation of unnecessary features and towards missing the goal of the software entirely.

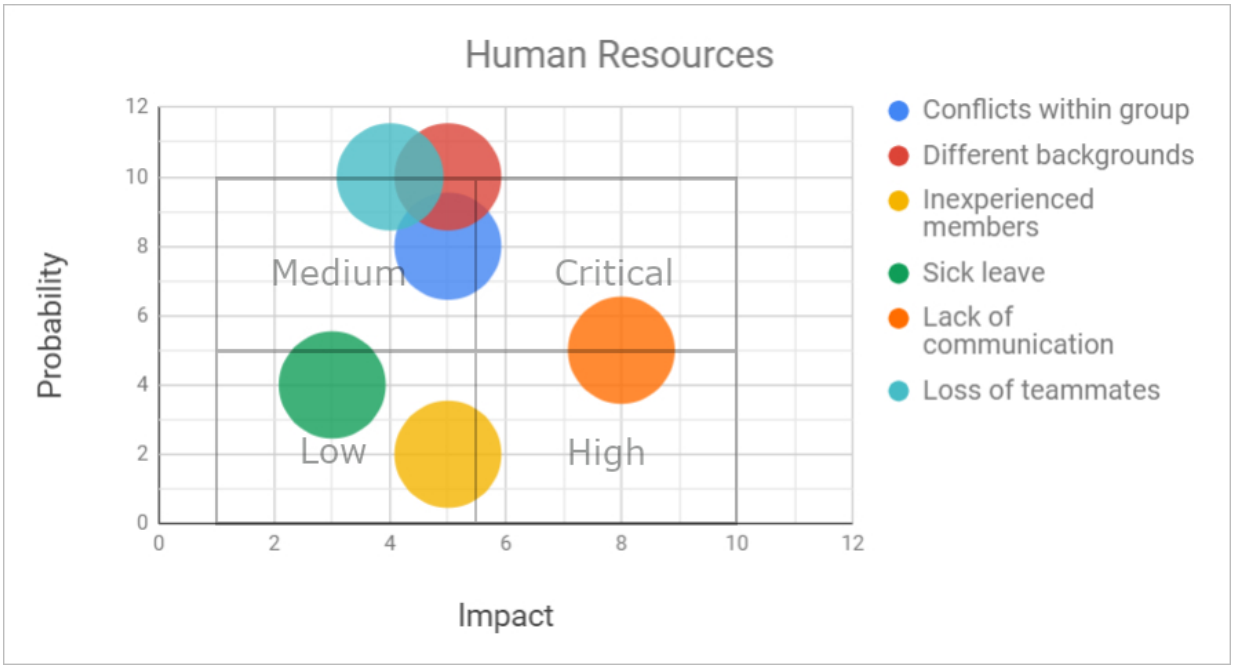


Figure 16: human resources

Human Resources Risk Estimations This category is an example of how risk was effectively reduced over time (Figure 16). By week four, none of the risks was in the critical section. That was not the case at the beginning of the project, since forming a new team and dealing with differences in personalities can be problematic. In particular, sick leave, a short term dropout and subsequent return of a team member turned out to have a much smaller effect than previously thought. Another reason why the risks in this category are deemed to be more manageable is due to the small team size and the small number of parties involved in the project. Commercial projects not only may involve a larger number of developers, but they also have business owners, customers and shareholders to worry about (Yourdon, 1997). Moreover, the clash of interests between different groups, also known as politics (Yourdon, 1997), does increase the probability of failure dramatically. Luckily, these types of risks are outside of the scope of this coursework.

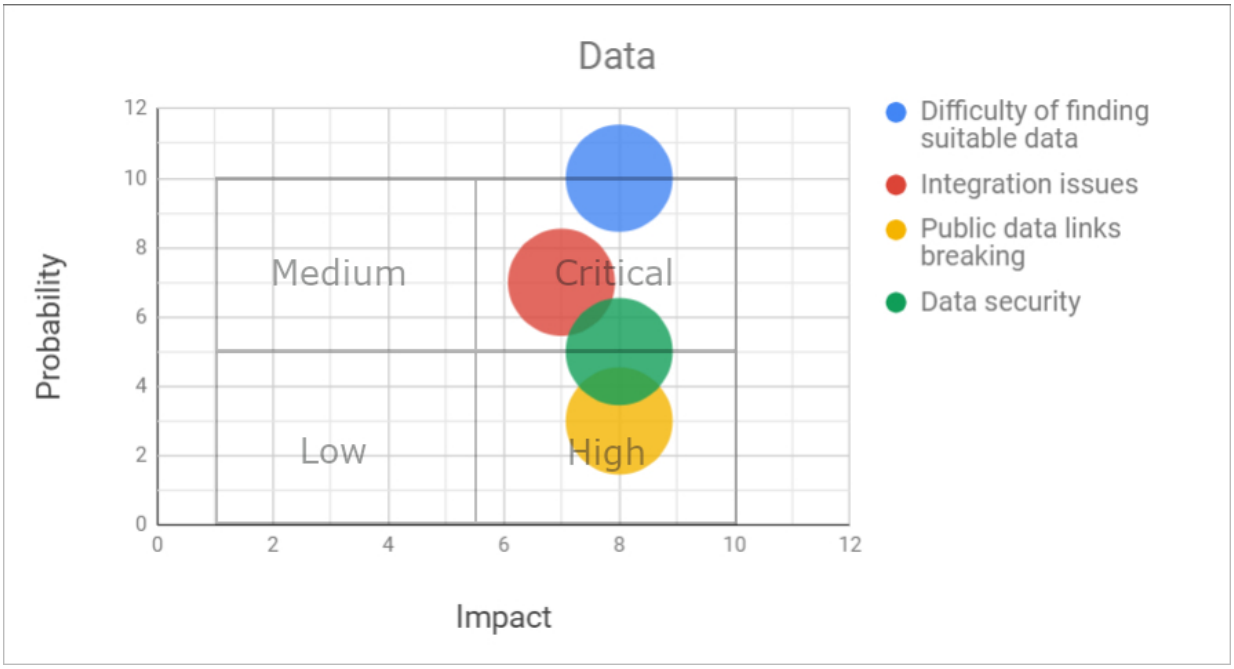


Figure 17: data management

Data Management Risk Estimations The core value of this project lies within the quantity and quality of its data. Having to use publicly available data, however, is immensely challenging for a wide variety of reasons. Thus, not being able to obtain suitable data and failing to integrate different data sources are the two critical risks in this section (Figure 17). Data security, albeit a risk as well, is deemed more manageable because the data itself is already public. One potential problem may be user data if personal accounts will be allowed. Lastly, public links may break over time, which is not too much of a concern because the duration of the project is limited.

7.4 Risk Mitigation Strategies

There are various ways of dealing with risks. A selection of risk management methods is given down below (Grant 2019, Week 1, p.20-21):

- **Avoid** Offset risky design area to team with more experience
- **Transfer** Getting risk off the critical path so it is less of a risk
- **Buy** Give team time to research graphics library for functionality
- **Eliminate** Remove risky functionality from the current project version and treat it as a research project
- **Publicise** Inform customers/upper management that the risk exists to minimise surprise if it occurs
- **Control** Alter schedule and resource allocation to accommodate the risk and lessen its effect
- **Remember** Document all risks which affect the project for reference in further projects

Organisational All risks in this category are dealt with the methods of control. More specifically, issues regarding time management, remote work and scheduling of tasks are actively resolved in team meetings. The only exception in this category is the potential reshuffling of teams, which cannot be planned for. Therefore, the latter risk is simply assumed.

Requirements The best way of avoiding trouble with requirements is to actively gather information (buy). In our case, the requirements were obtained by means of electronic and personal communication with the course organiser. To some extent, this could also fall into the category of publicising, because problems and questions concerning the requirements are in a way made public to participants outside of the team.

Implementation Implementing the actual software, especially if new technology is used, can be considered a multidimensional problem. If new skills are required, e.g. when learning a new programming language or framework, buying information about the risk is appropriate. Avoiding certain risks, for instance, over-engineering certain features have to be considered as well. The risk of introducing bugs can be mitigated by avoiding an overly complex design at the early stages of the project. Ultimately, simply allocating more resources to a problem (control) can also work, provided the additional time spent is within reason.

Human Resources Dealing with personal problems within the team can be separated into several stages. If no problems exist so far, conflict can be avoided by the team members by being more cautious about their actions (avoid). Of course, this only works up to a certain point; cultural difference and language barriers can bring about an unintentional misunderstanding. At that point, the damage can only be assumed and the risk mitigation strategy goes back to avoid. If problems are escalating, publishing the issues to superiors (e.g. the course coordinator) and asking for intervention may be considered as a last resort.

Data Gathering data from public sources is essentially a problem that can only be solved through more effort (control). Regarding the legal side of data management, getting well informed about relevant laws (buy), adding additional layers of data security within the software (control) and overall trying to minimise the exposure of data online (avoid) are the methods to be considered in this category.

8 Feedback and Future Development

As the development of the UI design was unfolding, feedback was given by the lecturer on potential areas of improvement. A list of suggestions can be seen down below.

Feedback on Home Page (UI)

- Progressive addition of options rather than offering all options simultaneously
- Start with the most basic options first, e.g. degree level
- Perhaps offer a tutorial once the user has signed up
- Consider different options for the list view of the university results, e.g. tables, lists and graphs
- Allow the option of showing numerical comparisons between single categories of different universities, e.g. employability, in a list view
- Provide an easy way to switch between aforementioned list and graphical representation

Feedback on Details Page (UI)

- Offer the option to change colours to increase accessibility

Feedback on Risk Analysis

- Storing weekly adjustments of risk analysis in an accessible location

Feedback on Data Management

- Including data for universities outside of the UK

At the start of the implementation, it is vital that the core functionality will be implemented first. Non-essential functionality ,e.g. logging in and signing up, will be postponed until the application reaches a more mature state. The same principle applies to the suggestions given above unless they prove to be useful for the core functionality itself.

9 References

9.1 Lecture Slides

Jackson, M. 2019, Lecture Week 5: Case study, Lecture slides for Software Development, EPCC, The University of Edinburgh, delivered 11th February 2019

Sloan, T. 2019, Lecture Week 3: Data Understanding and Cleaning, Lecture slides for Data Analytics with HPC, EPCC, The University of Edinburgh, delivered 28th January 2019

Grant, A. 2019, Lecture Week 1: Risks and Change, Lecture slides for Software Development, Organisation: EPCC, The University of Edinburgh, delivered 18th January 2019

Kavoussanakis, K. 2019, Lecture Week 3: Development Models, Lecture slides for Software Development, Organisation: EPCC, The University of Edinburgh, delivered 18th January 2019

Kennedy, J. 2019, Lecture Week 4: Project Teams, Lecture slides for Software Development, Organisation: EPCC, The University of Edinburgh, delivered 4th February 2019

Azzopardi, L. 2016, System Architecture Lecture slides for Internet Technology, Organisation: The University of Glasgow, delivered February 2016

Azzopardi, L. 2016, Web Application Frameworks Lecture slides for Internet Technology, Organisation: The University of Glasgow, delivered February 2016

9.2 Books

Yourdon, E. 1997, Death March-The Complete Software Developer's Guide to Surviving "Mission Impossible" Projects. Upper Saddle River, New Jersey: A Simon & Schuster Company, 50-51

9.3 Reports

Grant A. 2019, Software Development Assignment Description: Software Development Coursework, `https://www.learn.ed.ac.uk/bbcswebdav/pid-3615572-dt-content-rid-7415121_1/xid-7415121_1`