SotonMultiConnect

*The following describes a **client-server application**, intended to be user-friendly on the client side, and designed to help users (new students) in finding information meeting their interests in and about the department.*

Context

Arriving at university can be an unsettling time. New students can be helped to settle in if they have a clearer idea about the activities offered by the university and events of social groups, and general social interests of fellow students. Traditionally these have been communicated by adverts, flyers, the bunfight, piles of paper, verbal announcements and word of mouth. These methods are not always very efficient and can lead to information overload.

An alternative system which used technology to support and enhance information sharing might be more efficient. Such a system could be used to collate and associate information. Features could include shared contributions, collaborative updates and informal dissemination of information via a technology equivalent of word of mouth.

Many students already make use of university message boards and systems such as Facebook prior to joining university to try to build up social contacts, find out about the town and their chosen degree and gain a sense of what lies in store for them.

Challenge

In the time available, design and build a basic (proof of concept) system to store and share information for new students. The outcome will be a centralized user-generated and self-maintaining sytem; users can register an interest in things they find, and find other users, all happening in one place, in a social and dynamic way. The system will incorporate some aspects of the functionality described above (and defined more clearly below) . Is should be clearly designed to allow further/future enhancements and extensions. Designers should clearly indicate their intentions for enhancements and extensions in their code as comments.

It is envisaged that information within the system would be stored as a vast set of inter-connected nodes each of which can link to other nodes. The system ideally should support some kind of visual rendering of the various inter-relationships (from different perspectives such as hierarchical, associative, etc). The rendering for this proof of concept design can be very simple.

Examples of functionality/actions

For example the system should be able to represent

* **Nodes** - which are generally: **o Nodes** that link to other nodes (e.g. Sport-> Football-> 5-a-side-football; Pubs-> Open till 2am) + The links do not necessarily have to always be hierachical - e.g. node A could easily link to node B, which in turn links to node A + Users can join nodes + Nodes should have a name and perhaps some form of description/details **o Events** - which have some kind of time frame: e.g. one off, repeating, occasional, other, date/start time/end time + You might consider a club/special interest group to be a special example of an event o Any other types that you think might be useful	* **Users** - which include: o User id o Name o Home Location (e.g. node?) o Uni Location (e.g. node?) o Interests (e.g. nodes?) **User actions:** * Simple "registration" (just so that they have a profile) * Create link from one node to another * Create a node (must be "linked to" by another node), such as an interest, event, etc. * Join a node (which is perhaps equivalent to adding an interest) * Add details to a node (e.g. modify description?) * Find other users who have the same interests, live near, etc. * Navigate the node structure, using some kind user interface (including the simple visual rendering of "nearby nodes") * View user profiles

Checklist:

- Registering and keeping track of users and their profiles in the system
- Creating a node (e.g. representing an interest or activity) if it does not already exist, and adding content to that node
- Users "joining" an already existing node thus associating the users with the nodes (e.g. associating a user with an interests or event, and viceversa)
- Linking a node with another node or nodes (e.g. linking playing violin with an interest in music). Perhaps have two styles of links - "stems from" (e.g. playing violin stems from playing music) and "relates to" (e.g. a violin concert is related to an interest in playing the violin)
- Make use of existing keywords (represented as nodes) when new users sign up to facilitate beginning associations
- * Locate other users with similar node associations (such as similar interests, attending similar events)

Notes:

Technical solutions concerned with security do not need to be addressed in this implementation.

The system should ideally be designed to work efficiently for a large number of users (1000+)

Mark Scheme Each criteria should be used to derive a raw mark and then a mark for each criteria is awarded out of 20

Marking Criteria - 1 (worst) to 5 (great)
1 - Poor attempt or no attempt made at the requirements
2 - Some attempt made, but incomplete or non-functional at the requirements
3 - An average to good attempt made at meeting most of the requirements
4 - Very good attempt made with all requirements fulfilled to expectations
5 - Requirements fulfilled beyond expectations, with use of extensions and enhancements

Criteria	**Initial mark by each category /5**	**Overall adjusted/20**
1) User Functionality Recording Registering Logging in Storing information Searching	**/25**	**/20**
2) Node Functionality Creation of nodes Editing nodes Viewing nodes Storing content in nodes	**/20**	**/20**
3) Relationships Creation of relationships Relationships between nodes and other nodes Relationships between users and nodes or other users Browsing through relationships to directly or indirectly related nodes and users Searching through relationships	**/25**	**/20**
4) User Interface Basic user interaction Viewing nodes and relationships Viewing users Context sensitive interface Covers main functionality Low complexity	**/30**	**/20**
5) Coding Style Separation between backend, main logic and frontend Use of comments Good code cohesion Lack of code coupling Room and potential for code extension and enhancement	**/25**	**/20**
	Total Mark:	**/100**