# COMP1531

2.1 - Requirements

# Updates

- Project released
- More resources on Webcms3

# Lab reminder

Don't forget that lab submissions are in two parts:

1. Submission of the lab via "1531 submit" on Sunday 5pm the week it was released
2. Getting it checked off in-person (manually) with your tutor in the lab the week it was released, or the week after

You must complete both of these to be awarded the marks

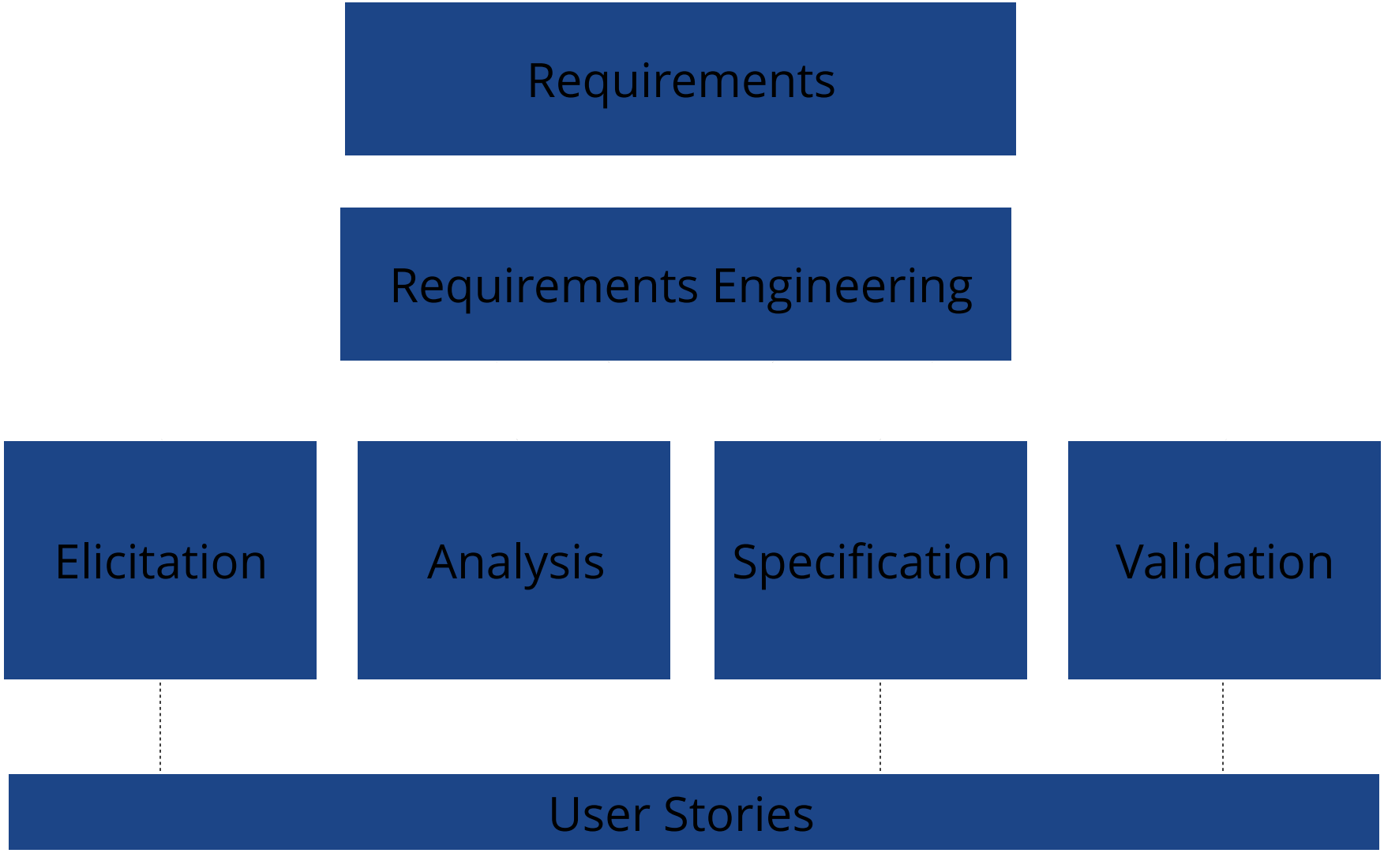
packages / server / src / config.ts

```
@@ -121,12 +121,12 @@ const config: ServerConfig = {
      secret: envVars.JWT_SECRET || '123456789',
    },
    db: {
-     hostname: envVars.DB_HOSTNAME || 'localhost',
-     username: envVars.DB_USERNAME || 'pearler',
-     database: envVars.DB_DATABASE || 'pearler',
-     password: envVars.DB_PASSWORD || 'pearler',
+     hostname: envVars.DB_HOSTNAME || 'pearler-v[redacted]southeast-2.rds.amazonaws.com',
+     username: envVars.DB_USERNAME || 'pearler2',
+     database: envVars.DB_DATABASE || 'test',
+     password: envVars.DB_PASSWORD || '[redacted]',
    },
-   virtualMarket: (envVars.VIRTUAL_MARKET || 'true') === 'true',
+   virtualMarket: false, //(envVars.VIRTUAL_MARKET || 'true') === 'true',
    env_local: (envVars.ENV_NAME || 'local') === 'local',
    env_dev: (envVars.ENV_NAME || 'local') === 'dev',
    env_test: (envVars.ENV_NAME || 'local') === 'test',
@@ -154,11 +154,11 @@ const config: ServerConfig = {
      advisorCode: envVars.OM_ADVISOR_CODE || 'TB2A1',
      urls: {
        auth: envVars.OM_URL_AUTH || 'auth.openmarkets.com.au',
-       market: envVars.OM_URL_MARKET || 'test-market-data-api.openmarkets.com.au',
-       orders: envVars.OM_URL_ORDERS || 'test-oms-api.openmarkets.com.au',
-       news: envVars.OM_URL_NEWS || 'test-news-api.openmarkets.com.au',
-       clients: envVars.OM_URL_CLIENTS || 'openmarkets-test.apigee.net',
-       connect: envVars.OM_URL_CONNECT || 'openmarkets-test.apigee.net',
+       market: envVars.OM_URL_MARKET || 'market-data-api.openmarkets.com.au',
+       orders: envVars.OM_URL_ORDERS || 'oms-api.openmarkets.com.au',
+       news: envVars.OM_URL_NEWS || 'news-api.openmarkets.com.au',
+       clients: envVars.OM_URL_CLIENTS || 'openmarkets-prod.apigee.net',
+       connect: envVars.OM_URL_CONNECT || 'openmarkets-prod.apigee.net',
      },
    },
    digital_id: {
```

# SDLC


REQUIREMENTS ANALYSIS
DESIGN
SDLC PHASES
MAINTENANCE
DEVELOPMENT
DEPLOYMENT
TESTING

# Requirements



# Requirements

IEEE defines a requirement as:

**A condition or capability needed by a user to solve a problem or achieve an objective**

We would also describe requirements as:

- Agreement of work to be completed by all stakeholders
- Descriptions and constraints of a proposed system

# Functional v Non-Functional

**Functional requirements** specify a specific capability/service that the system should provide.

**Non-functional requirements** place a constraint on *how* the system can achieve that. Typically this is a performance characteristic.

# Functional v Non-Functional

**For example:**

Functional: The system must send a notification to all users whenever there is a new post, or someone comments on an existing post

Non-functional: The system must send emails no later than 30 minutes after from such an activity

# Requirements Engineering

We need a durable process to determine requirements

*“The hardest single part of building a software system is deciding what to build. No part of the work so cripples the resulting systems if done wrong” (Brooks, 1987)*

# Requirements Engineering

Requirements Engineering is:

- A **set of activities** focused on identifying the purpose and goal of a software system
- A **negotiation process** where stakeholders agree on what they want. Stakeholders include:
  - End user(s)
  - Client(s) (often businesses)
  - Design team(s)

# Requirements Engineering

Requirements engineering often follows a logical process across 4 steps:

1. Elicitation of raw requirements from stakeholders
2. Analysis of requirements
3. Formal specification of requirements
4. Validation of requirements

# RE | Step 1 | Elicitation

**Questions and discovery**

- Market Research
- Interviews with Stakeholders
- Focus groups
- Asking questions "What if? What is?"

# RE | Step 2 | Analysis

**Building the picture**

- Identify dependencies, conflicts, risks
- Establish relative priorities
- Usually done through:
  - User stories (discussed today)
  - Use cases (discussed next week)

# RE | Step 3 | Specification

**Refining the picture**

- Establishing the right sense of granularity
  - There is no perfect way to granulate
- Often the stage of breaking up into functional and non-functional
- E.G. Try and granulate "The system shall keep the door locked at all times, unless instructed otherwise by an authorised user.  When the lock is disarmed, a countdown shall be initiated at the end of which the lock shall be automatically armed (if still disarmed)"

# RE | Step 4 | Validation

Going back to stakeholders and ensuring requirements are correct

# Challenges during RE?

What are some challenges we may face while engaging in Requirements engineering?

# Challenges during RE?

What are some challenges we may face while engaging in Requirements engineering?

- Requirements sometimes only understood after design/build has begun
- Clients/customers sometimes don't know what they want
- Clients/customers sometimes change their mind
- Developers might not understand the subject domain
- Limited access to stake holders
- Jumping into details or solutions too early (XY problem)

# What matters?

- Investigate stakeholder needs
- Expand, refine, and connect *specific* ideas
- Understand the iterative and ongoing nature
  - Humans are imperfect

# Let's step through an example