**Persiapan**

1. Pada folder ‘rapidtest’ yang telah kita buat di perkuliahan yang lalu, buatlah dua sub-folder baru, masing-masing dengan nama ‘model’ dan ‘view’. Sesuai dengan namanya, folder ‘model’ akan berisi berkas-berkas model, demikian pula pada folder ‘view’ akan berisi berkas-berkas view

Graphical user interface, table

Description automatically generated

1. Pindahkan (cut-paste) berkas ‘Peserta.php’ dan ‘Konektor.php’ ke dalam folder ‘model’

Text

Description automatically generated

1. indahkan (cut-paste) berkas ‘tambah-peserta.php’ ke dalam folder ‘view’. Setelah itu ubah namanya menjadi ‘view-tambah-peserta.php’

Text

Description automatically generated

1. Buka berkas ‘Peserta.php
2. Tambahkan method konstruktor pada kelas Peserta seperti kode di bawah ini:

Graphical user interface, text

Description automatically generated

1. Karena kita menambahkan constructor method pada kelas Peserta, maka pemanggilan kosntruktor di tempat lain juga harus menyesuaikan. Maka dari itu, buka berkas ‘tambah.php’, kemudian ganti isinya menjadi seperti kode di bawah ini (kode merah yang dicoret menandakan kode sebelumnya yang harus dihapus, sedangkan kode yang ditandai warna abu-abu adalah kode yang harus ditambahkan):

Text

Description automatically generated

1. Pada kode di poin nomor 3, ada sesuatu yang tidak elegan, yaitu pada kondisi percabangan (if statement). Kondisi if tersebut kita buat untuk memeriksa apakah ada formulir yang dikirim oleh klien atau tidak. Kita akan perbaiki dengan melakukan abstraksi terhadap kondisi tersebut sehingga kode menjadi lebih deskriptif. Abstraksi yang kita lakukan adalah dengan membuat fungsi baru sebagai berikut:

Text

Description automatically generated

1. Pada perkuliahan sebelumnya, formulir pendaftaran peserta rapid test langsung diakses oleh klien. Sekarang, setelah menggunakan MVC, maka formulir tersebut tidak langsung diakses oleh klien, melainkan harus melalui perantara controller (lihat kembali gambar 1). Formulir akan ditampilkan oleh controller apabila tidak ada permintaan proses tambah (atau dengan kata lain, fungsi adaPermintaanProsesTambah() bernilai false). Mari kita tambahkan kode menjadi seperti di bawah ini:

Text

Description automatically generated

1. Karena tadi kita juga memindahkan berkas “Peserta.php” ke folder “model”, maka kita kita juga perlu mengubah isi dari baris pernyataan require\_once menjadi:



1. Untuk menampilkan daftar peserta rapid test, kita perlu menambahkan business logic baru pada model Peserta. Silahkan buka berkas “Peserta.php”
2. Pada perkuliahan sebelumnya, kita telah mendefinisikan fungsi statis dengan nama getAll(). Sekarang kita lengkapi dengan mengisi *body*-nya seperti kode di bawah ini:

Text

Description automatically generated

1. Buatlah berkas controller baru dengan nama “index.php”
2. Karena kita akan menampilkan daftar peserta, maka kita perlu bantuan *model* Peserta untuk menjalankan *business logic*-nya. Tambahkan kode berikut ini pada *controller* “indeks.php”:

Text

Description automatically generated

1. Pada kode di poin nomor 3, terlihat bahwa fungsi tampilkanHalamanIndex melakukan passing variabel bernama $daftarPeserta yang diperoleh dari model. Variabel $daftarPeserta berisi daftar peserta dalam bentuk larik (array) Peserta. Larik tersebut akan kita proses pada view. Pada folder view, mari kita buat file dengan nama “view-indeks.php”, kemudian isi dengan kode seperti di bawah ini:

Text

Description automatically generated

1. Bila sudah selesai, silahkan saudara akses http://localhost/rapidtest/index.php dengan peramban web. Jika berhasil, maka saudara akan dapati daftar peserta dan tautan untuk menuju formulir penambahan peserta.
2. Buka berkas tambah.php, kemudian tambahkan menjadi seperti kode di bawah ini:

Text

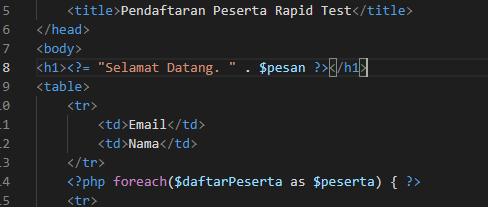
Description automatically generated

1. Pada kode di poin nomor 1, kita ingin mengirim pesan “Peserta berhasil ditambah.” setelah proses penambahan peserta untuk ditampilkan pada halaman utama. Oleh karena itu, kita perlu menyesuaikan controller dan view untuk halaman utama. Pertama-tama kita ubah controller “indeks.php” terlebih dahulu, tambahkan kode berikut ini:

Text

Description automatically generated

1. Berikutnya kita ubah view “view-index.php” dengan menyisipkan kode berikut:



1. Setelah selesai, cobalah menambahkan daftar peserta baru. Setelah itu periksa apakah pesan telah berhasil ditampilkan
2. Buka berkas view-index.php, kemudian sisipkan kode seperti di bawah ini:

Text

Description automatically generated

1. Dapat dilihat bahwa kita menggunakan tautan dengan query string email untuk mengubah data, sedangkan kita menggunakan formulir berisi sebuah hidden field email untuk menghapus data berdasarkan email.
2. Buka berkas “Peserta.php”, kemudian pada kelas Peserta kita akan mengisi *method* save() yang sebelumnya sudah kita buat namun masih kosong. Isilah *method* tersebut dengan kode seperti di bawah ini:

A screenshot of a computer

Description automatically generated with medium confidence

1. Perlu diingat, bahwa saat kita membuat tabel peserta, kolom email adalah PRIMARY KEY, sehingga query UPDATE menjadi seperti di atas. Dari query tersebut, dapat disimpulkan bahwa kita mengizinkan pengubahan data, baik perubahan pada email maupun nama.
2. Karena kita mengizinkan perubahan pada semua kolom, maka secara tampilan formulir nantinya, formulir untuk proses UPDATE (pengubahan data) hampir sama dengan formulir untuk proses INSERT (penambahan data) yang telah kita buat sebelumnya. Perbedaan utamanya adalah pada formulir penambahan data, kolom isian masih dalam kondisi kosong, sedangkan pada formulir pengubahan data, kolom isian sudah berisi data awal yang akan kita ubah. Oleh karena itu, kita perlu mengimplementasikan fungsi untuk mengambil data peserta (tunggal). Pada perkuliahan sebelumnya, kita telah menyiapkan fungsi get($email), kita akan ubah fungsi tersebut menjadi *static* dan kita lengkapi fungsi tersebut seperti kode di bawah ini:

Text

Description automatically generated

1. Sampai di sini, model Peserta.php telah siap melayani proses update data. Sekarang kita beralih ke bagian controller. Buatlah controller baru dengan nama ubah.php, kemudian isi dengan kode seperi di bawah ini:

Text

Description automatically generated

1. Sebagaimana query yang kita definisikan pada model Peserta.php, ada tiga parameter yang harus dipelihara, yaitu email (email asal sebelum diubah, sebagai primary key), email baru dan nama baru. Semuanya telah kita tangani pada controller di kode poin nomor 4 di atas.
2. Selanjutnya kita perlu membuat *view* untuk menampilkan formulir *update*. Silahkan saudara buat berkas dengan nama “view-ubah-peserta.php” kemudian isilah dengan kode seperti di bawah ini:

A screenshot of a computer

Description automatically generated

1. Jika diperhatikan, kode formulir ubah peserta sangat mirip dengan formulir tambah peserta (saudara bisa langsung *copy-paste* kemudian melakukan sedikitpenyesuaian). Pada formulir ubah peserta, kita menggunakan hidden input untuk menyimpan data email asal sebagai primary key pada proses update agar nilainya tidak diubah oleh pengguna, sedangkan untuk email baru dan nama baru menggunakan input teks yang dapat dimanipulasi dengan mudah oleh pengguna.
2. Silahkan saudara coba jalankan di peramban web untuk memastikan semua telah berjalan sesuai harapan.
3. Buka berkas “Peserta.php”, kemudian pada kelas Peserta kita ubah method delete() yang sebelumnya sudah kita definisikan menjadi seperti kode di bawah ini:

Text

Description automatically generated

1. Setelah itu, kita membuat controller baru untuk menghapus data. Buatlah controller baru dengan nama “hapus.php”, kemudian isilah seperti kode di bawah ini:

Text

Description automatically generated

1. Pada kasus hapus ini, kita tidak memerlukan view tersendiri, karena tombol hapus kita ikutkan pada “view-indeks.php”.
2. Silahkan saudara coba jalankan dan buka di peramban web, pastikan semua fungsionalitas telah berjalan sebagaimana mestinya seperti pada gambar 2.

Graphical user interface, text

Description automatically generated