

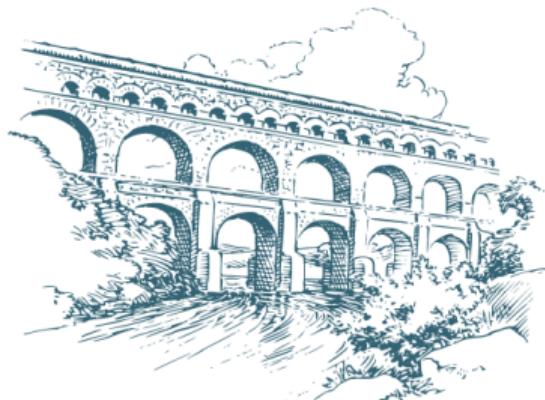
# Course Overview

## *Software Architecture*

Richard Thomas

February 23, 2026

*University of Queensland*



## What is the course about?

- Well, *software architecture*

## What is the course about?

- Well, *software architecture*
- Designing and building software systems

## What is the course about?

- Well, *software architecture*
- Designing and building software systems
  - Multiple *software components* that work together

## What is the course about?

- Well, *software architecture*
- Designing and building software systems
  - Multiple *software components* that work together
- Using *architecture patterns* to structure software systems to be *maintainable*

## What is the course about?

- Well, *software architecture*
- Designing and building software systems
  - Multiple *software components* that work together
- Using *architecture patterns* to structure software systems to be *maintainable*
- How to build software that is *reliable* and *fault tolerant*

## What is the course about?

- Well, *software architecture*
- Designing and building software systems
  - Multiple *software components* that work together
- Using *architecture patterns* to structure software systems to be *maintainable*
- How to build software that is *reliable* and *fault tolerant*
- How to build software that is *scalable*

# What will we be doing?

## Lectures

- Learn common *architecture patterns*

## Case Studies

## Practicals

# What will we be doing?

## Lectures

- Learn common *architecture patterns*
- Learn tools and techniques for *designing* and *implementing* software systems

## Case Studies

## Practicals

# What will we be doing?

## Lectures

- Learn common *architecture patterns*
- Learn tools and techniques for *designing* and *implementing* software systems
- Learn the principles for working with *distributed systems*

## Case Studies

## Practicals

# What will we be doing?

## Lectures

- Learn common *architecture patterns*
- Learn tools and techniques for *designing* and *implementing* software systems
- Learn the principles for working with *distributed systems*

## Case Studies

- Work on *case studies* that implement architectural patterns

## Practicals

# What will we be doing?

## Lectures

- Learn common *architecture patterns*
- Learn tools and techniques for *designing* and *implementing* software systems
- Learn the principles for working with *distributed systems*

## Case Studies

- Work on *case studies* that implement architectural patterns
- Hands-on practice with the tools and techniques for *designing* and *implementing* software systems

## Practicals

# What will we be doing?

## Lectures

- Learn common *architecture patterns*
- Learn tools and techniques for *designing* and *implementing* software systems
- Learn the principles for working with *distributed systems*

## Case Studies

- Work on *case studies* that implement architectural patterns
- Hands-on practice with the tools and techniques for *designing* and *implementing* software systems

## Practicals

- Develop stateless and persistent *RESTful web APIs*

# What will we be doing?

## Lectures

- Learn common *architecture patterns*
- Learn tools and techniques for *designing* and *implementing* software systems
- Learn the principles for working with *distributed systems*

## Case Studies

- Work on *case studies* that implement architectural patterns
- Hands-on practice with the tools and techniques for *designing* and *implementing* software systems

## Practicals

- Develop stateless and persistent *RESTful web APIs*
- Package software components into *Docker* containers

# What will we be doing?

## Lectures

- Learn common *architecture patterns*
- Learn tools and techniques for *designing* and *implementing* software systems
- Learn the principles for working with *distributed systems*

## Case Studies

- Work on *case studies* that implement architectural patterns
- Hands-on practice with the tools and techniques for *designing* and *implementing* software systems

## Practicals

- Develop stateless and persistent *RESTful web APIs*
- Package software components into *Docker* containers
- Deploy containers to cloud platforms using *Terraform*

# What will we be doing?

## Lectures

- Learn common *architecture patterns*
- Learn tools and techniques for *designing* and *implementing* software systems
- Learn the principles for working with *distributed systems*

## Case Studies

- Work on *case studies* that implement architectural patterns
- Hands-on practice with the tools and techniques for *designing* and *implementing* software systems

## Practicals

- Develop stateless and persistent *RESTful web APIs*
- Package software components into *Docker* containers
- Deploy containers to cloud platforms using *Terraform*
- Use cloud platform tools to *monitor* and *scale* applications

# *§ Assessment*

# Assessment

Cloud Infrastructure Assignment 40%

    API Functionality 10%

    Deployed to Cloud 10%

    Scalable Application 20%

Architecture Presentation 30%

Capstone Project 30%

(Delivering Quality Attributes Project)

# *Cloud Infrastructure*

1. Build a *RESTful web API* according to our specification
2. *Test* that the API satisfies the specification
3. *Deploy* the API to a cloud platform
4. *Scale* the API to handle *variable* and *high* loads

## *Capstone Project*

1. Teams will be allocated a project
2. *Design* and *implement* the project

# *Architecture Presentation*

- Team presents details of project architecture.
  - *Everyone* presents.
- *Individuals* present on different sets of questions
  - Compare and contrast with another architectural pattern
  - Pros and cons of architecture
  - Implementation characteristics of design
  - Potential security risks of architecture
- *Everyone* is expected to understand entire architecture
  - Questions can be directed to *anyone*

# *§ You and Us*

## *Who are we?*



Richard Thomas



Guangdong Bai



Vy Ho



Zaidul Alam



Thuy Dao



Nimesh Garg

*Question*

Who are *you*?

# *Course Website*

All course material is hosted on the course website

- <https://csse6400.uqcloud.net>

If you find any *errors* or wish to suggest any *improvements*,  
please submit a pull request on GitHub

- <https://github.com/CSSE6400/software-architecture>

# GitHub Username Registration Form: 6pm on Feb. 23<sup>1</sup>

You need access to the CSSE6400 organisation on GitHub

- *Practicals* – Access to code
- *Assessment* – Most submissions



<https://tiny.cc/csse6400reg>

---

<sup>1</sup>Yes, that is *today*