A software tool supporting a constructivist approach to assessing student team work in software development

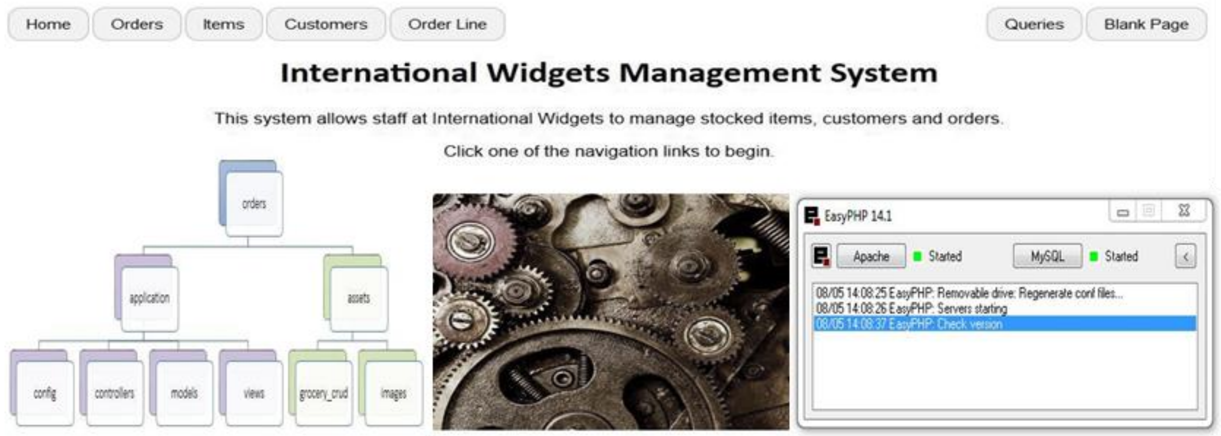
By Helen Partou

h.partou3@herts.ac.uk

and Lindsay Smith

l.1.smith@herts.ac.uk

School of Computer Science



International Widgets Management System

This system allows staff at International Widgets to manage stocked items, customers and orders.

Click one of the navigation links to begin.

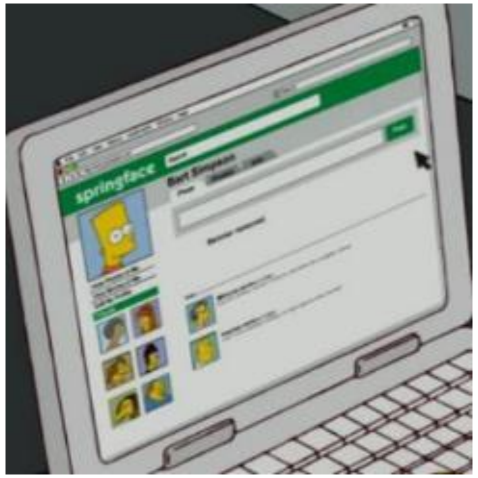
```
08/05 14:08:25 EasyPHP: Removable drive: Regenerate conf files...
08/05 14:08:26 EasyPHP: Servers starting
08/05 14:08:37 EasyPHP: Check version
```

School of Computer Science (SCS) students - ‘Zero to Hero’ in six weeks

- Preparing students for employment in the computing industry
    - Existing software tool (old platform): not ‘fit for purpose’
        - Technical components not scalable for new programming languages
        - Overly complicated to use and difficult to adapt
    - Relevant software engineering experience(s)
        - Context of software development is core delivery
            - Software tool (new platform) piloted in 2015-16 L7 Sem. A module
                - Expanded into L5 modules including online provision in Sem. B
    - Approx. 850+ students have now experienced the platform

Platform structure & demo

- Portable web-based platform
  - Robust: accessible with few unrecoverable technical failures
- Model-View-Controller (MVC) chosen architectural structure
  - Separates database, visual elements and programming interactions
  - ***Facilitates teamwork:*** version control compatible, e.g. Dropbox

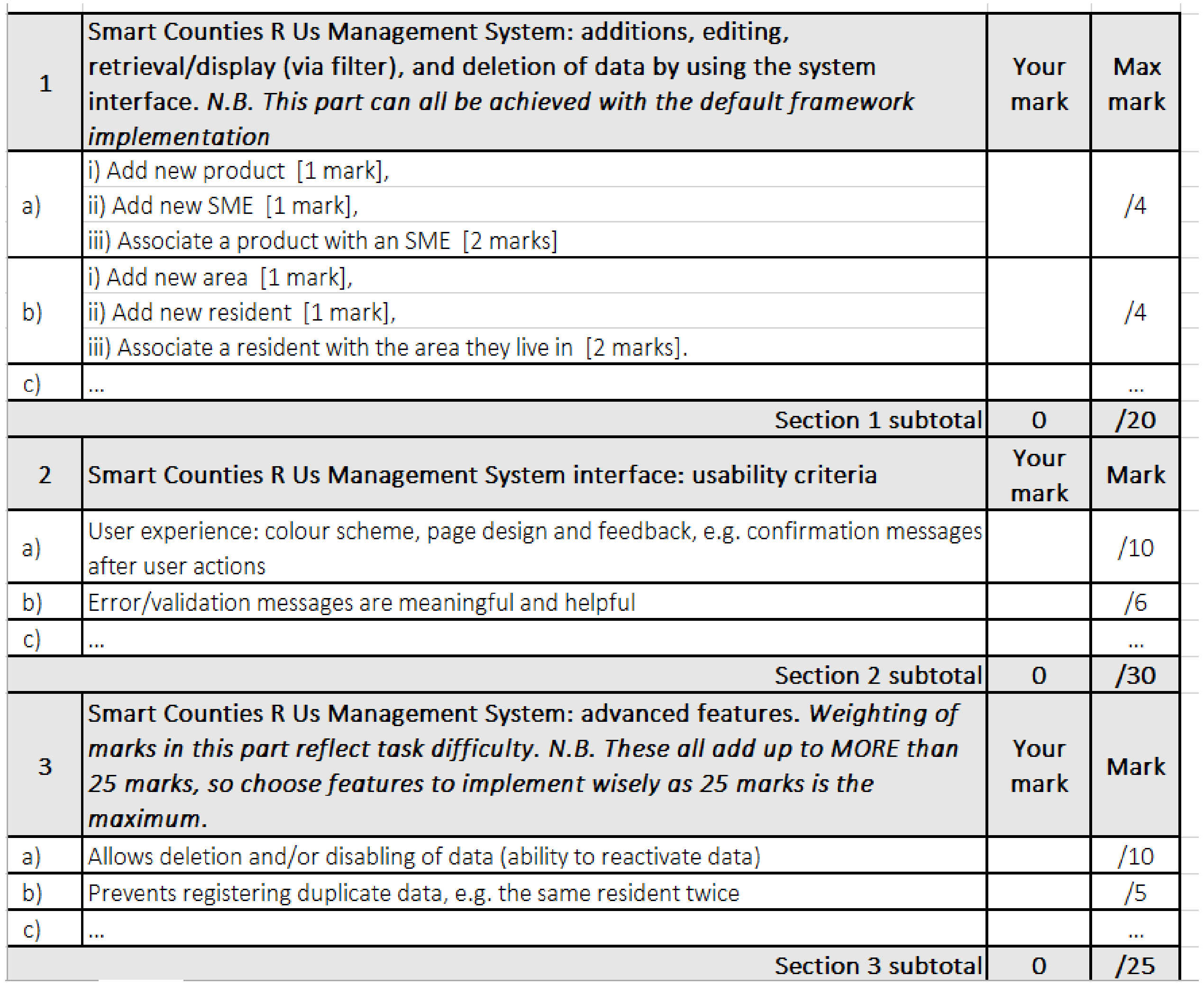


Assessment strategy – solving a problem

- Complexity of software development presents specific educational challenges for SCS students
  - ‘Soft’ skills focus on team working
    - **Passengers** (lack of interest, engagement and/or feeling of inferiority) vs. **diligent isolation** (poor delegation, perfectionism and/or presence of passengers)
  - ‘Hard’ skills focus on technological constraints
    - Time constraints: platform minimises technical complexity for development of solution application
- Industry value: why code in teams?
  - Software developers cannot put graduates ‘in front of a client’ [1]
  - QAA Computing benchmarks: software ‘exposure’ and ‘substantial’ group projects [2]

Assessment marking criteria – guide to team project management

- User Acceptance Tests (UATs)
  - check software is ‘fit for purpose’
- For ***Assessment*** (tutors simulate client) categorised marking criteria
  - ***Baseline*** = minimum engagement for a pass mark
  - ***Advanced*** = independent tasks gain higher marks
- UATs support delegation of tasks to team members



1	**Smart Counties R Us Management System: additions, editing, retrieval/display (via filter), and deletion of data by using the system interface.** ***N.B. This part can all be achieved with the default framework implementation***	**Your mark**	**Max mark**
a)	i) Add new product [1 mark], ii) Add new SME [1 mark], iii) Associate a product with an SME [2 marks]		/4
b)	i) Add new area [1 mark], ii) Add new resident [1 mark], iii) Associate a resident with the area they live in [2 marks].		/4
c)	…		…
	Section 1 subtotal	**0**	**/20**
2	**Smart Counties R Us Management System interface: usability criteria**	**Your mark**	**Mark**
a)	User experience: colour scheme, page design and feedback, e.g. confirmation messages after user actions		/10
b)	Error/validation messages are meaningful and helpful		/6
c)	…		…
	Section 2 subtotal	**0**	**/30**
3	**Smart Counties R Us Management System: advanced features.** ***Weighting of marks in this part reflect task difficulty. N.B. These all add up to MORE than 25 marks, so choose features to implement wisely as 25 marks is the maximum.***	**Your mark**	**Mark**
a)	Allows deletion and/or disabling of data (ability to reactivate data)		/10
b)	Prevents registering duplicate data, e.g. the same resident twice		/5
c)	…		…
	Section 3 subtotal	**0**	**/25**

Constructivism & Instructional Scaffolding

- Platform applications scalable to virtually any ‘real-world’ scenario
  - **Formative**: ‘Orders’ system included in platform
    - Minimises ‘expectation gap’ [3]
  - **Summative**: Olympic games, resourcing school productions, smart tech, etc.
    - Students ‘construct’ ideas

- Instructional Scaffolding:
  - Practical guides, FAQs, demonstration videos and hands-on lab supervision
    - Supports VARK (Visual/Auditory/Read-Write/Kinaesthetic) learning style

Critical reflections

- Future-proofing platform delivery
  - Industry-standard technologies
    - The Cloud, e.g. Git-based tools
  - Proactive planning: staff development time and resources
    - Current platform = 500+ staff hours (conservative estimate)
- Managing student teamwork autonomy
  - Staff familiarity, e.g. level of staff involvement in student teams
- Instructional scaffolding affected by VLE constraints
  - Students can face a challenge accessing teaching resources
- Criticality of case study for platform to support constructivism

References

[1] Matthews, D. (2016) 'What should computer science degree students learn?'. *Times Higher Education*. 10 March 2016. Available at: https://www.timeshighereducation.com/news/what-should-computer-science-degree-students-learn

[2] QAA (Quality Assurance Agency) (2016) *Subject Benchmark Statement: Computing. Available at:* http://www.qaa.ac.uk/publications/information-and-guidance/publication?PubID=3043#.Wx1yVSBG1Pa

[3] Christenson, S., Reschly, A. & Wylie, C. (2012) *Handbook of Research on Student Engagement*. New York: Springer