|  |  |  |
| --- | --- | --- |
| C:\Users\Admin\Pictures\JATA KPM.png  **BAHAGIAN PENDIDIKAN DAN LATIHAN TEKNIK VOKASIONAL**  **KEMENTERIAN PENDIDIKAN MALAYSIA**  **ARAS 5 & 6, BLOK E14, KOMPLEKS E,**  **PUSAT PENTADBIRAN KERAJAAN PERSEKUTUAN**  **KERTAS PENERANGAN**  ***(INFORMATION SHEET)*** | | |
| **KOD DAN NAMA PROGRAM NOSS** | IT-010-3:2016 PEMBANGUNAN APLIKASI | |
| **TAHAP** | 3 | |
| **KOD, NAMA CU DAN WA NOSS** | CU2/WA1 – INTERPRET APPLICATION MODULE DEVELOPMENT REQUIREMENT | |
| **KOD DAN NAMA PROGRAM KV** | TEKNOLOGI SISTEM PENGURUSAN PANGKALAN DATA DAN APLIKASI WEB | |
| **KOD DAN NAMA KURSUS KSKV** | KPD 2023 WEB PROGRAMMING | |
| **NO.DAN TAJUK STANDARD KANDUNGAN KSKV** | K1 INTERPRET APPLICATION MODULE DEVELOPMENT REQUIREMENT | |
| **NO. KOD NOSS** | IT-010-3:2016-C02/ P(2/11) | Muka Surat : 1 Drp : 8 |
| **NO. KOD KSKV** | KPD 2023 / P(2/11) |

**TAJUK/***TITLE* **:**

**INTERPRET APPLICATION MODULE DEVELOPMENT REQUIREMENT**

**TUJUAN/***PURPOSE* **:**

Kertas penerangan ini adalah bertujuan menerangkan mengenai :

* Pengenalan kepada TDD
* Kandungan ringkasan projek.
* Fungsi aplikasi *mock up*

**1.5 PENGENALAN KEPADA TEST DRIVEN DEVELOPMENT (TDD)**

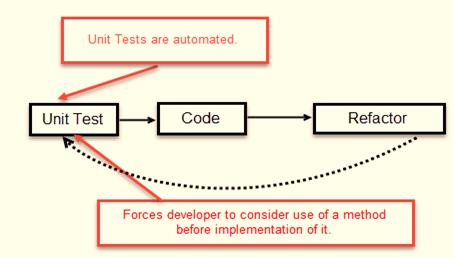
**Definisi TDD**

TDD bermula dengan mereka bentuk dan membangunkan test untuk setiap fungsi kecil aplikasi. Dalam pendekatan TDD, pertama test itu dibangunkan yang menyatakan dan mengesahkan apa kod akan lakukan.

Dalam proses Testing normal, kita menjana kod dan kemudian menguji untuk mengetahui lebih lanjut mengenai pengtest perisian merujuk Perisian Test bantuan. Test mungkin gagal kerana test yang dibangunkan sebelum pembangunan. Dalam usaha untuk lulus test, pasukan pembangunan mempunyai untuk membangunkan dan refactors kod. Pemfaktoran semula kod bermaksud mengubah beberapa kod tanpa menjejaskan tingkah laku.

**Rajah 4**

Konsep mudah TDD adalah untuk menulis dan membetulkan test gagal sebelum menulis kod baru (sebelum pembangunan). Ini membantu untuk mengelakkan pertindihan kod seperti yang kita menulis sedikit kod pada satu masa untuk lulus test. (Test adalah apa-apa tetapi keadaan keperluan yang kita perlu menguji untuk memenuhi mereka)



**Kitaran TDD mentakrifkan**

* Tulis test
* Membuatnya berjalan.
* Menukar kod untuk membetulkannya iaitu Refactor.
* Proses berulang.

**Beberapa penjelasan mengenai TDD:**

* TDD bukanlah tentang "Test" atau kira-kira "Design".
* TDD tidak bermakna "menulis beberapa test, kemudian membina satu sistem yang lulus test.
* TDD tidak bermakna "melakukan banyak Testing."

TDD ialah proses membangunkan dan menjalankan test automatik sebelum

pembangunan sebenar permohonan. Oleh itu, TDD kadang-kadang juga dipanggil

sebagai Test Kemajuan Pertama.

**Fungsi TDD**

* Menghasilkan aplikasi berkualiti tinggi dalam masa yang kurang daripada yang mungkin dengan kaedah yang lebih tua.
* Pelaksanaan yang betul TDD memerlukan pemaju dan penguji untuk menjangka dengan tepat bagaimana aplikasi dan ciri-cirinya akan digunakan dalam dunia sebenar.
* TDD mencipta suite regresi test sebagai kesan sampingan yang boleh meminimumkan test manual manusia, sementara mencari masalah sebelum ini, yang membawa kepada perbaikan lebih cepat.
  + Pemberitahuan ralat.

Pemaju menguji kod mereka tetapi dalam dunia pangkalan data, ini selalunya terdiri daripada test manual atau skrip sekali sahaja. Menggunakan TDD anda membina, sepanjang masa, satu set test automatik yang anda dan mana-mana pemaju lain boleh menjalankan semula sesuka hati.

* + Direka dengan lebih baik, bersih dan kod yang lebih boleh dilanjut.

Ia membantu untuk memahami bagaimana kod akan digunakan dan bagaimana ia berinteraksi dengan modul lain. Ia menyebabkan keputusan reka bentuk yang lebih baik dan kod yang lebih dikekalkan. TDD membolehkan menulis kod yang lebih kecil yang mempunyai tanggungjawab tunggal dan bukannya prosedur monolitik dengan pelbagai tanggungjawab. Ini menjadikan kod lebih mudah untuk difahami. TDD juga memaksa untuk menulis hanya kod pengeluaran untuk lulus test berdasarkan keperluan pengguna.

* Keyakinan kepada Refactor.

Jika anda refactor kod, ada boleh menjadi kemungkinan rehat dalam kod. Jadi mempunyai satu set ujian automatik anda boleh menetapkan mereka rehat sebelum dibebaskan. amaran yang sewajarnya akan diberikan jika rehat ditemui apabila ujian automatik digunakan. Menggunakan TDD, perlu keputusan dalam lebih cepat, kod yang lebih boleh dilanjut dengan bug lebih sedikit yang boleh dikemaskini dengan risiko yang minimum.

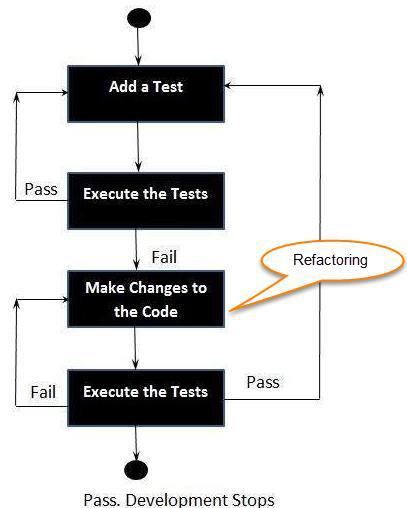
* Baik untuk kerja berpasukan.

Jika tiada mana-mana ahli pasukan, ahli pasukan lain boleh dengan mudah mengambil dan bekerja pada kod. Ia juga membantu perkongsian pengetahuan, dan dengan itu membuat pasukan yang lebih berkesan secara keseluruhan.

* Baik untuk Pemaju.

Walaupun pemaju perlu menghabiskan lebih banyak masa secara bertulis kes pengujian TDD, ia mengambil masa lebih kurang untuk debugging dan membangunkan ciri-ciri baru. Anda akan menulis lebih bersih, kurang rumit kod.

**Cara Melaksana**

****

**Rajah 5**

Berikut langkah-langkah yang menentukan bagaimana untuk melakukan test TDD,

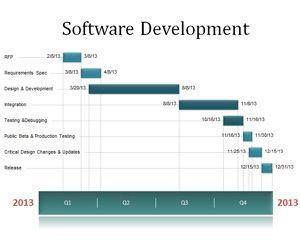
* Menambah test.
* Menjalankan semua test dan melihat jika apa-apa test baru gagal.
* Menulis beberapa kod.
* Test dikendali dan kod Refactor.
* Mengulangi.

**1.6 KANDUNGAN RINGKASAN PROJEK**

Sebuah Ringkas Projek adalah dokumen penting, yang mentakrifkan keperluan pelanggan utama untuk projek itu. Biasanya, mana-mana projek bermula dengan dokumen ini. Eksekutor menghantar Gambaran kepada pelanggan, kepada siapakah mengkonkretkan projek. Berdasarkan data yang disediakan, anda memulakan projek merealisasikan menjelang atau perbincangan lanjut mengenai butiran.

Sebuah Gambaran yang baik harus mengandungi semua maklumat penting mengenai projek dan perniagaan wasi perlu menilai dan meneruskan pembangunan projek pelanggan. Tidak ada template ringkas universal. Setiap jenis projek - reka bentuk, pembangunan web, pembangunan aplikasi mudah alih, PR dan lain-lain - memerlukan brief yang berbeza. Namun, struktur ringkas umum kekal sama dalam kebanyakan kes. Ia adalah unsur-unsur tertentu nic yang berbeza-beza.

Apabila menggariskan maklumat, jumlah data yang diperlukan boleh berbeza dengan ketara juga. Dalam kes-kes tertentu beberapa frasa yang singkat dan umum yang mencukupi (apabila hanya 1-2 muka surat ringkas yang lama), manakala dalam beberapa kes lain semua mata projek utama harus dinyatakan dalam butiran (kemudian ringkas boleh meregang hingga 10- 20 muka surat)

* Garis Masa Pembangunan

**Rajah 6**

* Nombor Modul

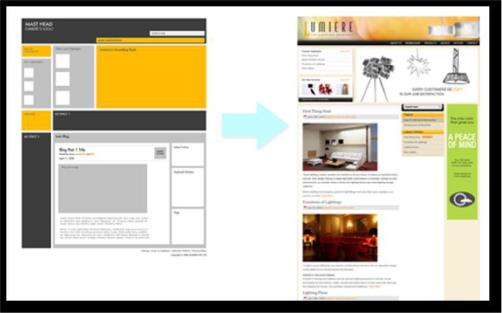
**Rajah 7**

* Assignasi Tugasan

**Rajah 8**

**1.7 Aplikasi Mock Up**

*Mock up* adalah proses merekabentuk peringkat sederhana yang boleh dikatakan sebagairekabentuk pra-keluaran yang di datangkan dengan paparan yang sebenar dan menggambarkan antara muka sebenar aplikasi yang mengandungi warna dan gaya (*style*) namun belum berfungsi. *Mock up* mengandungi struktur maklumat, memaparkan kandungan dan mendemonstrasikan fungsi yang asas dengan cara yang statik kerana ia tidak benar-benar berfungsi contohnya aplikasi tersebut tidak mampu menyimpan sebarang input dari pengguna kerana belum dihubungkan dengan pangkalan data. Antara perisian yang boleh membangunkan *mock up* ialah *Mokingbird, Mockup builder, proto.io* (untuk aplikasi mudah alih) dan banyak lagi.



**Rajah 9**

**Paparan *Wireframe* ke Paparan *Mockup***

Sumber : http://www.onextrapixel.com/

**KELEBIHAN PELAKSANAAN APLIKASI MOCK UP**

1. Mudah memahami keperluan sebenar pelanggan apabila pengguna terlibat secara aktif dalam pembangunan
2. Pembangun dapat membangunkan aplikasi atau menghasilkan produk akhir yang menepati kehendak pengguna
3. Oleh kerana di dalam kaedah ini menyediakan simulasi kepada pengguna, pengguna akan dapat pemahaman yang lebih baik daripada sistem yang dibangunkan.
4. Kesilapan boleh dikesan lebih awal.
5. Maklum balas pengguna yang lebih cepat boleh didapati membawa kepada penyelesaian yang lebih baik.
6. Fungsi yang hilang boleh dikenal pasti dengan mudah
7. Fungsi mengelirukan atau sukar dapat dikenal pasti
8. Dengan simulasi prototaip mungkin akan mencetus beberapa idea tambahan yang tidak dijangka malah lebih diperlukan

**KEKURANGAN PELAKSANAAN APLIKASI MOCK UP**

1. Mengabaikan struktur pembangunan yang lebih terancang dengan terus ke tahap pelaksanaan kod dan terus melakukan penambahbaikan. Analisis masalah tidak dilakukan dengan jelas.
2. Kaedah ini boleh meningkatkan kerumitan sistem di mana skop sistem itu boleh berkembang di luar rancangan asal.
3. Proses mungkin lebih rumit disebabkan banyak melibatkan pengguna yang kadangkala tidak memahami konsep sebenar sesuatu aplikasi.

IV. Pelanggan mungkin mahu terus menggunakan aplikasi prototaip tanpa menunggu aplikasi sebenar dihasilkan.

1. Usaha yang lebih daripada kumpulan pembangun untuk menghasilkan aplikasi tersebut dan terpaksa membuat penambahbaikan dari masa ke semasa.

**SOALAN /** *QUESTION:*

1. Terangkan fungsi “**Pemberitahuan ralat”** bagi *Test Driven Development (TDD).*

**RUJUKAN /** *REFERENCE:*

|  |  |
| --- | --- |
| 1. | http://slideplayer.info/slide/2772293/# |
| 2. | http://www.priorsolution.com/payment-gateway/ |
| 3. | http://www.onextrapixel.com/ |
| 4 | https://www.guru99.com/test-driven-development.html |