**Interim report**

**aims, objectives and literature survey**

**Aims and objectives:** I decided to choose this project because it is part of the computer science field that first drew me to the course and is a part of my career aspirations post university. Whilst studying A levels I took part in a week’s work experience with Sky and worked with a group of graduate front end developers on a streaming application tailored for business clients of the company. Being able to get hands on experience in the agile process of front end development is where I started thinking more about how we actually interact with computers and how much fine detail goes into making a fresh and usable interface. As well as this, my second year team project based on creating a website gave me the chance to design a user interface as part of a team. After finishing the team project and getting a better idea of how consumers want interfaces to work and look, I wanted to do something similar for the individual project in my final year.

With this project I would like to incorporate one of my passions which is sport. Using sport as a focal point will open up a variety of options for design within the interfaces I will produce. Having taken part in university sport over my time at university and also being elected to be the captain of a sports team I understand that fixture management can be difficult. As well as this I understand that students who take part in sport in any way do not have a centralised point for any and all information they may require. Therefore I would like to design some interfaces to be used by students and coaches that are members of clubs to check fixtures, post results and view the availability of students for fixtures in each team. While also providing further information for more recreational students in terms of memberships and key dates and times of activities.

Within this project I would like to explore accessibility and its effect on how different humans are able to interact with user interface.

**Background Theory:**

**Intro:** Human-Computer Interaction (HCI) research began in the early 1980’s due to the rise of personal computing in the late 1970’s. Prior to the 1980’s computing was carried out by IT professionals or dedicated individuals as a hobby. As a result of this the computer science community needed to find a way to present information for a wide variety of humans. HCI theory is broken up into three parts; how humans process information, how computers store and present information to its users and the interaction that occurs between the two to address the required action. I will now discuss these parts further and explain how they come together to have an impact on interface design.

**Human**s receive information via certain channels; visual, auditory, haptic and movement. Information is then stored in classes of memory; sensory, short-term, long-term and processed via reasoning, problem solving, skill acquisition and error which can be defined as cognitive tasks. Another process which falls under the umbrella of cognition is attention, this is the procedure of selecting things to concentrate on at one point in time from a range of possibilities. This allows us to focus on the information that is relevant to the task we want to achieve at that point in time. The success of the task is based on whether we have clear goals and how easy the environmental information is to interpret. As a result, dealing with human attention is paramount in dealing with HCI.

Earlier I explained the classes of memory that humans use to store information, this stored information is then used to respond appropriately to a certain task. However, it is not possible for humans to remember everything we ever store in memory or our brains would overload, instead information is filtered through based on how much attention is paid to it. It is also well known that humans tend to pay more attention to colours and shapes rather than numbers, words or speech .Here we can see how parts of cognition come together to deal with information processing in our brains. Therefore when dealing with HCI we need to reduce the load on user’s memory with simple procedures and prioritise recognition using menus and icons.

The human ability to learn is another important aspect to consider when discussing HCI theory which uses the concepts of attention and memory to recall similar past experiences and use previous responses to act upon a required task. We tend to learn a lot more from doing than following a set of instructions, and as a result it is important to design with exploration in mind in order to allow humans to learn the most efficient and productive solutions to problems. Learning is also about improving upon mistakes, allowing users to return to previous states will encourage learning and help guide them through the task. Overall, as humans we process information in a variety of ways and using the concepts of cognition we can employ the use of computers to deal with complex tasks much faster and easier than we could on our own.

**Computers** have similar human features like software packages that are used to process information with the help of input/output devices, memory and processing power which make up the hardware of a computer. However, computers have the ability to perform these tasks much faster providing solutions to complex queries that couldn’t be solved by the average human in the same time frame. Computing has evolved over time by increasing its accessibility in order to meet the requirements of users.

In the history of accessibility there are some substantial inventions that were created to assist people in completing tasks, for example in 1808 the first typewriter was built in order to help a blind person write legibly. These historical developments directly relate to how a variety of users with different physical and mental conditions which hinder their ability to perform certain tasks are still able to use computers today. There are a number of assistive technologies that have been produced to support HCI both in hardware and software. An example of this in software is screen readers which were developed by IBM in 1986 to support visually impaired staff by reading aloud content on computer screens, this was soon expanded to be available to all users of personal computers. An example of an assistive technology in hardware is an adaptive switch which allows people with movement-limiting disabilities to use computers without the need to perform complex actions by offering easier movement solutions such as pressing a button. There are different types of switches available such as joysticks, buttons and sound which can all be configured to a user’s specific needs. The use of assistive technology allows computers to adapt to different user bases and environments which makes user interaction simple for all.

In the introduction to the background theory I touched on the rise of personal computing being the primary source of HCI research. At this point in the short lifespan of computing, industry professionals were referred to as operators who used computers to perform specific tasks relating to one end product such as producing medical statistics. Computers had also started to become programmable by trained users due to the developments in hardware which made the machines smaller, cheaper, faster and easier to use.

In 1977 there was a major breakthrough for personal computing with Jobs and Wozniak exhibiting the Apple 2, which had a built in programming language called Beginners All-purpose Symbolic Instruction Code (BASIC). This language was developed with a design philosophy which emphasizes ease of use and was supported by hardware which can store programs and data on a compressed storage device. This then became the root cause of computer production where non-professional users could operate these machines without being subjected to intricate commands and system dialogs. Thus there is now a new community of users that have different ways of thinking compared to industry professionals, which is when the notion of cognitive engineering arose to talk about applications which are informed systematically and scientifically.

Cognitive Engineering defines the **interaction** between a human and a computer to address the translations between what the user wants and what the system does. This started HCI research by developing techniques to evaluate how humans interacted with computers, at which point relevant documentation was then produced to analyse how tasks were performed. Documentation outputted by systems used in personal computing moved away from producing technical descriptions and prioritised supporting users to achieve goals by recognising and recovering from error. This helps support the process of learning that I discussed previously where I stated that interaction should be designed with a way to guide users through error making system-based tasks easy to learn and easy to achieve.

Interfaces are used to deal with these interactions hence the style of an interface is determined by the style of interaction that needs to occur. At the origin of personal computing, HCI theory was limited to desktop applications such as spreadsheets and word-processing tasks. Desktop interfaces have gone through many changes starting with files and folders being represented as icons. However over time user’s desktops became cluttered with icons so a search functionality was incorporated to graphical user interfaces to reduce the need to keep icons visible on the desktop. The rise of the internet drove HCI further away from the desktop through applications such as e-mail. People started using computers to interact with other people in this rapidly developing environment known as social computing. These interfaces concentrated on collaborative work through instant messaging, forums and online communities such as Facebook and GitHub. Another reason for evolving interfaces is the introduction of different computing devices. Nowadays computing has been incorporated into various parts of daily tasks for humans, most notably laptops, mobile phones and cars. Desktop computing has come a long way since its inception where it now informs interface design to enhance human activity and experience with various systems.

Usability Testing is used to inform software development where non industry users are observed to see where they encounter problems and experience confusion. This is an important part of interaction design as it identifies non-functional issues with interfaces. The tests measure user interfaces (UI’s) based on how easy it is for a user to reach their goals. The bias of industry professionals is completely removed, and the focus is put on end users who will have differing skills and experience to developers and testers.

Users share common capabilities but also have distinct differences to be considered when designing HCI concepts. There are a number of **HCI accessibility issues** that user interface designers in many countries are legally obliged to incorporate. There is a common misconception that these issues are purely related to some form of physical or mental disability however they can be any form of difficulty a user may have in being able to productively use an interface. I will now discuss several areas that should be looked at when designing with accessibility in mind.

There are different forms of visual issues that need to be catered for. For example, users can have various deficiencies in colour vision which can include perceiving colour contrasts differently than in normal sight, the inability to see certain colours or even any colour and extreme sensitivity to flickering lights (photosensitive epilepsy). We then have to think about users that have limited or no vision at all. These cases vary from tunnel vision where users can only see central elements or blurred vision where text becomes very difficult to read etc. In each case there are a number of practices we can employ; ensuring content is separate from the structure of the interface allows web browsers to interpret the information and present it in alternative ways to support user requirements, providing text substitutes for images to allow screen readers to describe them, avoid the use of colours and instead use textures and icons to differentiate between elements and avoiding harsh flashes or transition animations.

Interfaces also cater for mobility issues which for web based products can be difficult to design for. It is important that the environment in which the interface is being used in is thought about as this may affect users which will have difficulties accessing or moving around when using it. We will need to allow alternative input devices to be used, such as eye tracking applications and mouth sticks to aid those without motor function in certain areas, as well as ensuring that precise mouse positioning is not required allowing keyboards to be used to traverse through links on browsers. Auditory disabilities are also tricky to design for web interfaces unless the product has a multimedia aspect to it. In these cases where a UI requires the use of audio clips or videos, transcripts should be provided to be made available through subtitles which helps those with auditory concerns and may also benefit non-native speakers of the language used in said media.

Cognitive disabilities will also affect the accessibility of a UI in different ways. Users can have issues with spatial reasoning which affects their ability to visualise the structure of information presented to them, implementing a sitemap alongside simple and efficient navigation will aid this issue. Some users have problems reading large amounts of text, so it is wise to promote scanning of key words and links to help ease the use of the interface. Users misspell words all the time regardless of their cognitive ability so providing a spell checker or similarity search when requiring user text input is a great design idea which will benefit users from other languages as well.

**Conclusion/applied knowledge:**

In conclusion, the background theory that I have explored has helped to me to follow the history of HCI and how user accessibility has evolved along with it. I have found that the clear turning point for HCI research is the rise of personal computing where the world of computers welcomed a host of new users with vast collection of cognitive thinking skills. Designing with usability in mind is a key concept of HCI, hence being able to produce intuitive and accessible interfaces is essential to enhance user experiences and promote inclusivity. I will now talk about some guidelines that are used in the real world to define accessibility attributes for UI’s.

Accessible Rich Internet Applications (ARIA) is a specification used for web UI’s that allows developers to implement solutions to accessibility issues that base HTML cannot solve on its own. Elements of interfaces can have specified attributes which provide extra information or checks when it is being translated into accessible formats for users. In the following example the Google developer website explains a use of ARIA for users with sight issues:

<li tabindex=”0” class= “checkbox” role=”checkbox” checked aria-checked=”true”>

Receive promotional offers

</li>

This code snippet uses the ARIA attributes “role” and “aria-checked” to explicitly identify the element as a checkbox with a Boolean check that we are verifying its use in the accessibility tree. It will now allow a screen reader to correctly report it as a checkbox.

The Web Content Accessibility Guidelines (WCAG) are essentially a step by step manual on making websites or application interfaces as useful as possible to all users. They are categorised into four principles; Perceivable “users must be able to comprehend the information being depicted. It can’t be invisible to all their senses”, Operable “the interface cannot require interaction that a user cannot perform”, Understandable “users must be able to understand the information as well as the operation of the user interface”, Robust “Content must be robust enough that it can be interpreted reliably by a wide variety of user agents, including assistive technologies: As technologies and user agents evolve, the content should remain accessible”. I will use the guidelines within these principles to assist in the development of the web UI and app UI.

**planning and time-scale**

**summary of completed work (SE Tools)**

Web UI: For my web-based UI I started by going back to the team project from year 2 where I was part of the front end team and responsible for the design and implementation of the e-commerce interface that the customers would use. I thought about what pages I would like to present, how they would interact with each other and how they would look. Starting with the foundation of any website I constructed a standard home page using html and gave it a title heading and some text to test that the structure of the file was correct. From this point my development strategy was to put myself in the mind of someone using the website and think about what I would like to achieve. With this in mind I created some links to various pages I would expect to be used for this style of website for which I had to create individual page files for each. Once navigating to one of these pages however I need a way to get back to the home page or to somewhere else on the website, hence a navigation bar was needed for all pages. I defined a div for this nav bar and applied some basic stylings to clearly define it to users.

Next I went across each of the pages and added some relevant content to fill the page up. Starting with the fixtures page I used a table to list some upcoming fixtures for different clubs within the university. This was later changed to include recent results and became a fixture centre page to allow users to keep updated on various clubs. I then went on to look at the **clubs** page where I wanted to use images to link to the individual pages for each club. I then did some research to find some copyright free images and made notes on any images that required acknowledgement. I used a card style format to layout the images for each of the eight example clubs and applied some CSS to clean up their presentation. Rather than creating similar pages for all clubs I used the American Football page as an example of how they would all look, filling content with headings and tables with relevant information on the clubs in terms of training, fixtures etc.

After this I looked at a login feature which I wanted to incorporate into the navigation bar of each page so that users will be able to access their accounts from anywhere on the site. My first thought was to make it simple for users to login via the nav bar using a form directly inside the div. I then changed my mind and made it simply a login link identical to the other links on the nav bar with some CSS to place it on the right side of the page. This link meant I needed a login page with no navigation just a single div which has a title and the login form with a submit button and a cancel button which returns users back to the home page.

The next pages I looked at were the venue and membership pages which also followed the card style format used in other pages. The venue page has a set of three cards with titles, images and descriptions of some locations where sport is hosted. The membership page uses cards for each unique membership type, there are two divs one for student opportunities membership and the other for gym membership. Users can add each membership card to a basket and checkout features will be implemented next term.

Text-based UI:

Git: I feel like I have made good use of my Git repository with regular and meaningful commits made over the five week period I spent in development. Every commit came with a message that clearly describes my thought process and the actions that I took. From my commits is was then able to summarise each commit into a diary entry which kept track of my progress throughout the project. I also made use of branches to separate work from the different parts of my project. From my master branch I created 3 new branches; WebUI\_DEV, monochromeUI\_DEV and report\_writing. I could then concurrently work on these distinct parts of the project and any time a significant task or part of each branch was complete I could merge it with the master branch and make sure the master branch is up to date with all aspects of the project. Another aspect of git that I used is the ability to checkout a set of working files or folders. I typically used this to checkout as little files as possible to reduce conflicting files. For example when writing my report I would just checkout the report file and changes made would only affect that single file.

**bibliography and citations (not inc word count)**

Human-Computer Interaction (3rd Edition) (website)

Interaction Design: Beyond Human-Computer Interaction (5th Edition), Wiley 2019

Sketching User Experiences: Getting the Design Right and the Right Design, Elsevier 2007

The Encyclopedia of Human-Computer Interaction, 2nd Ed.

<https://developers.google.com/web/fundamentals/accessibility/semantics-aria>

**some form of diary.(not inc word count)**

Plan: I started doing some light research in September leading up to the start of term and bookmarked some readings and websites with information that I either felt was interesting or of direct relevance to my project.

Design(sketches/research):

Touch screen UI:

**Appendix (file structure) (not inc word count)**

**Code snippets (SE tools) (code not inc word count)**