ELEC5619: OBJECT ORIENTED APPLICATION FRAMEWORKS

Semester 2, 2018 | 6 Credit Points | Mode: Normal-Day
Sessions Valid: Semester 2
Coordinator(s): David Boland

WARNING: This unit version is currently being edited and is subject to change!

1. INTRODUCTION

This unit aims to introduce students to the main issues involved in producing large Internet systems by using and building application frameworks. Frameworks allow great reuse so developers do not have to design and implement applications from scratch, as students have done in ELEC3610 The unit lays down the basic concepts and hands on experience on the design and development of enterprise systems, emphasizing the development of systems using design patterns and application frameworks.

A project-based approach will introduce the problems often found when building such systems, and will require students to take control of their learning. A project-based approach will introduce the problems often found when building such systems, and will require students to take control of their learning. Several development Java frameworks will be used, including Spring, Hibernate, and others. Principles of design patterns will also be studied.

2. LEARNING OUTCOMES

Learning outcomes are the key abilities and knowledge that will be assessed in this unit. See assessment summary table below for details of which outcomes are assessed where. Outcomes are listed according to the course goals that they support.

Design (Level 4)

1. Ability to deploy a large site using a particular application framework to the extent of the material presented.
2. Ability to undertake conception and design by investigating, researching and systematically compiling the various information resources available, in order to deliver an e-business project.

Engineering/IT Specialisation (Level 4)

3. Ability to evaluate the advantages and disadvantages of using domain specific frameworks with a particular focus on web applications.
4. Ability to customize a Content Management System (CMS) to particular user requirements using tools, principles and techniques developed throughout the course.
5. Ability to write and modify Java code using Spring and Hibernate in a web application.
6. Ability to compare and contrast different Web Application Frameworks available in the market using the principles and techniques developed in the course.
7. Ability to assess the differences between using application frameworks for software development and other methods with less reusability.

Communication (Level 3)

8. Capacity to write reports and make presentations to communicate technical and often complex material in clear and concise terms for a specific target audience.

Project and Team Skills (Level 3)

9. Ability to work in a group effectively and efficiently by assuming clearly defined roles and responsibilities and then interacting in a constructive manner with the group by both contributing and evaluating others' viewpoints in order to reach a multilateral agreement on and execution of the solution.

For further details of course goals related to these learning outcomes, see online unit outline at http://cusp.eng.usyd.edu.au/students/view-unit-page/alpha/ELEC5619 .

3. ASSESSMENT TASKS

ASSESSMENT SUMMARY

Assessment name	Team-based?	Weight	Due	Outcomes Assessed
Project	No	40%	Multiple Weeks	1, 2, 3, 4, 5, 8, 9
Mid-Sem Exam	No	30%	Week 9	1, 3, 4, 6, 7
Proposal - Draft	No	10%	Week 5	2, 8, 9
Proposal - final	No	10%	Week 9	8
Presentation	Yes	10%	Week 13	8

ASSESSMENT DESCRIPTION

Project: Prototype, Final application, Presentation, Learning Journal.

Mid-Sem Exam: concepts and programming languages used in the project

Proposal Draft: 1,500 word describing project

Proposal Final: 2,000 word document describing the project, including technical solution and addressing feedback

Presentation: Produce short group video describing project outcomes

ASSESSMENT GRADING

Final grades in this unit are awarded at levels of HD for High Distinction, DI (previously D) for Distinction, CR for Credit, PS (previously P) for Pass and FA (previously F) for Fail as defined by University of Sydney Assessment Policy. Details of the Assessment Policy are available on the Policies website at http://sydney.edu.au/policies . Standards for grades in individual assessment tasks and the summative method for obtaining a final mark in the unit will be set out in a marking guide supplied by the unit coordinator.
High Distinction, HD (85-100)

Show through independent work, ability to use and integrate several services and applications available in the framework.

Distinction, DI (75-84)

Compare WAFs through an analysis of advantages and disadvantages. Be able to relate pros and cons to a business problem. Experiment with the framework's applications and be able to show how they can be exploited to solve a particular business problem. Show consistent high quality programming and communication skills.

Credit, CR (65-74)

Be able to show consistent programming and communication skills

Deliver a working application with quality documentation. Proficiency in Linux development environments.

Pass, PA (50-64)

Showing motivation and initiative. Inconsistent programming and communication skills.

4. ATTRIBUTES DEVELOPED

Attributes listed here represent the course goals designated for this unit. The list below describes how these attributes are developed through practice in the unit. See Learning Outcomes and Assessment sections above for details of how these attributes are assessed.

Attribute	Method
Design (Level 4)	Learn to design a large web application using OO Application frameworks.
Engineering/IT Specialisation (Level 4)	Advanced Java programming skills using and developing frameworks.
Communication (Level 3)	Students write a proposal document and orally present to the class and external assessors.
Project and Team Skills (Level 3)	Students develop a web application in groups of 4-5. Each member is responsible for a component.

For further details of course goals and professional attribute standards, see the online version of this outline at http://cusp.eng.usyd.edu.au/students/view-unit-page/alpha/ELEC5619 .

5. STUDY COMMITMENT

Project Work - own time: Project

Presentation: Final presentations

Tutorial: Tutorial/Labs

Activity	Hours per Week	Sessions per Week	Weeks per Semester
Project Work - in class	3.00	1	7
Project Work - own time	6.00	1	7
Presentation	2.00	1	1
Tutorial	3.00	1	5

Standard unit of study workload at this university should be from 1.5 to 2 hours per credit point which means 9-12 hours for a normal 6 credit point unit of study. For units that are based on research or practical experience, hours may vary. For lecture and tutorial timetable, see University timetable site at: web.timetable.usyd.edu.au/calendar.jsp

6. TEACHING STAFF AND CONTACT DETAILS

COORDINATOR(S)

Name	Room	Phone	Email	Contact note
Dr Boland, David			david.boland@sydney.edu.au	

7. RESOURCES

RECOMMENDED REFERENCES

C. Walls and R. Breidenbach, *Spring in Action*. Manning Publications,

COURSE WEBSITE(S)

http://www.ee.usyd.edu.au/~rafa/elec5619/

NOTE ON RESOURCES

Readings (all are on e-reserve):

* Ralph E. Johnson. ``Frameworks = (components + patterns)``. Communications of the ACM. 40 (10). pp 39-42. (core - mid-term) DOI

* Mohamed Fayad and Douglas C. Schmidt. ``Object-oriented application frameworks``. Communications of the ACM. 40 (10). pp 32-38. (core- mid-term) DOI

* Rod Johnson. ``J2EE Development Frameworks``. IEEE Computer. January 2005. pp 107-110. (core- mid-term)

* Design Patterns: Elements of Reusble Object-Oriented Software. Gamma, Helm, Johnson and Vlissides. Introduction (Chapter 1) (Library) (core - mid-term)

* Maven: a developer`s notebook. Vincent Massol, Timothy M. O`Brien. O`Reilly 2005 - chapter 1 (core- mid-term)

* Chapter 5. Object Oriented Design and Patterns. Cay Horstmann. Wiley, 2002 (library) (core - mid-term).

* Mohamed Fayad, Douglas C. Schmidt., Ralph Johnson. Building Application Frameworks. John Wiley & Sons 1999. (Chapters 1 ad 2) (Library) (core - final)

* Essential Software Architectures. Ian Gorton. Springer 2006 - Chapter 1 (core - final)

* Software Architecture in Practice. Len Bass, Paul Clements and Rick Kazman, Addison-Wesley, 2003 - Chapter 11 ``The ATAM``.

* Ajax in action. Dave Crane, Eric Pascarello with Darren James. [library]

* Crossing Chasms: A pattern Language for Object-RDBMS Integration. K. Brown and B. Whitenack in Pattern Languages of Program Design by Vlissides. 1996. (Library) (informational)

* M. Matera and A. Maurino and S. Ceri and P. Fraternali. ``Model-driven design of collaborative web applications``. Softw. Pract. Exper. 33 (8). pp 701-732. (informational) DOI

8. ENROLMENT REQUIREMENTS

ASSUMED KNOWLEDGE

Java programming, and some web development experience are essential. Databases strongly recommended

PREREQUISITES

None.

9. POLICIES

ACADEMIC HONESTY

While the University is aware that the vast majority of students and staff act ethically and honestly, it is opposed to and will not tolerate academic dishonesty or plagiarism and will treat all allegations of dishonesty seriously.

All students are expected to be familiar and act in compliance with the relevant University policies, procedures and codes, which include:

- *Academic Honesty in Coursework Policy 2015*
- *Academic Honesty Procedures 2016*
- *Code of Conduct for Students*
- *Research Code of Conduct 2013* (for honours and postgraduate dissertation units)

They can be accessed via the University"s Policy Register: http://sydney.edu.au/policies (enter "Academic Honesty" in the search field).

Students should never use document-sharing sites and should be extremely wary of using online “tutor” services. Further information on academic honesty and the resources available to all students can be found on the Academic Integrity page of the University website: http://sydney.edu.au/elearning/student/EI/index.shtml

Academic Dishonesty and Plagiarism

Academic dishonesty involves seeking unfair academic advantage or helping another student to do so.

You may be found to have engaged in academic dishonesty if you:

- Resubmit (or "recycle") work that you have already submitted for assessment in the same unit or in a different unit or previous attempt;
- Use assignment answers hosted on the internet, including those uploaded to document sharing websites by other students.
- Have someone else complete part or all of an assignment for you, or do this for another student.
- Except for legitimate group work purposes, providing assignment questions and answers to other students directly or through social media platforms or document ("notes") sharing websites, including essays and written reports.
- Engage in examination misconduct, including using cheat notes or unapproved electronic devices (e.g., smartphones), copying from other students, discussing an exam with another person while it is in progress, or removing confidential examination papers from the examination venue.
- Engage in dishonest plagiarism.

Plagiarism means presenting another person's work as if it is your own without properly or adequately referencing the original source of the work.

Plagiarism is using someone else's ideas, words, formulas, methods, evidence, programming code, images, artworks, or musical creations without proper acknowledgement. If you use someone's actual words you must use quotation marks as well as an appropriate reference. If you use someone's ideas, formulas, methods, evidence, tables or images you must use a reference. You must not present someone's artistic work, musical creation, programming code or any other form of intellectual property as your own. If referring to any of these, you must always present them as the work of their creator and reference in an appropriate way.

Plagiarism is always unacceptable, regardless of whether it is done intentionally or not. It is considered dishonest if done knowingly, with intent to deceive or if a reasonable person can see that the assignment contains more work copied from other sources than the student's original work. The University understands that not all plagiarism is dishonest and provides students with opportunities to improve their academic writing, including their understanding of scholarly citation and referencing practices.

USE OF SIMILARITY DETECTION SOFTWARE

All written assignments submitted in this unit of study will be submitted to the similarity detecting software program known as **Turnitin**. Turnitin searches for matches between text in your written assessment task and text sourced from the Internet, published works and assignments that have previously been submitted to Turnitin for analysis.

There will always be some degree of text-matching when using Turnitin. Text-matching may occur in use of direct quotations, technical terms and phrases, or the listing of bibliographic material. This does not mean you will automatically be accused of academic dishonesty or plagiarism, although Turnitin reports may be used as evidence in academic dishonesty and plagiarism decision-making processes.

Computer programming assignments may also be checked by specialist code similarity detection software. The Faculty of Engineering & IT currently uses the MOSS similarity detection engine (see http://theory.stanford.edu/~aiken/moss/) . These programs work in a similar way to TII in that they check for similarity against a database of previously submitted assignments and code available on the internet, but they have added functionality to detect cases of similarity of holistic code structure in cases such as global search and replace of variable names, reordering of lines, changing of comment lines, and the use of white space.

See the policies page of the faculty website at http://sydney.edu.au/engineering/student-policies/ for information regarding university policies and local provisions and procedures within the Faculty of Engineering and Information Technologies.

10. WEEKLY SCHEDULE

Note that the "Weeks" referred to in this Schedule are those of the official university semester calendar https://web.timetable.usyd.edu.au/calendar.jsp

Week	Topics/Activities
Week 1	Introduction: About this course
	Design Patterns
Week 2	Spring
Week 3	Spring
Week 4	Maven
	Hibernate
Week 5	Maven
	Hibernate
	Assessment Due: Proposal - Draft
Week 6	Architecture Evaluations
Week 7	3. setup and configuration
	2. architecture,
	1. Introduction
	Sakai Introduction:
Week 8	Mid Term Exam (30%) - in lecture
Week 9	3. security
	Presentation Layer Frameworks: JSP, JSF, RSF
	Sakai persistance and security
	1. data storage models
	2. entities
	Assessment Due: Mid-Sem Exam
	Assessment Due: Proposal - final
Week 10	Application Frameworks
Week 11	Ajax architecture
Week 12	WebML and CASE tools (Webratio)
	Test Driven Development with JUnit4 and Mockito
Week 13	Text Mining and The Semantic Web
	Course Closure
	Assessment Due: Presentation