2.7 Pengujian T-Test

Pada penelitian ini digunakan uji T untuk menentukan apakah hipotesis yang sudah ditentukan diterima atau ditolak. Untuk menguji hipotesis diperlukan panduan daftar tabel t hitung dan pengujian dilakukan dengan melakukan penghitungan dengan rumus:

$$t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}}$$

Dimana :

t = nilai uji T

r = hasil perhitungan korelasi *Product Moment Pearson*

BAB III

LANDASAN TEORI

3.1 *Agile Software Development*

Istilah Agile sendiri terdiri dari dua pengertian, yaitu: pertama pengertian dari segi filosofi, dan kedua pengertian dari segi pedoman pengembangan perangkat lunak. Dari segi filosofi, agile mempunyai arti antara lain: mendorong demi terciptanya kepuasan pelanggan; mempercepat *delivery* perangkat lunak secara bertahap (incremental); tim proyek yang ramping dan mempunyai motifasi yang sangat tinggi; minimasi pekerjaan; serta menyederhanakan (birokrasi) keseluruhan proses

pembangunan perangkat lunak. Sedangkan dari segi pedoman pengambangan perangkat lunak, agile mempunyai pengertian, bahwa secara aktif dan berkesinambungan, antara pengembang dengan pelanggan harus senantiasa menjalin kerjasama dan komunikasi dengan baik (Scrum.co.id).

Agile Software Development merupakan salah satu dari beberapa metode yang digunakan dalam pengembangan *software*. *Agile Software Development* adalah jenis pengembangan sistem jangka pendek yang memerlukan adaptasi cepat dan pengembang terhadap perubahan dalam bentuk apapun. Dalam *Agile Software Development* interaksi dan personel lebih penting dari pada proses dan alat, software yang berfungsi lebih penting daripada dokumentasi yang lengkap, kolaborasi dengan klien lebih penting dari pada negosiasi kontrak, dan sikap tanggap terhadap perubahan lebih penting daripada mengikuti rencana.

Agile Software Development juga dapat diartikan sekelompok metodologi pengembangan software yang didasarkan pada prinsip-prinsip yang sama atau pengembangan system jangka pendek yang memerlukan adaptasi cepat dari pengembang terhadap perubahan dalam bentuk apapun (sumber: Wikipedia).

3.1.1 Prinsip Agile

Agile Software Development juga melihat pentingnya komunikasi antara anggota tim, antara orang-orang teknis

dan businessmen, antara developer dan managernya. Ciri lain adalah klien menjadi bagian dari tim pembangun software. Ciri-ciri ini didukung oleh 12 prinsip yang ditetapkan oleh Agile Alliance, yaitu:

1. Kepuasan klien adalah prioritas utama dengan menghasilkan produk lebih awal dan terus menerus.
2. Menerima perubahan kebutuhan, sekalipun diakhir pengembangan.
3. Penyerahan hasil/*software* dalam hitungan waktu beberapa minggu sampai beberapa bulan.
4. Pihak bisnis dan pengembang harus bekerja sama setiap hari selama pengembangan berjalan.
5. Membangun proyek dilingkungan orang-orang yang bermotivasi tinggi yang bekerja dalam lingkungan yang mendukun dan yang dipercaya untuk dapat menyelesaikan proyek.
6. Komunikasi dengan berhadapan langsung adalah komunikasi yang efektif dan efisien
7. Software yang berfungsi adalah ukuran utama dari kemajuan proyek
8. Dukungan yang stabil dari sponsor, pembangun, dan pengguna diperlukan untuk menjaga perkembangan yang berkesinambungan
9. Perhatian kepada kehebatan teknis dan desain yang bagus meningkatkan sifat agile
10. Kesederhanaan penting
11. Arsitektur, kebutuhan dan desain yang bagus muncul dari tim yang mengatur dirinya sendiri

12. Secara periodik tim evaluasi diri dan mencari cara untuk lebih efektif dan segera melakukannya.

Kedua belas prinsip tersebut menjadi suatu dasar bagi model-model proses yang punya sifat agile. Beberapa jenis proses permodelan yang termasuk kedalam metode *Agile Software Development* : Extreme Programming (XP), Adaptif Software Development (ASD), Dynamic System Development Method (DSDM), Scrum, Crystal, Feature Driven Development (FDD), Agile Modeling (AM).

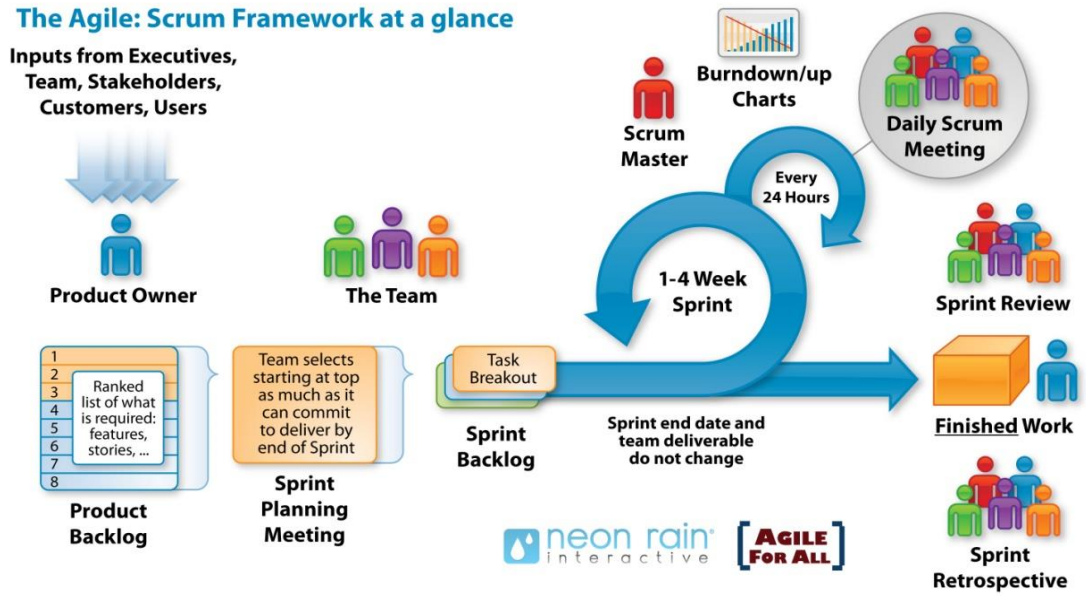
3.1.2 Kelebihan Agile

1. Meningkatkan kepuasan kepada klien
2. Pembangunan system dibuat lebih cepat
3. Mengurangi resiko kegagalan implementasi software dari segi non-teknis
4. Jika pada saat pembangunan system terjadi kegagalan,kerugian dari segi materi relative kecil.

3.1.3 Proses Agile

Agile Process merupakan sekelompok aktifitas pembangunan perangkat lunak secara iteratif yang menekankan pada aktifitas konstruksi (desain dan koding). Agile Process mengeliminasi sebagian besar waktu untuk melakukan perencanaan sistem dan berusaha sebisa mungkin mematuhi jadwal *delivery* sistem yang telah dijanjikan. Requirements yang dibutuhkan secara langsung di-*drive* oleh pelanggan itu sendiri, dan apabila terjadi perubahan

terhadap *requirements* tersebut, pengembang dituntut mampu beradaptasi dengan perubahan yang terjadi. Berikut merupakan salah satu proses dari model Agile Software Development, yaitu scrum:



Sumber: http://tech.pristine.io/

Gambar 3.1 *Agile Work Cycle*

1. First meeting

Proses scrum diawali dengan pembuatan tujuan yang akan dicapai dan penentuan product backlog. Product backlog dikuantisasi waktu dengan satuan hari (antara 1-20 hari). Product backlog merupakan kombinasi antara story-based work (pekerjaan yang berbasis use case/product feature) dan task-based work. Misalnya, "Tambahkan validasi pada semua form". Product backlog

diprioritaskan oleh product owner. Product backlog berisi list yang diprioritaskan dari fitur-fitur atau perubahan yang akan ada pada produk.

2. Sprint planning meeting

Merupakan meeting untuk product owner, scrum team dan orang-orang yang berkepentingan. Dalam meeting ini menentukan sprint goal yaitu tujuan yang ingin dicapai pada scrum sprint berikutnya (30 hari kedepan). Sprint goal biasa adalah minimum fungsionalitas yang harus dicapai. Jika sprint goal tidak dicapai maka dilakukan abnormal termination. Dalam sprint planning meeting ini juga membuat sprint backlog yaitu list dari pekerjaan yang akan dilakukan selama sprint. Sprint backlog merupakan bagian produck backlog yang didetailkan. Sprint backlog dikuantisasi waktu berdasarkan jam (bukan hari yaitu antara 1-16). Sprint backlog harus transparan untuk semua orang dalam tim. Meeting ini tidak lebih dari atau 8 jam saja. Dengan 4 jam pertama adalah waktu yang digunakan untuk Product Owner menjelaskan atau presentasi tentang prioritas dari product backlog. Kemudian tanya jawab dari tim tetang isi, maksud, tujuan dari item yang ada di product backlog. Empat jam berikutnya adalah sesi untuk tim merencanakan Sprint. Fokusnya pada melakukan pekerjaan bukan berfikir mengenai bagaimana mengerjakannya.

3. Daily Scrum meeting (Inspect and adapt cycle)

Meeting ini merupakan meeting harian selama tidak lebih dari 15 menit yang di dalamnya hanya sekedar sharing apa yang sudah dilakukan kemarin, sekarang dan rencana untuk besok. Yang boleh bicara dalam tim ini adalah scrum master dan anggota tim (*developer*). Orang lain yang bekepentingan dapat ikut dalam tim tetapi tidak boleh berkomunikasi (berbicara). Biasanya Scrum master menanyakan 3 pertanyaan dari kepada anggota tim. Pertanyaannya adalah sebagai berikut :

1. Apa yang sudah kamu lakukan kemarin (selama 24 jam kebelakang)?
2. Apa yang akan dikerjakan pada esok hari (24 jam mendatang)?
3. Hal apa yang bisa menghentikan pekerjaan besok hari (kendala)?

4. Sprint review meeting

Meeting ini merupakan meeting setelah aktivitas selama 2 minggu atau 1 bulanan (Sprint) berakhir, yang kemudian diikuti oleh sprint planning meeting untuk Sprint berikutnya. Meeting ini mereview atas Sprint yang sudah dilaksanakan kemudian memperbarui (update) sprint backlog yang merefleksikan berapa lama waktu yg dibutuhkan untuk menyelesaikan pekerjaan (task)

5. Sprint retrospective meeting

Meeting ini merupakan meeting setelah sprint review meeting dan sebelum sprint planning meeting berikutnya. Meeting ini merupakan meeting yang dihadiri Scrum Master

dan tim developer untuk merevisi proses dan cara kerja Scrum, proses development agar Sprint berikutnya lebih efektif dan enjoyable.

Mengapa agile dibutuhkan?

Sebelum mengetahui mengapa agile penting dalam proyek pengembangan perangkat lunak, kita perlu mengetahui beberapa alasan mengapa banyak proyek yang tidak sukses. Seperti sudah disebutkan pada latar belakang masalah adalah alasan pertama yakni sedikitnya keterlibatan user atau customer dalam proyek. Dalam beberapa kasus yang menggunakan metode pengembangan waterfall, user hanya terlibat pada fase analisis dan testing saja. Sehingga, kebanyakan user terkadang menerima begitu saja hasil yang diberikan oleh developer dengan sedikit sekali testing terhadap perangkat lunaknya.

Alasan kedua yakni requirement yang 'buruk'. Beberapa hal yang mungkin perlu kita jadikan catatan bahwa mustahil untuk mengumpulkan requirement yang tepat di awal-awal project dan semua requirement yang telah berhasil dikumpulkan tidak bisa dijamin untuk tidak berubah. Alasan lain yang menyebabkan buruknya requirement adalah karena user atau customer kadang tidak tahu apa yang mereka butuhkan.

Alasan ketiga yakni schedule atau jadwal yang tidak realistis, terkadang jadwal tidak disepakati dari pihak-pihak yang terlibat dalam pengembangan, seperti tim yang menganalisa project terkadang memutuskan jadwal

proyek akan selesai dalam waktu tertentu yang terkadang menjadi beban bagi tim yang mengembangkan perangkat lunaknya(tim developer). Tim developer terkadang tidak memiliki cukup waktu untuk mengembangkan perangkat lunak sehingga kadang membuat keterlambatan dalam penyerahan perangkat lunak yang jadi.

Alasan keempat yakni sedikitnya testing. Pada banyak metode klasik, testing dilakukan di akhir-akhir fase, sehingga sangat sedikit testing yang dilakukan. Sedikitnya testing ini bisa mengakibatkan kemungkinan ditemukannya bug dan perbaikan dalam sistem sangatlah kecil. Hal ini mungkin menguntungkan bagi Developer, akan tetapi bisa berakibat buruk pada masa perawatan system setelah system diproduksi.

Alasan kelima yakni minimnya adaptasi dengan perubahan requirement. Terkadang rencana bisa berubah di tengah jalan, rencana yang tadinya seharusnya lurus bisa jadi berliku-liku seiring dengan berubahnya permintaan customer. Apa yang seharunya dikerjakan nanti belakangan, terkadang ditemukan penyelesaiannya terlebih dahulu. Serta masih banyak lagi alasan-alasan lainnya yang sering membuat sebuah project mengalami kegagalan.

Apa hubungannya dengan agile? Secara teori agile dapat menanggulangi masalah-masalah ini berdasarkan dari prinsip-prinsip agile di atas.

Beberapa keunggulan agile:

a. Proses *Iterative* dan *Incremental*.

b. Requirement dapat berubah sewaktu-waktu.
c. Pelacakan requirement dengan melihat Backlog produk.
d. Adanya keterlibatan user secara aktif.
e. Rilis yang lebih cepat dan berkala, fungsi dirilis setiap akhir iterasi.
f. Testing dilakukan setiap saat.

Terdapat beberapa faktor yang dapat mempengaruhi kesuksesan sebuah proyek dengan metode agile. Menurut Tsun dan Dac-Buu (2007) Menyatakan bahwa faktor kesuksesan proyek agile dibagi dalam 4 kategori, yaitu *Organizational, People, Process* dan *Technical*. Dimana faktor-faktornya dapat dilihat sebagai berikut:

Tabel 3.1 Faktor yang Mempengaruhi Kesuksesan *Agile Software Development*

Kategori	Faktor
Organizational	- *Management Commitment* - *Organizational Environment* - *Team Environment*
People	- *Team Capability* - *Customer Involvement*
Process	- *Project Management Process* - *Project Definition Process*
Technical	- *Agile Software Techniques* - *Delivery Strategy*

Faktor-faktor ini berhubungan dengan bagaimana orang-orang yang terlibat di dalamnya menjalani proses proyek itu selama proses agile itu berjalan.

Dalam metode agile kolaborasi sebuah tim merupakan hal yang penting dalam mencapai kesuksesan proyek. Kolaborasi itu sendiri memerlukan suatu kinerja tim yang baik selama proyek berjalan. Faktor baik atau tidaknya kinerja tim agile itu sendiri dibagi menjadi empat kategori utama, yaitu *Team Composition*, *Team Support*, *Team Management and Structure* dan *Team Communication* (Ecless, dkk, 2010), dimana faktor-faktornya dapat dilihat pada tabel berikut:

Tabel 3.2 Faktor Kinerja Tim Agile

Kategori Utama	Faktor
Team Composition	- Peran - Komposisi Tim - Individualitas
Team Support	- Sumber Daya - Moral - Kepercayaan - Fun - Komitmen
Team Management & Structure	- Manajemen Konflik - Goal - Kinerja Target - Pendekatan Pekerjaan

Team Communication	- Komunikasi - *Feedback*

3.2 Myers-Briggs Type Indicator (MBTI)

Asesmen Myers-Briggs Type Indicator (MBTI) adalah kuesioner psikometri yang dirancang untuk mengukur preferensi psikologis bagaimana seseorang memandang dunia dan membuat keputusan. Preferensi ini didasarkan dari teori tipologis diusulkan oleh Carl Gustav Jung dan pertama kali diterbitkan pada tahun 1921 dalam buku *Psychological Types*. Jung berteori bahwa ada empat fungsi psikologis utama yang kita mengalami dunia: Sensasi, intuisi, perasaan, dan pikiran. Salah satu dari keempat fungsi ini akan ada yang lebih dominan.

Para pengembang original dari inventori kepribadian seperti Katharine Cook Briggs dan putrinya, Isabel Briggs Myers. Keduanya, setelah mempelajari secara ekstensif karya Jung, ternyata minat mereka tentang perilaku manusia diabdikan untuk membuat teori psikologis dapat digunakan secara praktis. Mereka mulai menciptakan indikator selama Perang Dunia II. Mereka percaya bahwa pengetahuan tentang preferensi kepribadian akan membantu perempuan yang memasuki pekerjaan di bidang industri pertama kalinya untuk mengidentifikasi jenis pekerjaan selama perang dimana mereka akan merasa "paling nyaman dan efektif". Kuesioner pertama adalah *Type Indicator*

Myers-Briggs, yang diterbitkan pada tahun 1962. MBTI berfokus pada populasi normal dan menekankan nilai perbedaan yang alami. Robert Kaplan dan Dennis Saccuzzo percaya bahwa "asumsi yang mendasari MBTI adalah bahwa kita semua memiliki preferensi tertentu dalam cara kita menafsirkan pengalaman kita, dan ini mendasari preferensi minat, kebutuhan, nilai-nilai, dan motivasi".

3.2.1 Konsep

Sebagai panduan MBTI, indikatornya adalah "dirancang untuk menerapkan teori, karena teori harus dipahami untuk memahami MBTI". Pokok dari Type Indicator Myers-Briggs adalah teori tipe psikologis seperti yang dikembangkan oleh Carl Jung. Jung mengusulkan adanya dua pasang dikotomi (pembagian) fungsi kognitif:

1. Fungsi "rasional" (*judging*) : pikiran dan perasaan
2. Fungsi "irasional" (*perceiving*) : *sensing* dan intuisi

Jung percaya bahwa untuk setiap orang masing-masing fungsi utama yang dinyatakan baik dalam bentuk introvert atau extravert. Dari konsep asli Jung, Briggs dan Myers mengembangkan teori mereka sendiri mengenai tipe psikologis.

a. Tipe

Model tipologis Jung menganggap tipe psikologis mirip dengan wenangan kiri atau kanan: orang lahir dengan, atau

mengembangkan, memiliki cara dan pilihan tertentu untuk berpikir dan bertindak. Jenis MBTI beberapa diantaranya berbeda secara psikologis yang kemudian menjadi empat pasang berlawanan, atau dikotomi, dengan 16 jenis kemungkinan tipe psikologis. Tak ada jenis yang lebih baik atau lebih buruk, namun, Briggs dan Myers berteori bahwa individu secara alami lebih memilih satu kombinasi dari keseluruhan jenis perbedaan. Dengan cara yang sama, menulis dengan tangan kiri adalah kerja keras untuk orang yang menulis dengan tangan kanan, sehingga orang-orang cenderung untuk menemukan menggunakan preferensi psikologis yang lebih sulit, bahkan jika mereka merasa bisa menjadi lebih mahir (perilaku fleksibel) dengan praktek dan pengembangan.

16 tipe ini biasanya disebut dengan singkatan dari empat huruf-huruf awal dari masing-masing preferensi mereka berdasarkan empat tipe (kecuali dalam kasus intuisi, yang menggunakan singkatan N untuk membedakannya dari Introversi). Misalnya:

- ESTJ: extraversion (E), penginderaan (S), pemikiran (T), penghakiman (J)
- INFP: introvert (I), intuisi (N), perasaan (F), persepsi (P)
- ESFP: extraversion (E), penginderaan (S), perasaan (F), persepsi (P)
- INTJ: introvert (I), intuisi (N), pemikiran (T), penghakiman (J)

- ENTP: extraversion (E), intuisi (N), pemikiran (T), persepsi (P)
- ISFJ: introvert (I), penginderaan (S), perasaan (F), penghakiman (J)

  Dan seterusnya untuk semua 16 jenis kombinasi yang mungkin.

b. Empat Dikotomi

Dichotomies			
Extraversion	(**E**)	–	(**I**) Introversion
Sensing	(**S**)	–	(**N**) Intuition
Thinking	(**T**)	–	(**F**) Feeling
Judging	(**J**)	–	(**P**) Perception

Ket : Empat pasang preferensi atau dikotomi ditunjukkan dalam tabel di atas.

1. Extrovert (E) vs. Introvert (I) (Aspek Sikap).

Ekstrovert artinya tipe pribadi yang suka bergaul, menyenangi interaksi sosial dengan orang lain, dan berfokus pada the world outside the self. Sebaliknya tipe introvert adalah mereka yang senang menyendiri, reflektif, dan tidak begitu suka bergaul dengan banyak orang. Orang introvert lebih suka mengerjakan aktivitas yang tidak banyak menutut interaksi semisal membaca, menulis, dan berpikir secara imajinatif.

2. ***Sensing (S) vs. Intuitive (N (Aspek Fungsi).*** Tipe dikotomi kedua ini melihat bagaimana seseorang memproses data. Sensing memproses data dengan cara bersandar pada fakta yang konkrit, factual facts, dan melihat data apa adanya. Sensing adalah concrete thinkers. Sementara tipe intuitive memproses data dengan melihat pola dan impresi, serta melihat berbagai kemungkinan yang bisa terjadi. Intutive adalah abstract thinkers.
3. ***Thinking (T) vs. Feeling (F)(Aspek Fungsi).*** Tipe dikotomi yang ketiga ini melihat bagaimana orang berproses mengambil keputusan. Thinking adalah mereka yang selalu menggunakan logika dan kekuatan analisa untuk mengambil keputusan. Sementara feeling adalah mereka yang melibatkan perasaan, empati serta nilai-nilai yang diyakini ketika hendak mengambil keputusan.
4. ***Judging (J) vs. Perceiving (P)(Aspek Gaya Hidup).*** Tipe dikotomi yang terakhir ini ingin melihat derajat fleksibilitas seseorang. Judging disini bukan berarti judgemental (atau menghakimi). Judging disini diartikan sebagai tipe orang yang selalu bertumpu pada rencana yang sistematis, serta senantiasa berpikir dan bertindak secara sekuensial (tidak melompat-lompat). Sementara tipe perceiving adalah mereka yang bersikap fleksibel, adaptif, dan bertindak secara random untuk melihat beragam peluang yang muncul.