View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Analisis Harga Pokok Produksi Rumah Pada

RANCANG BANGUN *IMAGE BASED* CAPTCHA (*COMPLETELY AUTOMATED PUBLIC TURING TEST TO TELL COMPUTER AND HUMAN APART*) TERINTEGRASI DENGAN *JIGSAW PUZZLE* MENGGUNAKAN HTML5

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Teknik Pada
Jurusan Teknik Informatika

Oleh :

HENDRA AULIA
10751000224

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU
PEKANBARU
2013

RANCANG BANGUN *IMAGE BASED* CAPTCHA (*COMPLETELY AUTOMATED PUBLIC TURING TEST TO TELL COMPUTER AND HUMAN APART*) TERINTEGRASI DENGAN *JIGSAW PUZZLE* MENGGUNAKAN HTML5

HENDRA AULIA
10751000224

Tanggal Sidang : 11 Oktober 2013
Periode Wisuda : 28 November 2013

Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Sultan Syarif Kasim Riau
Jl. Soebrantas No.155 Pekanbaru

ABSTRAK

CAPTCHA (*Completely Automated Public Turing Test to tell Computer and Human Apart*) adalah program komputer yang dapat melakukan tes dimana sebagian manusia dapat lulus, namun *bot* komputer tidak. CAPTCHA pada umumnya digunakan untuk memverifikasi pengunjung situs di internet dengan menggunakan tulisan terdistorsi pada sebuah gambar. CAPTCHA menghadapi sejumlah serangan dan potensi masalah dalam penggunaannya. Sejumlah Solusi dikembangkan untuk menghadapi serangan bot komputer, salah satunya dengan menggunakan skema *mouse intervention*. Dalam tugas akhir ini, pengembangan *Image Based* CAPTCHA diintegrasikan dengan kombinasi *Jigsaw Puzzle* dan dilakukan dengan menggunakan teknologi HTML5. Penelitian ini menggunakan metode pengembangan penelitian RAD (*Rapid Application Development*). CAPTCHA yang telah dikembangkan diuji dengan menggunakan teknik *Blackbox*, pengujian dimensi gambar, *Security*, dan *User Acceptence Test*. Hasil penelitian ini membuktikan bahwa penggunaan *Jigsaw Puzzle* CAPTCHA ini dapat mengurangi potensi masalah yang ada, namun tetap memudahkan manusia dalam menggunakannya.

Kata Kunci: *Bot*, HTML5, *Image Based* CAPTCHA, *Mouse intervention*

LEMBAR PENGESAHAN

RANCANG BANGUN *IMAGE BASED* CAPTCHA (*COMPLETELY AUTOMATED PUBLIC TURING TEST TO TELL COMPUTER AND HUMAN APART*) TERINTEGRASI DENGAN *JIGSAW PUZZLE* MENGGUNAKAN HTML5

TUGAS AKHIR

oleh:

HENDRA AULIA
10751000224

Telah dipertahankan di depan sidang dewan penguji

Sebagai salah satu syarat untuk memperoleh gelar sarjana Teknik Informatika

Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau

Di Pekanbaru, pada tanggal11 Oktober 2013

Pekanbaru. 11 Oktober 2013

Mengesahkan,

Dekan

Dra. Hj. Yenita Morena, M.Si
NIP. 19601125 198503 2 002

Ketua Jurusan

Elin Haerani, ST, M.Kom
NIP. 19810523 200710 2 003

DEWAN PENGUJI

Ketua	: Benny Sukma Negara, S.T, M.T
Sekretaris	: Benny Sukma Negara, S.T, M.T
Penguji I	: Reski Mai Candra, S.T, M.Sc
Penguji II	: Muhammad Affandes, M.T

KATA PENGANTAR

Assalammu'alaikum wa rahmatullahi wa barakatuh.

Alhamdulillahi Rabbil Alamin, segala puji syukur kehadirat Allah SWT yang senantiasa melimpahkan rahmat dan karunia-Nya, sehingga penulis mampu menyelesaikan Tugas Akhir ini dengan baik. Shalawat serta salam terucap buat junjungan kita Rasulullah Muhammad SAW karena jasa Beliau yang telah membawa manusia dari zaman kebodohan ke zaman yang penuh dengan ilmu pengetahuan seperti sekarang ini.

Tugas Akhir ini disusun sebagai salah satu syarat untuk mendapatkan kelulusan pada jurusan Teknik Informatika Universitas Islam Negeri Sultan Syarif Kasim Riau. Banyak sekali pihak yang telah membantu penulis dalam penyusunan Tugas Akhir ini, baik berupa bantuan materi ataupun berupa motivasi dan dukungan kepada penulis. Semua itu tentu terlalu banyak bagi penulis untuk membalasnya, namun pada kesempatan ini penulis hanya dapat mengucapkan terima kasih kepada :

1. Bapak Prof. Dr. H. M. Nazir, selaku Rektor Universitas Islam Negeri Sultan Syarif Kasim Riau.
2. Ibu Dra. Hj. Yenita Morena, M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau.
3. Ibu Elin Haerani, S.T, M.Kom, selaku Ketua Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau.
4. Bapak Benny Sukma Negara, M.T selaku Pembimbing, yang telah memberi bimbingan, arahan, saran dan motivasi yang sangat berharga dalam menyusun Tugas Akhir ini.
5. Bapak Reski Mai Candra, ST, M.Sc selaku penguji I. Terimakasih untuk ilmu, saran dan perbaikannya untuk menyempurnakan Tugas akhir ini.

6. Bapak M. Affandes, S.T, M.T selaku penguji II sekaligus sebagai koordinator Tugas Akhir yang telah banyak membantu dalam menyusun jadwal dan koordinasi dengan para pembimbing dan sesuatu hal yang memperlancar jalannya Tugas Akhir ini.
7. Seluruh dosen Jurusan Teknik Informatika UIN Suska Riau yang telah memberikan ilmu dan pengetahuan yang bermanfaat kepada penulis selama mengikuti perkuliahan di Jurusan Teknik Informatika.
8. Kepada Ayahanda H. Arlen dan Ibunda Hj. Herlinda tercinta yang selalu memberikan doa, motivasi, bimbingan yang tiada hentinya, serta telah banyak berkorban demi keberhasilan anaknya dan merupakan motivasi saya untuk memberikan yang terbaik.
9. Kepada adinda Felycia Belri budiyani, S.Kel yang selalu memberi dukungan, doa, dan semangatnya hingga saya dapat menyelesaikan Tugas Akhir ini.
10. Teman-teman seperjuangan di kampus (Ligar Sekar Wangi, Frima Hayati, M. Riendra Putra, Ardian Saputra, Inggih Permana, Erias Fantoni dan khususnya TIF angkatan 2007).
11. Seluruh pihak yang belum penulis cantumkan, terima kasih.

Penulis menyadari bahwa dalam penulisan Tugas Akhir ini masih banyak kesalahan dan kekurangan, oleh karena itu kritik dan saran yang sifatnya membangun sangat penulis harapkan untuk kesempurnaan Tugas Akhir ini. Akhirnya penulis berharap semoga Tugas Akhir ini dapat memberikan sesuatu yang bermanfaat bagi siapa saja yang membacanya. Amin.

Wassalamu'alaikum wa rahmatullahi wa barakatuh

Pekanbaru, Oktober 2013

Penulis

DAFTAR ISI

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Salah satu teknik untuk mengamankan *website* terhadap serangan kejahatan komputer yaitu dengan menggunakan CAPTCHA (*Completely Automated Public Turing test to tell Computers and Humans Apart*). CAPTCHA adalah sebuah tes yang dibangkitkan secara otomatis untuk membedakan manusia dengan program komputer, dengan cara memberikan pengujian yang dapat diselesaikan dengan mudah oleh manusia, tetapi sulit untuk program komputer (Tanvee dkk, 2011). CAPTCHA dikembangkan pertama kali pada tahun 2000 oleh Luis von Ahn, Manuel Blum, Nicholas Hopper dan John Langford dari Universitas Carnegie Mellon untuk *website* Yahoo!. CAPTCHA pada umumnya berupa gambar berukuran kecil bertuliskan kode tertentu, dimana pengguna diminta untuk mengetikkan pada kotak isian dan biasanya digunakan untuk men-*submit* sesuatu. Salah satu contohnya dapat ditemukan pada formulir pendaftaran *email* gratis di Yahoo!. Prosedur pengisian ini digunakan untuk mencegah *submit* otomatis oleh *bot*, serta membuat kerja *bot* menjadi lebih sulit menerka kode tersebut. Dengan kata lain, CAPTCHA adalah program komputer yang dapat melakukan tes dimana sebagian besar manusia dapat lulus, namun program komputer tidak.

Namun dalam perkembangannya, CAPTCHA sudah banyak dipecahkan oleh *bot* komputer. Salah satu CAPTCHA yang telah dipecahkan adalah gimpy. Greg Moly dan Jitendra Malik di dalam jurnalnya yang berjudul *Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA* telah berhasil memecahkan CAPTCHA gimpy dengan menggunakan teknik OCR *attack* (*Optical Character Recognition*) dan *Dictionary attack*. OCR *attack* yang dimaksudkan di sini adalah penggunaan aplikasi OCR untuk mengambil dan mengidentifikasi karakter-karakter yang terdapat pada CAPTCHA, sedangkan *dictionary* dan *database attack* adalah sebuah metode serangan yang digunakan berdasarkan pada kamus dan *database* yang menjadi sumber data CAPTCHA.

Sejumlah solusi telah dikembangkan agar sebuah situs dapat mencegah eksplorasi tersebut. Triet Giang dan Fei Liu mengajukan skema CAPTCHA baru untuk mengurangi dampak serangan terhadap CAPTCHA, yakni skema *mouse intervention*. *Mouse intervention* adalah mengambil alih fungsi *keyboard* sebagai alat untuk meng*input* CAPTCHA kepada pergerakan *mouse*. Triet Giang menjelaskan bahwa penggunaan skema *mouse intervention* memiliki kelebihan dibandingkan skema CATCHA yang lain karena tidak rentan terhadap serangan yang biasa menyerang CAPTCHA berbasis *text* pada umumnya, seperti *OCR attack, dictionary attack*, dan *database attack* serta aman terhadap *simple brute force attack.*

Jigsaw Puzzle CAPTCHA adalah salah satu contoh penggunaan skema *mouse intervention*. Dalam jurnal yang berjudul "*JigCAPTCHA: An Advance Image-Base CAPTCHA Integrated with Jigsaw Piece Puzzle Using AJAX* " Nitisha Payal, Nindhi Chaudhary, dan Parma Nand Astya menyatakan bahwa metode *Drag and Drop Jigsaw Puzzle* lebih menarik daripada CAPTCHA berbasis *text*. Karena *Jigsaw Puzzle* CAPTCHA terlihat lebih seperti sebuah permainan (*game*) dibandingkan sebuah tes logika. Dengan demikian, pengguna tidak merasakan sedang melaksanakan ujian intelektual.

Pengembangkan CAPTCHA dengan skema *mouse intervention* di dalam penelitian ini menggunakan teknologi HTML5. HTML5 memiliki *feature* terbaru yaitu *Event Drag and Drop* yang dapat dimanfaatkan untuk mengembangkan metode *Drag and Drop Jigsaw Puzzle*. Selain itu, HTML5 dipilih karena tidak membutuhkan tambahan *plugin* dalam penggunaannya. Ilustrasi dari CAPTCHA yang akan dikembangkan adalah dengan menampilkan potongan gambar utuh dari sebuah objek yang kemudian diacak dengan membentuk potongan *puzzle*. Kemudian *user* atau pengguna CAPTCHA dituntut untuk menyusun kembali potongan gambar tersebut dengan cara menggeser (*Drag*) dan menempatkan (*Drop*) potongan gambar pada posisi yang seharusnya sehingga menghasilkan suatu gambar yang utuh. CAPTCHA yang telah dikembangkan akan diuji dengan diintegrasikan ke sebuah *form* sederhana. Jika CAPTCHA berhasil diselesaikan, maka akan tampil sebuah notifikasi bahwa data

dapat di *submit*. Sedangkan jika tidak berhasil menyelesaikan CAPTCHA, maka data tidak bisa di*input*.

Penelitian ini diharapkan akan menghasilkan sebuah alternatif CAPTCHA yang memiliki sisi sekuritas yang kuat sehingga tahan terhadap berbagai serangan namun tetap *user-friendly* di sisi penggunaannya.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang diuraikan di atas, maka dapat diambil rumusan permasalahan adalah bagaimana mengembangkan aplikasi CAPTCHA yang dapat mencegah dan mengurangi serangan oleh *bot*, namun tetap nyaman dari sisi penggunaannya.

1.3 Batasan Masalah

Cakupan masalah yang dibahas dibatasi dalam hal-hal berikut :

1. CAPTCHA akan menggunakan gambar sebuah objek, objek yang dipilih akan disesuaikan bentuk dan resolusinya sehingga mudah dikenali oleh manusia dan memudahkan dalam proses pembuatan potongan *puzzle*.
2. CAPTCHA dikembangkan hanya untuk lingkungan perangkat komputer dengan operasi sistem *windows*.
3. CAPTCHA diuji dengan 4 pengujian yaitu: *blackbox*, pengujian dimensi gambar, *Security* dan *User Acceptence Test* (UAT).
4. CAPTCHA akan diintegrasikan pada aplikasi *form* sederhana. *Form* tersebut hanya sebagai tempat pengujian fungsional CAPTCHA, tidak berpengaruh pada pengembangan CAPTCHA.

1.4 Tujuan

Tujuan dari penelitian ini adalah dihasilkannya suatu alternatif CAPTCHA yang dapat menghilangkan atau mengurangi kelemahan dan potensi masalah dari CAPTCHA yang telah ada saat ini.

1.5 Sistematika Penulisan

Sistematika penulisan Tugas Akhir ini diatur sedemikian rupa sehingga segala kebutuhan yang dipergunakan di dalam pengembangan perangkat lunak tersebut dapat dipahami dengan mudah.

Adapun sistematika penulisan laporan Tugas Akhir ini adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini menjelaskan latar belakang, rumusan masalah, tujuan, metode penelitian dan sistematika penulisan laporan.

BAB II TINJAUAN PUSTAKA

Bab ini memuat landasan teori yang berkaitan dengan Tugas Akhir ini.

BAB III METODOLOGI PENELITIAN

Bab ini menjelaskan metodologi yang digunakan dalam penelitian serta analisa kebutuhan dasar pengembangan CAPTCHA.

BAB IV ANALISA DAN PERANCANGAN

Pada bab ini membahas analisa langkah kerja dan perancangan CAPTCHA yang akan dikembangkan sesuai dengan metodologi yang digunakan.

BAB V IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dibahas implementasi dan pengujian hasil penelitian sesuai yang tertera pada rumusan masalah dan pembahasan.

BAB VI PENUTUP

Penutup merupakan jawaban atas rumusan masalah dalam penelitian dan juga intisari dari penelitian serta saran untuk penelitian selanjutnya.

BAB II
TINJAUAN PUSTAKA

2.1 Keamanan komputer

Istilah keamanan komputer (*computer security*) memiliki interpretasi berbeda-beda tergantung pada zaman digunakannya istilah tersebut. Awalnya, keamanan komputer berkaitan dengan menjaga ruangan tempat komputer tersimpan agar aman dari pengrusakan, termasuk menyediakan fasilitas pendingin ruangan dan listrik. Selanjutnya, keamanan berkembang ke arah keamanan data dan proteksi validasi. Seiring dengan perkembangan komputer *desktop* dan penggunaannya di rumah-rumah, keamanan komputer mengarah kepada bentuk perlindungan terhadap pencurian data dan penyerangan melalui jaringan. Keamanan komputer modern memasukkan upaya menjamin kontinuitas proses bisnis (*business continuity*), sedangkan keamanan industri berkaitan dengan hilangnya kontrol akibat dari pencurian, pengrusakan, dan mata-mata, termasuk kontrol personil dan pengamanan secara fisik untuk melindungi perusahaan secara keseluruhan (Rick, 2006).

Keamanan komputer dan jaringan dibangun dengan tiga konsep dasar disingkat dengan C-I-A (Rick, 2006) yaitu:

1. *Confidentiality*, data dikatakan rahasia (*confidential*) jika data tersebut tersembunyi bagi setiap orang kecuali terhadap orang yang memiliki hak akses untuk menggunakannya.
2. *Integrity*. Data dikatakan memiliki integritas (*integrity*) jika tetap identik dengan kondisi terakhir pengguna yang memiliki hak akses menggunakannya.
3. *Availability*. Data dikatakan tersedia (*available*) jika dapat diaksess oleh pengguna yang memiliki hak akses dalam format dan waktu yang ditentukan.

Berkaitan dengan keamanan komputer (Michael, 2005), ada beberapa hal yang perlu diperhatikan yaitu:

1. *Secrets* (kerahasiaan). Sejumlah informasi yang tersimpan di dalam komputer bersifat rahasia, misalnya kata kunci yang digunakan untuk akses ke sistem informasi, data identitas pengguna bahkan rahasia yang jika tersebar dapat berdampak ke kehidupan nyata. Dalam hal ini, keamanan yang dimaksud adalah menjaga agar rahasia tersebut tidak jatuh ke tangan yang salah.
2. *Scarce resource* (keterbatasan sumber daya). Setiap komputer memiliki keterbatasan pemrosesan oleh CPU per-detik, keterbatasan jumlah memori, keterbatasan kapasitas *disk*, dan terbatasnya *bandwidth* komunikasi. Dalam hal ini keamanan yang dimaksud adalah mencegah penyalahgunaan sumber daya tersebut, baik secara sengaja ataupun tidak sengaja.
3. *Good netizenship*. *netizen* adalah seseorang yang secara aktif telibat dalam komunikasi *online* dan menggunakan *internet* untuk melakukan aktifitas sebagai bagian dari kelompok sosial di *internet*, misalnya menyampaikan dan menerima pendapat, menyebarkan informasi, menjadikan diri sebagai bagian dari sumber daya intelektual dan sosial komunitas. Secara umum, *netizen* adalah pengguna intenet. Istilah *netizen* diperkenalkan oleh Michael Hauben. Jika sebuah komputer sudah terhubung ke *internet*, maka serangan terhadap rahasia dan sumber daya lokal pada komputer tersebut dapat berdampak terhadap komputer lain di dunia. Dalam dunia yang saling terkoneksi ini, setiap *programmer* dan *sysadmin* memiliki tanggung jawab terhadap *programmer* dan *sysadmin* lainnya untuk menjamin kode dan sistem bebas dari eksploitasi, baik secara sengaja ataupun tidak sengaja, yang dapat mempengaruhi komputer lain di jaringan. Oleh karena itu, reputasi sebagai *netizen* yang baik bergantung kepada keamanan dari sistemnya.

2.2 CAPTCHA

Di lingkungan *internet* yang pada umumnya bersifat publik, *anonymous*, selalu aktif dan tidak termonitor, situs *web* berpotensi untuk diserang. Ancaman keamanan terhadap situs-situs ini tidak hanya melibatkan pengguna yang tidak memiliki hak akses, tetapi lebih berasosiasi dengan *automated* atau *mechanical pseudo-users* yang disebut juga *robot*. Dimana komputer menyamar sebagai manusia dengan tujuan untuk berinteraksi dengan situs *web*. Dengan kata lain situs *web* yang dirancang untuk diakses oleh manusia, memang hanya dapat diakses oleh manusia. Hal ini dapat dilakukan dengan membuat tempat pemeriksaan atau *gateway* pada aplikasi, agar hanya manusia yang mampu melewatinya. Dengan suksesnya melakukan hal ini maka akan menjamin serangan otomatis ke situs menjadi tidak mungkin.

CAPTCHA atau singkatan dari *Completely Automated Public Turing test to tell Computers and Humans Apart* yang dibuat pertama kali pada tahun 2000 oleh Louis von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford, dikemukakan oleh Ahn, L., et al (2004:58) bahwa CAPTCHA *"it is a test, any test, that can be automatically generated, which most humans can pass, but that current computer programs cannot pass"*. CAPTCHA pada umumnya diilustrasikan sebagai sebuah proses dimana sistem membangkitkan sebuah gambar dari sekumpulan karakter yang dirusak kemudian ditampilkan dilayar dan *user* diminta untuk menebak gambar tersebut dan mengisikannya pada *textbox* yang disediakan. Jika jawabannya tidak tepat maka *user* tidak bisa malanjutkan ke proses selanjutnya.

2.2.1 Karakteristik CAPTCHA

CAPTCHA adalah sebuah proses yang secara otomatis membangkitkan sebuah tes dengan karakteristik sebagai berikut (Setiawan, 2012):

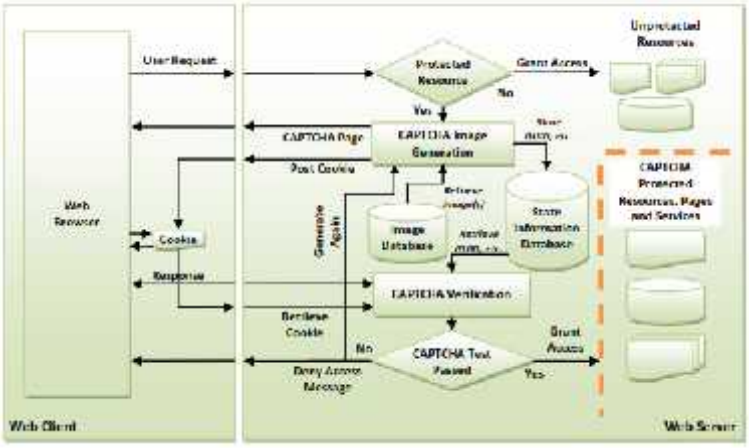
1. *Automated*, tantangan yang dilakukan harus dihasilkan secara otomatis dan dapat ditingkatkan *level* kesulitannya dengan mudah oleh komputer.

2. *Open*, *database* dan algoritma dari tantangan yang dilakukan harus bersifat publik.
3. *Usable*, tantangan harus mudah untuk diselesaikan oleh manusia dalam waktu yang wajar.
4. *Secure*, tantangan yang dilakukan harus sulit bagi komputer untuk memecahkan algoritmanya.

Meskipun CAPTCHA pada dasarnya adalah suatu tes untuk membedakan manusia dari komputer namun CAPTCHA tidaklah cukup hanya dengan memasang sebuah *checkbox* yang berisi pernyataan bahwa apakah pengujung manusia atau bukan karena bagaimanapun cukup mudah untuk membuat *script* untuk memanipulasi jawaban dari tes tersebut.

2.2.2 Sistem Kerja CAPTCHA

Sebuah *web server* memegang sumber daya publik dan terlindungi yang terdapat pada halaman *web*. Data disimpan didalam sebuah *database* atau *files* atau layanan lainnya yang akan dipergunakan oleh manusia sebagai klien. Permintaan pengguna terhadap sumber daya dikirim melalui komputer klien ke *server*, yang mana sumber daya tersebut tidak memiliki perlindungan. Jika sumber daya dilindungi oleh CAPTCHA, akses akan diberikan jika telah melewati atau menyelesaikan uji CAPTCHA. Penjelasan terhadap cara kerja CAPTCHA dijelaskan pada gambar 2.1.



Gambar 2.1 Sistem Kerja CAPTCHA (Banday, 2009)

Server menggunakan beberapa algoritma pengolahan gambar CAPTCHA untuk mengolah sebuah gambar CAPTCHA. Berbeda teknik CAPTCHA, maka berbeda pula algoritma yang digunakan dan pengolahan gambar yang mungkin disimpan di *database* gambar. Informasi pernyataan (*the state information*) terdapat pada *Global Unique Identifier* (GUID) pada pihak klien, dan solusi CAPTCHA disimpan di *The State Information Database* (SID) pada *server*. Penyimpanan GUID klien memastikan bahwa hanya klien yang menerima CAPTCHA yang mendapatkan solusi yang sesuai. Selain menyimpan solusi CAPTCHA dan informasi pernyataan (*the state information*), SID pada *server* juga memungkinkan untuk menyimpan bentuk hash atau enkripsi pada sisi klien. Sebuah halaman *web* yang berisi gambar CAPTCHA dihasilkan dan *cookies* dibentuk di *browser* pada sisi pengguna. Operator manusia memberi respon kepada uji CAPTCHA dan respon dikirim dari klien ke *server*. *Server* memverifikasi solusi CAPTCHA sesuai dengan yang disimpan pada GUID dan GUID klien mengirimkan solusi. Solusi disediakan oleh klien kemudian dibandingkan dengan solusi pada SID atau *cookies* dan ditentukan akses diberikan atau tidak. Jika akses ditolak, sebuah pesan akan ditampilkan untuk mengulang proses uji CAPTCHA kembali.

Pada implementasinya, ada beberapa CAPTCHA yang dapat memblok sementara jika terjadi kesalahan yang berulang. Jika CAPTCHA telah disahkan oleh klien, akses terhadap sumber daya akan diberikan tanpa memberikan uji pemeriksaan lebih lanjut.

2.2.3 Serangan Terhadap CAPTCHA

Serangan terhadap CAPTCHA di antaranya adalah sebagai berikut (Michel, 2005):

1. *Brute force*, dengan cara sederhana menebak dan menelusuri semua kemungkinan berdasarkan *entry* yang ada di dalam kamus. Serangan ini efektif untuk persoalan yang melibatkan penggunaan kata aktual.
2. *Artificial intelligence techniques*, untuk menganalisa atau mempersempit kemungkinan jawaban sampai ke kondisi dimana *brute force* berkemungkinan berhasil. Fungsi pengenalan objek (*object recognition*) dapat digunakan untuk mengenai huruf dan angka yang telah didistorsi.
3. *Hijacking attacks*, sangat efektif karena serangan ini mengeliminasi kebutuhan penyerang untuk memproses CAPTCHA. dihadapkan dengan kebutuhan untuk menjawab persoalan CAPTCHA, maka *hijacker* mengatur situasi dimana ia dapat menantang pengguna lain dalam *setting* berbeda. Misalnya seorang *spammer* ingin mendaftar secara gratis *email account*, mungkin akan membuat situs gratis dan mengiklankan dengan menggnakan *engine spam*-nya. Apabila pengunjung membuka situs tersebut, maka *script* registrasi yang dibuat *spammer* akan menginisialisasi registrasi *email*, dan menampilkan CAPTCHA yang dimiliki *email* sebagai bagian dari syarat untuk mengakses situs tersebut. Jika pengunjung menjawab dengan benar maka jawaban tersebut dikirimkan ke situs penyedia *email* untuk memperoleh akses.

2.2.4 Potensi Masalah CAPTCHA

Potensi masalah dengan menggunakan CAPTCHA diantaranya adalah (Michael, 2005) :

1. *Hijacking CAPTCHA is relatively easy*. Sebuah situs dibuat dengan tujuan melakukan *proxy* terhadap situs lainnya yang menggunakan CAPTCHA. Jika situs ini bisa mendapatkan 50 ribu orang untuk mengunjunginya dan orang-orang tersebut dapat menyediakan jawaban untuk setiap persoalan CAPTCHA, maka situs ini dapat membuktikan bahwa *script* situs ini

adalah manusia sebanyak 50 ribu kali. Untuk itu diperukan upaya untuk mencegah seseorang untuk menjalankan *script* dan mengakses persoalan CAPTCHA di situs yang asli.

2. *The more CAPTCHAs are used, the better AI attack scripts get at reading them*, Jika sebuah grup peneliti mengembangkan CAPTCHA yang lebih sulit, maka grup lainnya mencoba menemukan cara untuk mengalahkannya dan sering berhasil.
3. *Machine effort*, untuk menampilkan CAPTCHA diperlukan waktu dan memori komputer. Bahkan CAPTCHA paling sederhana memerlukan proses *server* untuk digunakan, setidaknya dalam pembuatan gambar dan akses ke *database*.
4. *Too complex*, CAPTCHA yang telalu kompeks mungkin tidak dapat dibaca oleh manusia.
5. *User difficulties*, Kemampuan pengguna untuk menerjemahkan. Bahkan CAPTCHA yang biasa mungkin gagal dikenali oleh manusia.

2.3 Klasifikasi CAPTCHA

Pada saat ini CAPTCHA secara garis besar diklasifikasikan sebagai berikut (Soni, 2010):

1. CAPTCHA menggunakan Gambar
2. CAPTCHA menggunakan Suara

2.3.1 CAPTCHA Menggunakan Gambar

Metode CAPTCHA yang menggunakan Gambar dapat dibagi menjadi 3 jenis yaitu *Optical Character Recognition* (OCR), *Visual Pattern Recognition Bongo,* dan Kombinasi dari gambar dan karakter alphanumeric. Dalam metode CAPTCHA berbasis OCR, pengguna diberikan gambar sebuah kata yang telah diberikan distorsi dan efek-efek pengganggu. Karena adanya efek pada gambar, maka komputer akan mengalami masalah dalam memecahkan susunan karakter tersebut, sedangkan manusia dapat mengenalinya. Tetapi metode ini biasanya menghasilkan ketidakpuasan pengguna karena tingkat kesulitan pembacaan

karakter yang ditampilkan. Di sisi lain, CAPTCHA berbasis OCR telah banyak dibuat program pemecahnya (OCR-*Breaker*) Contoh CAPTCHA yang menggunakan metode ini adalah *Gimpy, EZ-Gimpy, PessimalPrint, BaffleText* dan *ScatterType*.

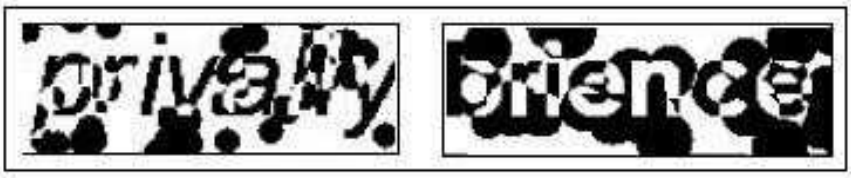
EZ-Gimpy, CAPTCHA jenis ini dikembangkan oleh *The School of Computer Science di Carnegie-Mellon University* dan digunakan oleh yahoo, tujuannya adalah untuk melindungi berbagai layanan yang diberikan oleh yahoo termasuk layanan email gratis. Teknik yang digunakan adalah dengan mengambil kata dari sebuah kamus yang berisi 850 kata dalam bahasa inggris kemudian dibuat gambar dan dirusak yang selanjutnya ditampilkan kepada *user*, berikut ini adalah gambar dari CAPTCHA EZ-Gimpy yang digunakan oleh Yahoo:

Gambar 2.2 CAPTCHA *EZ-Gimpy* yang digunakan Yahoo!

PessimalPrint, Sama halnya dengan *EZ-Gimpy*, CAPTCHA jenis ini pun menggunakan kamus. Kamus tersebut hanya mengandung 70 kata dengan setiap kata terdiri dari lima sampai delapan huruf. Gambar di bawah ini menunjukkan CAPTCHA *PesimmalPrint* yang kemungkinan menggambarkan sebuah string "*reason*"atau "rcason":

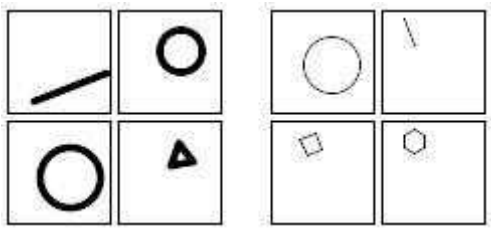
Gambar 2.3 CAPTCHA PesimmalPrint

BaffleText, CAPTCHA jenis ini memiliki perbedaan dengan dua jenis CAPTCHA sebelumnya, dimana kata yang ditampilkan bukanlah kata yang memiliki arti. Di bawah ini adalah salah satu contoh dari CAPTCHA *BaffleText*



Gambar 2.4 CAPTCHA BaffleText

Visual Pattern Recognition Bongo, CAPTCHA jenis ini terdiri dari dua kumpulan gambar. Setiap gambar di salah satu sisi memiliki gambar yang mirip di sisi lainnya. *User* diminta untuk memasangkan gambar dari kumpulan yang satu dengan kumpulan yang lainya. Berikut ini adalah gambar dari *Visual Pattern Recognition Bongo* di mana gambar terdiri dari dua bagian, yakni bagian kiri dan kanan:



Gambar 2.5 Visual *Pattern Recognition* Bongo

Kombinasi dari gambar dan karakter alphanumerik, Sistem menampilkan gambar-gambar yang sering ditemui dalam kehidupan sehari-hari, kemudian *user* diminta untuk mengisikan karakter yang secara umum telah melekat pada gambar yang dimaksud. CAPTCHA jenis ini lebih sulit karena *user* diharuskan berpikir untuk menemukan karakteristik yang dimaksud, ini menyebabkan kemungkinan solusi (jawaban) tidak hanya satu tetapi tergantung dari pengguna.

2.3.2 CAPTCHA Berbasis Suara

Sistem mengeluarkan suara yang dibangkitkan dari sekumpulan karakter secara otomatis, proses tersebut dilakukan oleh *Text-to-Speech Syntheziser* (TTS) kemudian ditambahkan dengan suara-suara gemuruh untuk sedikit mempersulitnya. Batasan dari bahasa dan dialek merupakan sebuah kesulitan dalam mengembangkan CAPTCHA jenis ini.

Berikut ini adalah tabel yang menunjukkan apakah suatu jenis CAPTCHA rentan terhadap suatu serangan atau tidak:

Tabel 2.1 Perbandingan Jenis CAPTCHA dari Kemungkinan Serangan

Jenis CAPTCHA	**Kemungkinan Serangan**			
	OCR	***Dictionary***	***Simple Brute Force***	***Database***
Bongo	No	No	Yes	Yes
Gimpy	Yes	Yes	No	No
BaffleText	Yes	No	No	Yes
Pix	No	No	No	Yes
PesimmalPrint	Yes	Yes	No	No
Sound	No	No	No	Yes

Pada tabel 2.1 di atas, sebagai contoh terlihat bahwa jenis CAPTCHA Bongo rentan terhadap serangan *simple brute force attack* dan *database attack* tetapi di sisi lain CAPTCHA jenis ini tidak rentan terhadap *OCR attack* dan *dictionary attack.*

Tabel 2.2 Perbandingan Kebutuhan Perangkat Lunak CAPTCHA

Jenis CAPTCHA	**Kebutuhan Perangkat Lunak**	
	Database	***Image Processing***
Bongo	Yes	No
Gimpy	No	Yes
BaffleText	Yes	Yes
Pix	Yes	No
PesimmalPrint	No	Yes
Sound	Yes	No

Tabel 2.2 menunjukkan kebutuhan perangkat lunak dari masing-masing jenis CAPTCHA, sebagai contoh adalah CAPTCHA jenis sound di mana CAPTCHA jenis ini membutuhkan perangkat lunak *database* tetapi tidak membutuhkan perangkat lunak untuk *image processing*. Sedangkan tabel 2.3 di bawah ini menunjukkan apakah suatu jenis CAPTCHA mudah dibaca oleh manusia (*user*) atau tidak.

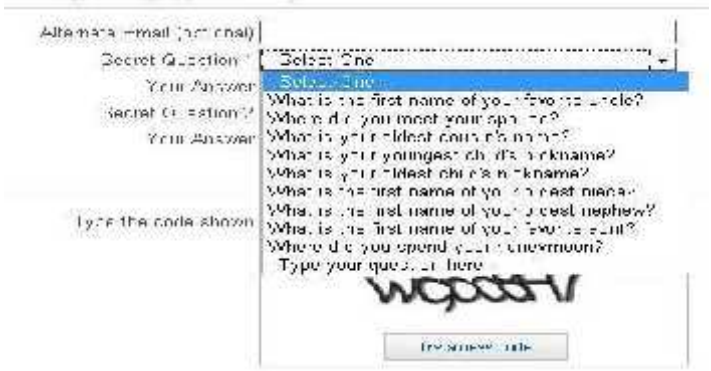
Tabel 2.3 Perbandingan Kemudahan Manusia Membaca CAPTCHA

Jenis CAPTCHA	Mudah Dibaca Manusia
Bongo	Yes
Gimpy	No
BaffleText	No
Pix	Yes
PesimmalPrint	No
Sound	Yes

Secara umum permasalahan yang akan ditampilkan oleh CAPTCHA haruslah dipilih secara acak dan dibangkitkan secara otomatis. Pada beberapa jenis CAPTCHA tingkat kesulitannya akan bertambah seiring dengan pertambahan jumlah sumber CAPTCHA dalam basis data.

2.4 *Mouse Intervention*

Selain jenis-jenis CAPTCHA di atas, terdapat juga skema-skema pengembangan CAPTCHA yang lain seperti *advanched visual scheme*, *expert system*, dan *mouse intervention scheme*. *Advanced visual scheme* menggunakan gambar objek lain selain karakter, CAPTCHA yang termasuk ke dalam skema ini diantaranya adalah simplified 3D to 2D dan *image and characters*. Berbeda dengan yang lainnya, skema *expert system* menggunakan variabel linguistik, contohnya *user* diberikan pertanyaan 'Jika anda seorang laki-laki, maka bagaimana anak anda memanggil anda?'. Bagi komputer hal ini cukup sulit walaupun beberapa aturan logika telah disimpan dalam basis datanya. Gambar 2.1 adalah gambar dari CAPTCHA dengan skema *expert system*, pada gambar tersebut di bagian *dropdown* terdapat sekumpulan kalimat pertanyaan di mana pada bagian *textfield* "*Your Answer*" *user* diharuskan untuk mengisi jawaban.



Gambar 2.6 CAPTCHA dengan Skema *Expert System*

Sedangkan skema *mouse intervention* adalah sebuah skema pengembangan CAPTCHA yang akan digunakan di dalam penelitian ini. Skema ini menarik karena kebanyakan dari CAPTCHA yang telah diuraikan di atas selalu menggunakan *keyboard* sebagai alat untuk menjawab permasalahan, berbeda dengan skema *mouse intervention* yang menggunakan *mouse,* selain itu menurut Giang, T. & Liu, F skema *mouse intervention* merupakan skema yang aman terhadap *OCR attack, dictionary attack, database attack*, dan *simple brute force attack.* Di mana semua proses yang dilakukan untuk menyelesaikan tes CAPTCHA hanya menggunakan *mouse*. Skema paling sederhananya adalah dengan menampilkan *keypad virtual* dan *user* diminta untuk menekan beberapa tombol yang telah ditentukan dalam *keypad virtual* tersebut.

Berdasarkan Giang, T. & Liu, F, terdapat empat resiko keamanan yang menjadi konsentrasi pembahasan pada penelitian ini yaitu *OCR attacks, dictionary attacks, database attacks,* dan *simple brute force attacks*. *OCR attack* yang dimaksud adalah penggunaan dari aplikasi OCR untuk memindai gambar, memperbaikinya, dan mengambil karakter-karakter yang ditampilkan pada CAPTCHA, kemudian *dictionary attack* adalah metode penyerangan dengan menggunakan kata-kata yang sudah dipersiapkan di dalam sebuah kamus, sedangkan *database attack* adalah metode yang digunakan ketika *database* sumber berhasil dibobol dan kumpulan *file* sumber di dalamnya yang bisa berupa *file* gambar ataupun audio berhasil didapatkan,

dan yang terakhir *simple brute force attack* adalah metode yang digunakan dengan cara mencoba semua kemungkinan solusi yang terdapat di dalam sebuah CAPTCHA. Berikut adalah tabel perbandingan resiko keamanan dan kebutuhan perangkat lunak pendukung antara skema-skema di atas dengan skema *mouse intervention* (Giang, T. & Liu, F, 2006).

Tabel 2.4 Perbandingan Skema CAPTCHA dari berbagai Serangan

Skema CAPTCHA	**Kemungkinan Serangan**			
	OCR	***Dictionary***	***Simple brute force***	***Database***
3D to 2D	No	No	Yes	No
I&C	No	No	No	Yes
Expert *System*	No	Yes	Yes	Yes
MIC	No	No	No	No

Tabel 2.4 di atas menunjukkan apakah suatu skema pengembangan CAPTCHA rentan terhadap sebuah serangan atau tidak. Misalnya skema 3D to 2D yang tidak rentan terhadap *OCR attack, dictionary attack,* dan *database attack,* namun skema ini rentan terhadap *simple brute force attack.*

Tabel 2.5 Perbandingan Kebutuhan Perangkat Lunak Skema CAPTCHA

Skema CAPTCHA	**Kebutuhan Perangkat Lunak**	
	Database	***Image Processing***
3D to 2D	No	Yes
I&C	Yes	No
Expert *System*	Yes	No
MIC	No	No

Tabel 2.5 di atas menunjukkan kebutuhan perangkat lunak dari masing-masing skema, sebagai contoh skema 3D to 2D yang tidak membutuhkan perangkat lunak *database* namun membutuhkan perangkat lunak *image processing*.

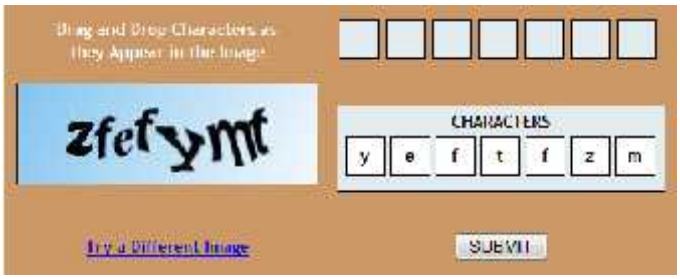
Tabel 2.6 Perbandingan Kemudahan Manusia dalam Membaca CAPTCHA

Skema CAPTCHA	**Mudah Dibaca Manusia**
3D to 2D	No
I&C	No
Expert *System*	No
MIC	Yes

Pada tabel 2.6 ditunjukkan apakah sebuah skema itu mudah dibaca oleh manusia ataupun tidak, dari tabel di atas maka bisa terlihat bahwa MIC (*mouse intervention* CAPTCHA) yang merupakan CAPTCHA yang dikembangkan dengan skema *mouse intervention* mudah dibaca manusia dibandingkan dengan skema lainnya pada tabel.

2.5 *Drag And Drop* CAPTCHA

Kegiatan ini menampilkan teknik CAPTCHA yang menggunakan aksi mouse sebagai pembeda antara manusia dengan komputer. Dalam beberapa penelitian terbaru, terdapat pengembangan terhadap penggunaan parsial *mouse*, yaitu untuk meng-*click* objek spesifik yang bergerak atau meng-*click* bagian dari gambar untuk menyelesaikan uji CAPTCHA. Pendekatan baru ini disebut *drag and drop* CAPTCHA. *Drag and Drop* CAPTCHA ini menggunakan aksi menyeret dan menjatuhkan objek pada wilayah tertentu. Pada tes ini, pengguna dapat menyelesaikan CAPTCHA normal, tetapi tidak bisa menuliskan jawaban pada *textbox*. Sebaliknya, pengguna hanya perlu melakukan *drag and drop* objek pada masing-masing posisi seperti gambar dibawah ini.



Gambar 2.7 Contoh *Drag and Drop* CAPTCHA

Syarat objektif pada perancangan *drag and drop* CAPTCHA adalah (Desai,2009):

1. Kemudahan pengoperasian

   Dalam uji CAPTCHA, pengguna harus melakukan *drag and drop* pada gambar yang dapat diseret (*dragable*). Tidak diperlukan kemampuan khusus untuk menjawab tes, dan dirancang agar pengguna dapat mengerti cara pemakaian tanpa diberikan petunjuk yang spesifik.

2. Mudah diselesaikan manusia

   Untuk melakukan aksi *drag and drop* objek, harus sangat mudah dilakukan oleh manusia tanpa harus menganalisa khusus atau memiliki kemampuan teknik yang khusus.

3. Sulit diselesaikan program komputer

   Perancangan *drag and drop* dan algoritma untuk mengidentifikasi karakter blok dan susunannya harus disesuaikan agar sulit untuk diselesaikan oleh program komputer.

4. Penggunaan *bandwith* yang kecil

   Dengan menggunakan pemrograman dasar berupa HTML dan Javascript, maka konsumsi *bandwith* akan lebih sedikit dibanding penggunaan animasi flash atau video.

5. Mudah diimplementasikan

   perancangan dikembangkan dengan HTML dan Javascript, sehingga mudah pengimplementasiannya. Selain itu pengembangan konsepnya menjadi lebih gampang. Desain ini mempertahankan prinsip: mudah dimengerti, konsumsi *bandwith* kecil, tetapi tetap *powerfull* dalam mengamankan dari serangan *bot*.

2.6 *Jigsaw Puzzle*

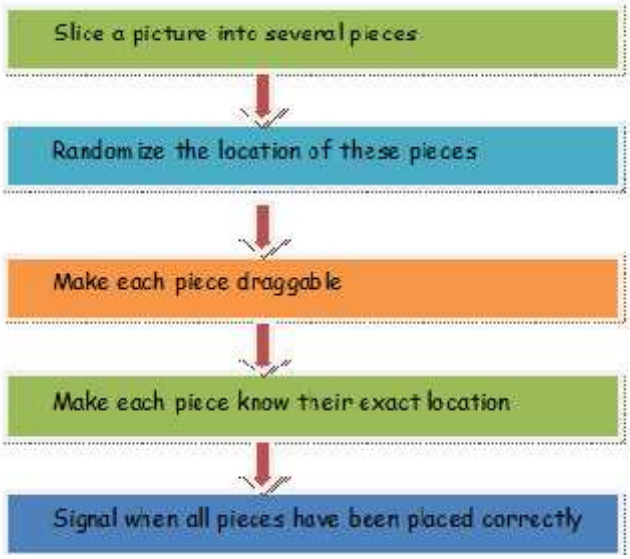
Jigsaw puzzle adalah *puzzle* berupa gambar utuh yang dibagi menjadi beberapa potongan dengan desain khusus dan dapat disambung-sambung kembali membentuk gambar yang utuh. Bentuk potongannya bisa berbagai macam, tidak beraturan dan tidak sama satu dengan yang lain. Banyaknya potongan juga bervariasi, semakin banyak jumlah potongannya, semakin tinggi tingkat kesulitannya. Demikian juga ukuran besar/kecilnya potongan, ikut menentukan tingkat kesulitan *puzzle*. Setelah susunan potongan gambar selesai disusun maka akan ditampilkan gambar yang lengkap.

Jigsaw Puzzle pertama kali diproduksi pada tahun 1766 oleh John Spilsbury seorang ahli pembuat peta. *Puzzle* tersebut berupa peta dan digunakan untuk pembelajaran ilmu geografi bagi anak-anak sekolah. Dengan menyusun

kepingan-kepingan *puzzle* peta tersebut, murid akan belajar tentang lokasi, posisi, dan hubungan geografi antar masing-masing negara.

Gambar 2.8 Contoh *Jigsaw Puzzle*

Untuk membangun sebuah *Jigsaw Puzzle*, diperlukan sebuah mesin yang memiliki algoritma untuk membangkitkan *Jigsaw Puzzle* tersebut. Berikut langkah-langkah cara pembangunannya pada gambar 2.9 di bawah ini.



Gambar 2.9 Pembangunan mesin *Jigsaw Puzzle*

Langkah kerja mesin *jigsaw puzzle* adalah sebagai berikut:

1. *Masking, randomizing dan assigning* dari *drag event handler* dilakukan pada gambar yang disediakan.
2. Potongan gambar ditampilkan di landasannya.
3. Gambar diverifikasi untuk memastikan lokasinya.
4. Setelah pemotongan, pembentukan diaplikasikan pada potongan.

5. Pembentukan area untuk *puzzle* yang akan diselesaikan.
6. Potongan yang telah dipotong diacak lokasinya.
7. *Event handler* mengurus perintah *drag and drop.*
8. Jarak antara kordinat sumber ke tujuan potongan *puzzle* dihitung.
9. Setiap potongan dibandingkan terhadap kordinat untuk menguji apakah telah diseret ke tempat yang tepat.
10. Setelah susunan *drag and drop* lengkap, gambar asli akan ditayangkan dan pesan pernyataan bahwa *puzzle* telah diselesaikan akan ditayangkan.

2.7 HTML5

HTML5 adalah sebuah bahasa yang digunakan untuk mengatur dan menampilkan sebuah konten pada halaman *web*. HTML5 adalah revisi kelima dari HTML standar yang diciptakan tahun 1990 dan di standarkan menjadi HTML4 pada tahun 1997. Salah satu perbedaan HTML5 dari HTML4 adalah diperkenalkannya beberapa elemen *tag* HTML baru seperti article, aside, audio, canvas, datalist, footer, header, section, video, dsb. Selain itu beberapa *Application Programming Interface* (API) baru juga diperkenalkan di dalam HTML5 seperti *web storage*, *offline web application*, *drag and drop*, dan lainnya.

Dari jenis-jenis API HTML5 yang sudah diuraikan pada paragraf di atas, *drag and drop* adalah API HTML5 yang akan diterapkan di dalam pengembangan kali ini. API *drag and drop* dipilih karena dengan mempertimbangkan kelebihannya yaitu HTML5 natif *drag and drop* tidak membutuhkan *third-party plugin* pada sisi *browser* seperti halnya flash yang membutuhkan plugin *flash player* untuk menjalankannya.

2.8 *Rapid Application Development* (RAD)

Rapid application development (RAD) atau *rapid prototyping* adalah model proses pembangunan perangkat lunak yang tergolong dalam teknik *incremental* (bertingkat) (Setiawan,2011). RAD menekankan pada siklus pembangunan pendek, singkat, dan cepat. Waktu yang singkat adalah batasan yang penting untuk model ini. *Rapid application development* menggunakan

metode iteratif (berulang) dalam mengembangkan sistem dimana *working model* (model kerja) sistem dikonstruksikan di awal tahap pengembangan dengan tujuan menetapkan kebutuhan (*requirement*) pengguna. Model kerja digunakan hanya sesekali saja sebagai basis desain dan implementasi sistem akhir.

Rapid Application Development (RAD) adalah metodologi pengembangan perangkat lunak yang berfokus pada membangun aplikasi dalam waktu yang sangat singkat. Istilah ini menjadi kata kunci pemasaran yang umum menjelaskan aplikasi yang dapat dirancang dan dikembangkan dalam waktu 30-90 hari, tapi itu awalnya ditujukan untuk menggambarkan suatu proses pembangunan yang melibatkan *application prototyping* dan *iterative development*.

Tujuan utama dari semua metode sistem *development* adalah memberikan suatu sistem yang dapat memenuhi harapan dari para pemakai, akan tetapi sering kali di dalam melakukan pengembangan suatu sistem tidak melibatkan para pemakai system secara langsung, sehingga hal ini menyebabkan sistem informasi yang dibuat jauh dari harapan pemakai yang dapat berakibat sistem tersebut walaupun dapat diterima tetapi para pemakai enggan untuk menggunakannya atau bahkan para pemakai menolak untuk menggunakannya. Pada saat RAD diimplementasikan, maka para pemakai bisa menjadi bagian dari keseluruhan proses pengembangan sistem dengan bertindak sebagai pengambil keputusan pada setiap tahapan pengembangan. RAD bisa menghasilkan suatu sistem dengan cepat karena sistem yang dikembangkan dapat memenuhi keinginan dari para pemakai sehingga dapat mengurangi waktu untuk pengembangan ulang setelah tahap implementasi.

2.8.1 Kelemahan Metode Konvensional

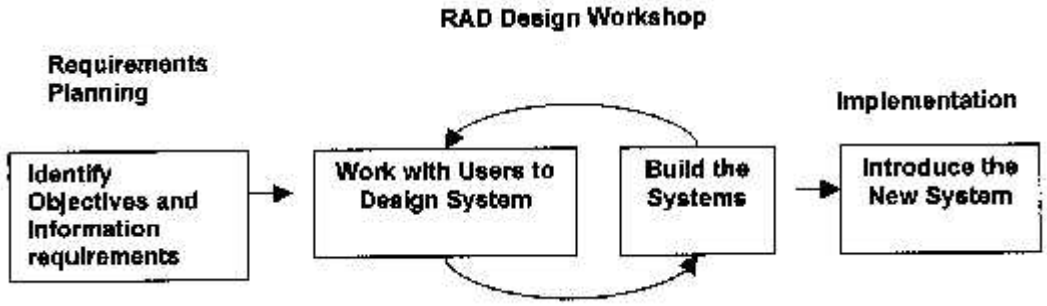
Adapun kelemahan-kelemahan yang terdapat pada metode konvensional adalah sebagai berikut:

1. Dengan metode konvensional, maka terdapat batas waktu yang cukup lama mulai dari pembuatan sistem sampai dengan konsumen dapat menggunakan sistem tersebut.

2. Dengan metode konvensional, apabila proses pengembangan suatu sistem membutuhkan waktu yang lama maka kebutuhan konsumen pada sistem akan mengalami perubahan seiring dengan perubahan proses bisnis yang dilakukan oleh konsumen.
3 Dengan metode konvensional, sistem yang dikembangkan tidak akan mempunyai manfaat apabila belum diselesaikan seluruhnya.

2.9 Tahapan Pada RAD

Metode RAD mempunyai 3 tahapan utama seperti yang terlihat pada gambar 2.10.



Gambar 2.10 Tahapan RAD

2.9.1 Rencana Kebutuhan (*Requirement Planning*)

Pada tahap ini, *user* dan *analyst* melakukan semacam pertemuan untuk melakukan identifikasi tujuan dari aplikasi atau system dan melakukan identifikasi kebutuhan informasi untuk mencapai tujuan. Pada tahap ini hal terpenting adalah adanya keterlibatan dari kedua belah pihak, bukan hanya sekedar persetujuan akan proposal yang sudah dibuat. Untuk lebih jauh lagi, keterlibatan *user* bukan hanya dari satu tingkatan pada suatu organisasi, melainkan beberapa tingkatan organisasi sehingga informasi yang dibutuhkan untuk masing-masing *user* dapat terpenuhi dengan baik. Di samping itu, dapat juga melakukan koordinasi dengan *Chief Information Office* (CIO) atau bagian perencana strategis terutama untuk mengembangkan suatu aplikasi *E-commerce* berbasis *Web* untuk mendapatkan informasi yang lebih detail akan tujuan dari

suatu organisasi. Pertemuan semacam ini seringkali disebut *Joint Aplication Development.*

2.9.2 Proses Desain (*Design Workshop*)

Pada tahap ini adalah melakukan proses desain dan melakukan perbaikan-perbaikan apabila masih terdapat ketidaksesuaian desain antara user dan *analyst*. Untuk tahap ini maka keaktifan *user* yang terlibat sangat menentukan untuk mencapai tujuan, karena user bisa langsung memberikan komentar apabila terdapat ketidaksesuaian pada desain. Biasanya, *user* dan *analyst* berkumpul menjadi satu dan duduk di meja melingkar dimana masing-masing orang bisa melihat satu dengan yang lain tanpa ada halangan.

Apabila memungkinkan, maka masing-masing *user* diberikan satu komputer yang terhubung satu dengan yang lain, sehingga masing-masing bisa melihat desain yang dibuat dan langsung memberikan komentar. Hal ini sering kali disebut dengan *Group Decision Support System* (GDSS). Pada beberapa kasus, GDSS ini merupakan suatu langkah yang ideal, karena user dan analyst dapat menyetujui desain yang dibuat untuk kemudian dilanjutkan oleh *programmer* dalam pembuatan *prototype* dari aplikasi yang dimaksud dengan langsung menampilkan kepada user hasilnya dengan cepat. Pada tahap desain ini membutuhkan waktu beberapa hari, akan tetapi bisa semakin lebih lama, tergantung dari besar kecilnya sistem yang dibuat. Pada selang waktu tersebut, *user* bisa memberikan tanggapan akan sistem yang sudah dikembangkan untuk selanjutnya dilakukan perbaikan-perbaikan. Dengan demikian proses pengembangan suatu sistem membutuhkan waktu yang cepat.

2.9.3 Implementasi (*Implementation*)

Setelah desain dari sistem yang akan dibuat sudah disetujui baik itu oleh user dan *analyst*, maka pada tahap ini *programmer* mengembangkan desain menjadi suatu program. Setelah program selesai baik itu sebagian maupun secara keseluruhan, maka dilakukan proses pengujian terhadap program tersebut apakah terdapat kesalahan atau tidak sebelum diaplikasikan pada suatu organisasi. Pada

saat ini maka user bisa memberikan tanggapan akan sistem yang sudah dibuat serta persetujuan mengenai sistem tersebut. Adapun hal terpenting adalah bahwa keterlibatan *user* sangat diperlukan supaya sistem yang dikembangkan dapat memberikan kepuasan kepada *user*, dan di samping itu, sistem yang lama tidak perlu dijalankan secara paralel dengan sistem yang baru.

2.9.4 Tahapan Keseluruhan

Dengan berdasarkan pada tahapan-tahapan tersebut di atas maka proses utama pengembangan suatu sistem dengan menggunakan metode RAD adalah sebagai berikut :

1. Pengembang membuat *prototype* berdasarkan kebutuhan-kebutuhan yang sudah didefinisikan sebelumnya
2. Desainer melakukan penilaian terhadap *prototype*
3. User melakukan uji coba pada *prototype* dan memberikan masukan mengenai kebutuhan-kebutuhan yang kurang.
4. *User* dan *developer* melakukan pertemuan untuk memberikan penilaian terhadap produk secara bersama-sama, menyesuaikan kebutuhan serta memberikan komentar apabila diperlukan perubahan.
5. Semua kebutuhan akan sistem dan perubahan-perubahan yang terjadi dilakukan proses "*timeboxed*" dengan mempunyai 2 kemungkinan. Pertama, perubahan yang tidak dapat ditampung seperti yang sudah direncanakan harus dihilangkan. Atau kedua, jika diperlukan, kebutuhan-kebutuhan yang bersifat sekunder ditiadakan.

2.10 Unsur-unsur RAD

RAD memiliki banyak unsur-unsur yang membuat sebuah metodologi yang unik termasuk *prototyping, iterative development, time boxing, team members, management approach*, dan RAD *tools*.

2.10.1 *Prototyping*

Sebuah aspek kunci dari RAD adalah pembangunan prototipe untuk tujuan membangkitkan kembali desain untuk kebutuhan pengguna. Tujuannya adalah untuk membangun sebuah fitur ringan yang hasil akhirnya dalam jumlah pendek dengan waktu yang memugkinkan. Prototipe awal berfungsi sebagai bukti konsep untuk klien, tetapi lebih penting berfungsi sebagai titik berbicara dan alat untuk kebutuhan pemurnian. Mengembangkan prototipe cepat dicapai dengan *Computer Aided Engineering CASE tools Software* yang berfokus pada menangkap persyaratan, mengkonversi mereka ke model data, mengubah model data ke database, dan menghasilkan kode semua dalam satu alat. CASE *tools* populer di 80an dan awal 90an, tetapi sebagai teknologi telah berubah (dan COBOL telah menjadi usang) beberapa alat mengambil keuntungan penuh dari potensi penuh dari teknologi KASUS alat. Perusahaan rasional adalah yang paling terkenal meskipun prototipe potensi pembangkitnya terbatas. Pada Otomatis Arsitektur produk cetak biru kami berfokus pada peningkatan tingkat aplikasi enterprise web yang berfungsi sebagai prototipe karena kecepatan yang mereka dapat diciptakan (dalam menit).

2.10.2 *Iterative Development*

Iterative Development berarti menciptakan versi yang lebih fungsional dari sebuah sistem dalam siklus pembangunan pendek. Setiap versi ditinjau dengan klien untuk menghasilkan persyaratan untuk membuat versi berikutnya. Proses ini diulang sampai semua fungsionalitas telah dikembangkan. Panjang ideal iterasi adalah antara satu hari (yang lebih dekat dengan Metodologi Agile) dan tiga minggu. Setiap siklus pengembangan memberikan pengguna kesempatan untuk memberikan umpan balik, memperbaiki persyaratan, dan kemajuan melihat (dalam pertemuan sesi fokus grup). Hal ini akhirnya pembangunan berulang yang memecahkan masalah yang melekat dalam metodologi fleksibel dibuat pada 1970an.

2.10.3 *Time boxing*

Time boxing adalah proses menunda fitur untuk versi aplikasi di masa mendatang untuk melengkapi versi saat ini sebagai ketepatan waktu. Ketepatan waktu merupakan aspek penting dari RAD, karena tanpa itu ruang lingkup dapat mengancam untuk memperpanjang iterasi pembangunan, sehingga membatasi umpan balik dari klien, meminimalkan manfaat dari pembangunan berulang, dan berpotensi mengembalikan proses kembali ke pendekatan metodologi air terjun.

2.10.4 *Team Member*

Metodologi RAD merekomendasikan penggunaan tim kecil yang terdiri dari anggota yang berpengalaman, serbaguna, dan motivasi yang mampu melakukan peran ganda. Sebagai klien memainkan peran penting dalam proses pembangunan, sumber daya klien khusus harus tersedia selama awal *Joint Application Development* (JAD) sesi serta *Focus Group Sessions* dilakukan pada akhir siklus pengembangan. Pengembangan tim (juga dikenal sebagai SWAT atau *Skilled Workers with Advance Tools*) idealnya harus memiliki pengalaman di *Rapid Application Development* dan harus memiliki pengalaman dengan *Computer Aided Software Engineering*. Pendekatan manajemen Aktif dan manajemen yang terlibat sangat penting untuk mengurangi risiko siklus pengembangan diperpanjang, kesalahpahaman klien, dan melebihi tenggat waktu. Di atas manajemen semua harus kuat dan konsisten dalam keinginan mereka untuk menggunakan metodologi *Rapid Application Development*. Selain menegakkan waktu yang ketat, manajemen harus fokus pada pemilihan anggota tim, motivasi tim, dan pada kliring hambatan birokrasi atau politik.

2.10.5 RAD *Tools*

Salah satu tujuan utama dari metodologi *Rapid Application Development* yang dikembangkan oleh James Martin pada tahun 1980an adalah untuk memanfaatkan teknologi terbaru yang tersedia untuk mempercepat pembangunan. Jelas teknologi tahun 1980 sudah kuno, tetapi fokus RAD tentang alat terbaru adalah sama pentingnya hari ini seperti ketika metodologi awalnya diciptakan.

2.11 Model RAD

RAD memiliki tahapan pembangunan yang disebut Model RAD. Model RAD digunakan untuk membantu pengembang perangkat lunak dalam tahap analisa dan pengembangan. Model RAD terbagi 5 yaitu sebagai berikut:

1. Bussiness modeling.

   Aliran informasi di antara fungsi-fungsi bisnis dimodelkan dengan suatu cara untuk menjawab pertanyaan–pertanyaan berikut :

   a. informasi apa yang mengendalikan proses bisnis?
   b. Informasi apa yang di munculkan?
   c. Siapa yang memunculkanya?
   d. Kemana informasi itu pergi?
   e. Siapa yang memprosesnya?

2. Data modeling

   a. Aliran informasi yang didefinisikan sebagai bagian dari fase business modeling disaring ke dalam serangkaian objek data yang dibutuhkan untuk menopang bisnis tersebut.
   b. Karakteristik (disebut atribut) masing – masing objek diidentifikasi dan hubungan antara objek – objek tersebut didefinisikan.
   c. Bagian dari pemodelan bisnis yang didefinisikan ke dalam sekumpulan objek data.
   d. Karakteristik (atribut) dari setiap objek diidentifikasikan dan hubungannya .

3. Prosess modeling

   a. Aliran informasi yang didefinisikan di dalam fase data modeling ditransformasikan untuk mencapai aliran informasi yang perlu bagi implementasi sebuah fungsi bisnis. Gambaran pemrosesan diciptakan untuk menambah, memodifikasi, menghapus, atau mendapatkan kembali sebuah objek data.
   b. Objek data akan diimplementasikan pada fungsi bisnis.
   c. Deskripsi proses dibangun untuk penambahan modifikasi, penghapusan, atau

d. pengambilan kembali objek data.

4. *Application generation*
   a. RAD mengasumsikan pemakaian teknik generasi ke empat. Selain menciptakan perangkat lunak dengan menggunakan bahasa pemrograman generasi ketiga yang konvensional. Pada semua kasus, alat-alat bantu otomatis dipakai untuk memfasilitasi konstruksi perangkat lunak.
   b. Melakukan penggunaan kembali komponen yang ada (jika mungkin).
   c. Membuat kembali penggunaan kembali komponen jika dibutuhkan.
5. *Testing and turnover*

   Karena proses RAD menekankan pada pemakaian kembali banyak komponen program telah diuji. Hal ini mengurangi keseluruhan waktu pengujian. Tetapi komponen baru harus di uji dan semua *interface* harus dilatih secara penuh.

2.12 Keuntungan dan Kerugian RAD

Dalam menggunakan RAD ada beberapa hal yang perlu diperhatikan terutama berkaitan dengan keuntungan dan kerugian. Beberapa keuntungan dalam menggunakan metode RAD adalah sebagai berikut:

1. Membeli sistem yang baru memungkinkan untuk lebih menghemat biaya ketimbang mengembangkan sendiri.
2. Proses pengiriman menjadi lebih mudah, hal ini dikarenakan proses pembuatan lebih banyak menggunakan potongan-potongan *script*.
3. Mudah untuk diamati karena menggunakan model *prototype*, sehingga user lebih mengerti akan sistem yang dikembangkan.
4. Lebih fleksibel karena pengembang dapat melakukan proses desain ulang pada saat yang bersamaan.
5. Bisa mengurangi penulisan kode yang kompleks karena menggunakan *wizard*.
6. Keterlibatan *user* semakin meningkat karena merupakan bagian dari tim secara keseluruhan.

7. Mampu meminimalkan kesalahan-kesalahan dengan menggunakan alat-alat bantuan (*CASE tools*).
8. Mempercepat waktu pengembangan sistem secara keseluruhan karena cenderung mengabaikan kualitas.
9. Tampilan yang lebih standar dan nyaman dengan bantuan *software-software* pendukung.

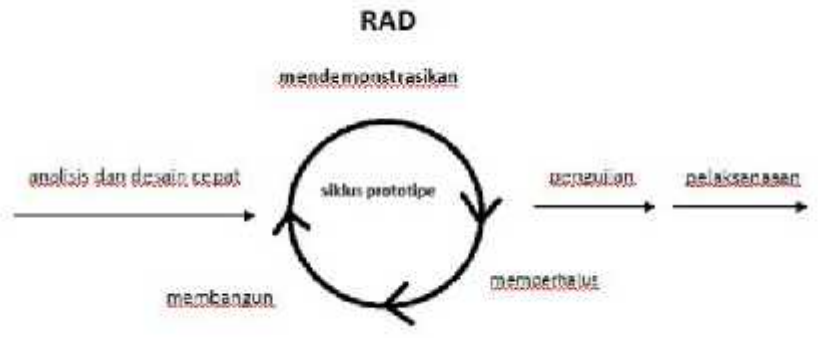
Beberapa kerugian dalam menggunakan metode RAD adalah sebagai berikut :

1. Dengan melakukan pembelian belum tentu bisa menghemat biaya dibandingkan dengan mengembangkan sendiri.
2. Membutuhkan biaya tersendiri untuk membeli peralatan-peralatan penunjang seperti misalnya *software* dan *hardware*.
3. Kesulitan melakukan pengukuran mengenai kemajuan proses.
4. Kurang efisien karena apabila melakukan pengkodean dengan menggunakan tangan bisa lebih efisien.
5. Ketelitian menjadi berkurang karena tidak menggunakan metode yang formal dalam melakukan pengkodean.
6. Lebih banyak terjadi kesalahan apabila hanya mengutamakan kecepatan dibandingkan dengan biaya dan kualitas.

BAB III
METODOLOGI PENELITIAN

Untuk memudahkan penyusun dalam melakukan penelitian, dibutuhkan desain penelitian. Adapun tahapan-tahapan dalam desain penelitian yang dilakukan penyusun dalam proses penelitian skripsi yang berjudul "Rancang Bangun *Image Based* CAPTCHA terintegrasi dengan *Jigsaw Puzzle* menggunakan HTML5" dapat dilihat secara jelas pada gambar 3.1 dibawah ini yang menunjukan rencana atau struktur penelitian yang digunakan untuk memecahkan permasalahan yang diangkat dalam penelitian ini.

Tahapan pertama desain penelitian yang akan dilakukan adalah mengumpulkan data-data yang dibutuhkan. Data-data yang dibutuhkan antara lain adalah informasi tentang CAPTCHA, termasuk karakteristik, jenis-jenis dan tata cara pembuatannya. Selain itu dikumpulkan juga data tentang metode pengembangan perangkat lunak. Setelah data-data dirasa sudah mencukupi, kemudian mempersiapkan alat dan bahan yang akan digunakan untuk penelitian. Setelah persiapan selesai, kemudian masuk ke tahapan pengembangan perangkat lunak. Berikut ini adalah gambar dari desain penelitian serta tahapan-tahapannya yang terdapat di dalam desain penelitian tersebut:



Gambar 3.1 Diagram RAD

Dalam metodologi penelitian dijabarkan tahapan-tahapan yang dilakukan dalam penelitian. Metodologi penelitian terdiri dari beberapa tahapan yang terkait secara sistematis. Tahapan ini diperlukan untuk memudahkan dalam melakukan penelitian. Tahapan yang dilakukan dalam penelitian adalah sebagai berikut :

3.1 Pengumpulan Data

Pada tahapan ini dilakukan persiapan-persiapan penelitian, seperti penyusunan latar belakang, observasi terhadap CAPTCHA yang sudah ada, perumusan masalah serta penentuan ruang lingkup penelitian lainnya. Pada tahapan ini dilakukan studi literatur untuk mencari dan mengumpulkan informasi terkait CAPTCHA dan HTML5. Selain itu juga dilakukan pencarian informasi tentang metode pengembangan perangkat lunak *Rapid Application Development*.

3.2 Analisa Kebutuhan Sistem

Tahap ini merupakan kegiatan mengumpulkan kebutuhan sistem secara lengkap kemudian dianalisa dan didefinisikan kebutuhan yang harus dipenuhi oleh aplikasi yang akan dibangun. Tahap ini harus dikerjakan secara lengkap untuk bisa menghasilkan desain yang lengkap. Setelah tahapan pengumpulan data selesai, langkah berikutnya adalah menganalisa hal-hal yang berhubungan dengan CAPTCHA meliputi:

a. Menganalisa *Flowchart*, Diagram Konteks dan DFD yang menggambarkan sistem kerja CAPTCHA.
b. Menganalisa penggunaan *Jigsaw puzzle* CAPTCHA yang akan diintegrasikan di dalam aplikasi CAPTCHA ini.

3.3 Prototype

Model *prototipe* (*prototyping model*), merupakan suatu teknik untuk mengumpulkan informasi tertentu mengenai kebutuhan-kebutuhan informasi pengguna secara cepat. Berfokus pada penyajian dari aspek–aspek *software* tersebut yang akan nampak bagi pelanggan atau pemakai (contohnya pendekatan *input* dan format *output*). *Prototipe* tersebut dievaluasi oleh pelanggan/pemakai

dan dipakai untuk menyaring kebutuhan pengembangan *software*. Metode dengan menyajikan gambaran yang lengkap tentang sistemnya, pemesan dapat melihat pemodelan sistem dari sisi tampilan maupun teknik prosedural yang akan dibangun. Proses *Prototyping* pada penelitian ini dilakukan dalam 5 langkah:

1. Membuat desain antar muka CAPTCHA
2. Pembuatan *script* untuk membangkitkan gambar CAPTCHA
3. Pengembangan *script* untuk mengacak posisi gambar CAPTCHA.
4. Pembuatan *client side script* untuk menangani *event drag and drop.*
5. Integrasi CAPTCHA yang telah dihasilkan pada sebuah aplikasi *web.*

3.4 Pengujian

Pengujian perangkat lunak ini dilakukan dengan metode pengujian *Black Box,* yaitu dengan cara pengujian fungsi-fungsi yang dimiliki oleh *system*. Pengujian perangkat lunak dilakukan dengan diintegrasikan pada sebuah halaman *form* komentar yang akan dicoba berbagai kemungkinan langkah yang akan diambil. Pengujian yang dilakukan terhadap CAPTCHA yang dihasilkan di dalam penelitian ini terdapat empat yakni pengujian *Blackbox,* pengujian dimensi gambar, pengujian *Security* , dan pengujian *User Acceptance Test.*

3.4.1 Pengujian *Blackbox*

Tahap pengujian *Blackbox* dilakukan dengan tujuan untuk menjamin sistem yang dibuat sesuai dengan hasil analisis dan perancangan serta menghasilkan kesimpulan apakah sistem tersebut sesuai dengan yang diharapkan. Pengujian dibagi dua yaitu pengujian menampilkan *Jigsaw Puzzle* CAPTCHA dan pengujian sistem. Pengujian ini dilakukan setelah CAPTCHA berhasil diintegrasikan pada *form* komentar. Pengujian menampilkan CAPTCHA dilakukan untuk melihat hasil pengembangan dan antarmuka agar sesuai dengan hasil analisa. Pada pengujian sistem terdapat dua skenario, yaitu:

1. Skenario Normal

   Skenario ini digunakan untuk menguji CAPTCHA dengan memasukkan data yang valid dan menyusun CAPTCHA dengan benar. Hasil yang

diharapkan dari pengujian ini adalah data-data registrasi bisa tersimpan ke *database* dan *user* bisa melakukan *submit* dengan sukses.

2. Skenario Alternatif

   Skenario ini digunakan untuk menguji CAPTCHA dengan memasukkan data registrasi yang tidak valid dan menyusun CAPTCHA dengan salah. Hasil yang diharapkan dari pengujian ini adalah data-data registrasi tidak bisa tersimpan ke *database* dan registrasi *user* gagal

Setelah dilakukan pengujian dengan menggunakan dua skenario di atas, maka masing-masing skenario menghasilkan hasil yang sesuai dengan harapan dari tiap-tiap skenario.

3.4.2 Pengujian Dimensi Gambar

Pengujian ini difokuskan kepada penggunaan ukuran gambar sebagai bagian utama dari CAPTCHA yang dikembangkan. Pada pengujian ini, CAPTCHA diberikan situasi dengan mencoba berbagai ukuran gambar yang kemudian akan diimplementasikan oleh sistem CAPTCHA. Hasil dari pengujian ini akan memberikan informasi tentang dampak dari pengaturan berbagai ukuran gambar terhadap CAPTCHA yang dikembangkan. Dimensi gambar yang digunakan dalam pengujian ini dibatasi menjadi 4 ukuran yaitu: 64x64 pixel, 128x128 pixel, 192x192 pixel dan 256x256 pixel.

3.4.3 Pengujian *Security*

Pengujian ini difokuskan kepada keamanan CAPTCHA terhadap berbagai serangan yang umum terjadi pada CAPTCHA konvensional. Metode serangan yang akan diuji adalah OCR *attack* (*Optical Character Recognition*), *Database attack*, *Dictionary attack* dan *simple brute force*. OCR *attack* yang dimaksudkan di sini adalah penggunaan aplikasi OCR untuk mengambil dan mengidentifikasi karakter-karakter yang terdapat pada CAPTCHA. *dictionary* dan *database attack* adalah sebuah metode serangan yang digunakan berdasarkan pada kamus dan *database* yang menjadi sumber data CAPTCHA. Sedangkan pengujian

dengan metode *simple brute force* dengan menggunakan langkah paksa dalam proses *submit* CAPTCHA.

3.4.4 Pengujian *User Acceptance Test*

Pengujian ini difokuskan kepada *feedback* pengguna terhadap CAPTCHA yang telah dikembangkan. Dari hasil pengujian akan mendapatkan penilaian pengguna dari faktor kenyamanan dan kesulitan yang dihadapi saat melakukan penyelesaian CAPTCHA. Berikut beberapa kemungkinan situasi pengujian yang dapat dilakukan pada penelitian ini adalah (Saini,2012):

1. Rata-rata waktu yang diperlukan pengguna untuk menyelesaikan *puzzle* untuk beberapa jenis ukuran matriks. Kondisi ukuran matriks yang akan diujikan adalah: 1x2, 2x2, 2x3, dan 3x3.
2. Ukuran matriks yang paling nyaman diselesaikan oleh pengguna.
3. Perbandingan kenyamanan dan kesulitan penggunaan antara CAPTCHA yang dikembangkan dengan CAPTCHA yang telah ada.
4. Kesulitan yang didapatkan oleh pengguna pada CAPTCHA yang dikembangkan.

3.5 Analisa Hasil Pengujian dan Kesimpulan

Dalam tahap ini dilakukan analisa akhir terhadap hasil pengujian yang telah dilakukan, untuk mengetahui apakah pengembangan *Image Based* CAPTCHA yang terintegrasi dengan *Jigsaw Puzzle* dapat berjalan dan sesuai dengan yang diharapkan.

Pada tahap ini juga ditarik kesimpulan-kesimpulan dalam perancangan dan implementasi keamanan berbasis CAPTCHA tersebut serta saran untuk perbaikan pengembangan tugas akhir ini.

BAB IV
ANALISA DAN PERANCANGAN

Pada perancangan perangkat lunak berbasis komputer, analisa memegang peranan yang penting dalam membuat rincian sistem baru. Analisa perangkat lunak merupakan langkah pemahaman persoalan sebelum mengambil tindakan atau keputusan penyelesaian hasil utama. Sedangkan tahap perancangan sistem adalah membuat rincian sistem hasil dari analisa menjadi bentuk perancangan yang dimengerti pengguna.

Pada bab ini, dilakukan analisa terhadap perancangan CAPTCHA yang akan dikembangkan. Penggunaan metodologi RAD (*Rapid Application Development*) akan menghasilkan 3 model analisa yaitu bisnis model, data model dan proses model. Tahap bisnis model akan membahas tentang aliran kerja CAPTCHA dari awal sampai akhir secara alur bisnis. Pada tahap data model akan menjelaskan aliran data yang di*input* dan dihasilkan pada aplikasi yang dibuat. Tahap ini akan dibantu dengan menggunakan diagram konteks. Pada tahap proses model, akan dijelaskan aliran proses yang lebih detail dengan menggunakan bantuan *data flow diagram*. Dengan begitu diharapkan akan menghasilkan penjelasan data yang lengkap sehingga dapat dijadikan landasan pada pelaksanaan pengembangan CAPTCHA ini.

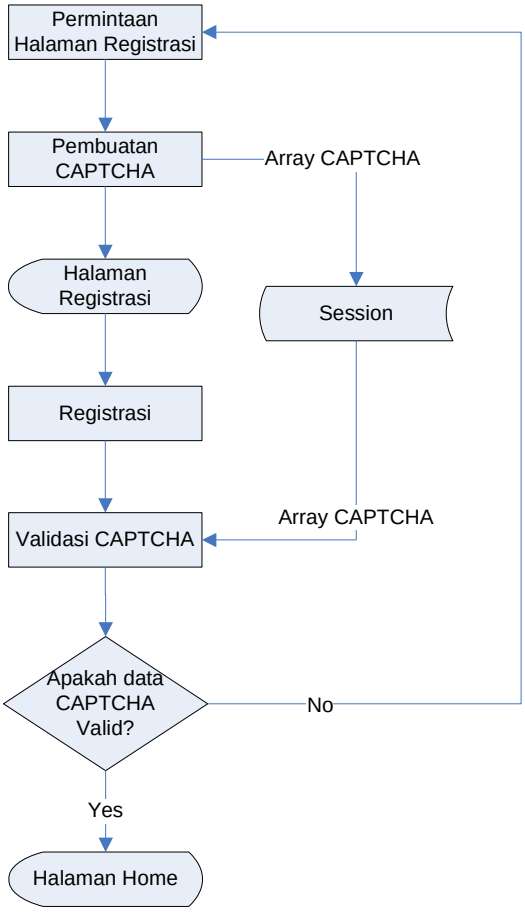
Selanjutnya akan membahas perancangan, mulai dari tahap perancangan antar muka, hingga tahap sistem kerja dari CAPTCHA. Pada tahap ini akan dirangkum semua analisa dan dikonversi kepada langkah kerja yang nyata dalam pengembangan CAPTCHA. Pada bagian akhir akan membahas mengenai teknik pengujian yang akan dilaksanakan dan skema antisipasi yang diajukan agar CAPTCHA yang dihasilkan dapat bertahan menghadapi ancaman serangan CAPTCHA.

4.1 Bisnis Model *Jigsaw puzzle* CAPTCHA

Analisa skema dengan melakukan pengamatan langsung terhadap objek penelitian, yaitu pada CAPTCHA. Observasi tersebut menghasilkan daftar dari kebutuhan yang biasanya ada di setiap CAPTCHA jenis gambar. Disimpulkan fitur yang biasa ada dalam CAPTCHA tersebut adalah sebagai berikut:

1. CAPTCHA terdiri dari tiga bagian utama yaitu bagian pembangkit CAPTCHA, bagian penampil CAPTCHA, dan bagian pengecekkan CAPTCHA.
2. Setidaknya terdapat beberapa proses utama pada file pembangkit CAPTCHA, yaitu pembangkitan string acak, menyimpannya di *session*, pembuatan gambar dari string tersebut, dan mempersiapkannya untuk ditampilkan pada *browser*.
3. Bagian penampil CAPTCHA adalah sebuah bagian yang biasanya terdapat di dalam dokumen HTML yang menampilkan gambar hasil pembangkitan bagian pembangkit CAPTCHA.
4. Ketika pengguna submit data ke *server*, maka terdapat baris kode di *server* yang akan membaca kembali solusi yang diberikan pengguna dan membandingkannya dengan solusi yang benar.
5. Jika solusi dari pengguna sama dengan solusi sebenarnya maka tes CAPTCHA lulus/valid dan proses yang diinginkan bisa dilanjutkan tetapi jika sebaliknya yaitu solusi dari pengguna tidak sama dengan solusi yang sebenarnya maka tes CAPTCHA tidak lulus/tidak valid dan proses yang diinginkan tidak bisa dilanjutkan.

Jigsaw puzzle CAPTCHA ditampilkan dan divalidasi berdasarkan kode yang dimasukkan oleh pengguna berupa langkah-langkah dalam bentuk *flowchart* seperti pada gambar 4.1.



Gambar 4.1 Flowchart *Jigsaw* CAPTCHA

Pada gambar di atas, alur dimulai dari permintaan halaman registrasi oleh pengguna melalui *browser*, setelah itu *form* registrasi dan CAPTCHA dipersiapkan untuk ditampilkan pada halaman registrasi. Ketika proses pembuatan CAPTCHA, susunan valid dari potongan gambar di simpan pada *session* php yang selanjutnya akan digunakan untuk dicocokkan dengan susunan yang diberikan pengguna terhadap CAPTCHA yang sama, jika susunan dari pengguna sama dengan susunan valid pada *session* php maka registrasi sukses dan sebaliknya registrasi gagal dengan catatan bahwa data registrasi lainnya adalah data yang valid.

Untuk jelasnya, langkah kerja image CAPTCHA sebagai berikut:

1. *Generate* Gambar.
2. Pembentukan *Puzzle*.
3. Posisi *puzzle* disimpan di *session.*

4. CAPTCHA ditampilkan ke pengguna.
5. Pengguna menyusun *puzzle* yang ditampulkan di layar hingga selesai.
6. Hasil penyusunan dibandingkan dengan data yang disimpan di *session*.

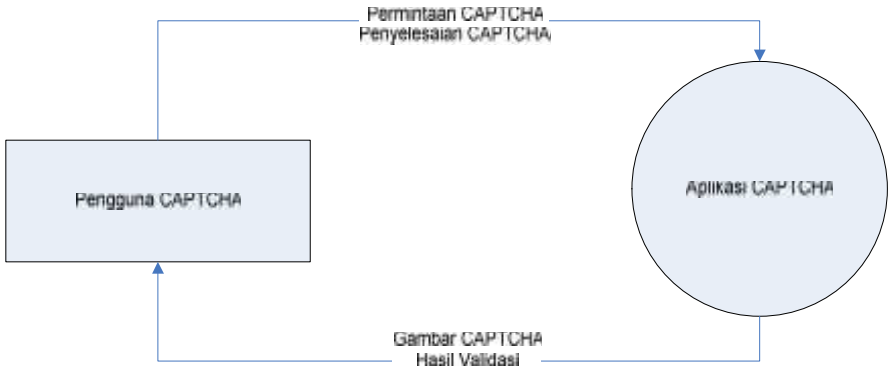
Setelah dilakukan analisa dan observasi, ditentukan kebutuhan dasar sistem adalah:

1. Pengguna dapat melihat CAPTCHA pada sebuah *form*.
2. CAPTCHA yang ditampilkan ke pengguna adalah potongan-potongan gambar

   yang ditampilkan secara acak di dalam sebuah kotak yang seukuran dengan gambar seutuhnya.
3. Pengguna dapat menggeser (*drag*) dan meletakkan (*drop*) potongan gambar pada posisi yang mereka inginkan pada CAPTCHA.
4. Pengguna mengirim (*submit*) *form* dalam keadaan potongan gambar sudah tersusun dengan benar, maka muncul notifikasi berkaitan dengan CAPTCHA dan tombol *submit* untuk memproses data atau perintah.
5. Pengguna tidak dapat mengirim (*submit*) *form* dalam keadaan potongan gambar belum tersusun dengan benar karena tombol *submit* terkunci sebelum *puzzle* selesai diselesaikan. Pengaturan *session* akan membatasi hak akses perintah.

4.2 Data Model *Jigsaw puzzle* CAPTCHA

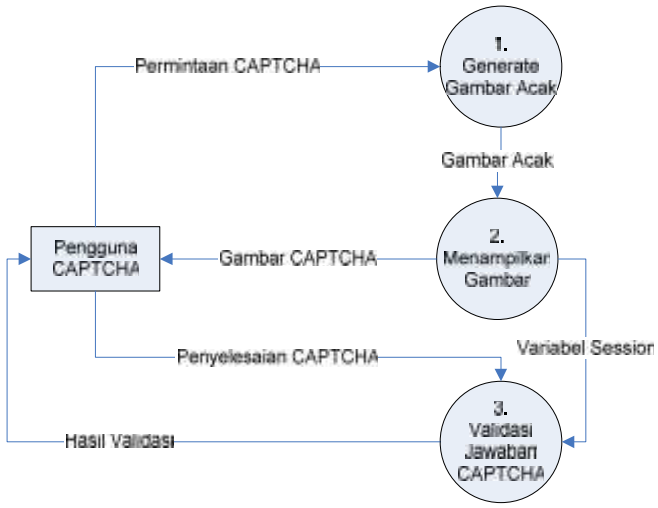
Diagram konteks dan diagram alir data digunakan untuk mendeskripsikan secara fungsional aliran informasi yang ditransformasikan pada saat data bergerak dari *input* menjadi *output*. Diagram konteks selalu mengandung satu proses yang mewakili sistem secara keseluruhan. Pada diagram konteks untuk *Jigsaw puzzle* CAPTCHA, entitas adalah pengunjung situs *web*. Pengunjung mengakses halaman *web* tertentu untuk meminta ditampilkan persoalan CAPTCHA. Berdasarkan teks persoalan yang ditampilkan, pengunjung memasukkan jawaban persoalan. Hasil pemeriksaan dengan membandingkan teks persoalan dengan jawaban persoalan disampaikan kepada pengunjung. Diagram Konteks pada pengembangan *Jigsaw puzzle* CAPTCHA ini dijelaskan pada gambar 4.2.



Gambar 4.2 Diagram konteks CAPTCHA

4.3 Proses Model *Jigsaw puzzle* CAPTCHA

Diagram aliran data digunakan untuk menggambarkan aplikasi CAPTCHA yang akan dikembangkan secara logika. Diagram alir data (DFD Level 1) dari konteks terdiri atas 3 buah proses dan sejumlah aliran data pada proses-proses tersebut.



Gambar 4.3 Diagram alir data level 1

Keterangan dari setiap proses dan aliran data pada DFD level 1 dapat dilihat pada tabel 4.1 sebagai berikut.

Tabel 4.1 Keterangan proses dan aliran data DFD level 1

No	Nama Proses	Masukan	Keluaran	Deskripsi
1	*Generate* Gambar CAPTCHA	Permintaan Gambar	Gambar Acak	Proses meng-*generate* gambar sehingga menghasilkan *Jigsaw puzzle*
2	Tampilkan Gambar	Gambar berupa *Jigsaw puzzle* yang diacak	Posisi Gambar CAPTCHA dan variabel *session*	Proses menampilkan gambar CAPTCHA
3	Validasi CAPTCHA	Posisi gambar Variabel *session*	Hasil validasi	Membandingkan hasil CAPTCHA dengan variabel *session* dan menyampaikan hasil validasi ke pengguna.

4.4 Perancangan *Jigsaw puzzle* CAPTCHA

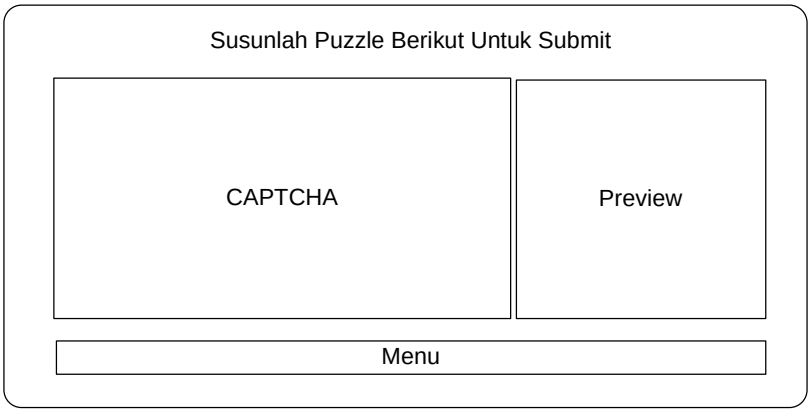
Pada sub bab perancangan akan dilakukan pengembangan *Jigsaw puzzle* CAPTCHA sesuai dengan hasil analisa pada bagian sebelumnya. Berdasarkan analisa yang telah dilakukan, dirangkum 6 langkah pengembangan CAPTCHA yaitu:

1. Membuat desain antar muka CAPTCHA
2. Pembuatan *script* untuk membangkitkan gambar CAPTCHA
3. Pengembangan *script* untuk mengacak posisi gambar CAPTCHA.
4. Pembuatan *client side script* untuk menangani *event drag and drop.*
5. Integrasi CAPTCHA yang telah dihasilkan pada sebuah aplikasi *web.*
6. Perancangan validasi

4.4.1 Perancangan *Interface*

Pada tahap perancangan *interface* ini dihasilkan desain antarmuka CAPTCHA seperti gambar di bawah ini. Gambar yang ada di dalam CAPTCHA

tersebut selanjutnya akan berubah secara acak sesuai sumber gambar inputan.



Gambar 4.4 Keadaan awal antar muka CAPTCHA

Gambar tersebut menunjukkan CAPTCHA dalam keadaan baru terbuka. Pada CAPTCHA tersebut terdapat beberapa bagian, yaitu:

a. Bagian teks judul: Berisi nama dari CAPTCHA itu sendiri. Teks tersebut tidak mengandung tautan.
b. Bagian utama: Bagian ini adalah yang paling utama dari CAPTCHA. Bagian ini mengandung potongan-potongan gambar yang telah diacak-acak, pengguna diminta untuk menyusun potongan gambar tersebut dengan cara *drag and drop* gambar pada posisi yang benar.
c. Bagian *Preview*: Bagian ini menampilkan gambar utuh dari *puzzle* yang ada pada bagian utama sehingga memudahkan pengguna dalam penyusunan *puzzle*.
d. Bagian menu: Bagian ini menampilkan menu antara lain *zoom in*, *zoom out*, dan *refresh*.

Untuk memindahkan potongan gambar pengguna cukup *drag* potongan gambar yang akan dipindahkan dan *drop* potongan gambar tersebut dapat potongan gambar lainnya di dalam CAPTCHA tersebut, maka potongan gambar tempat *drop* akan diganti oleh potongan gambar sebelumnya yang di bawah sebelumnya. pengguna bisa menggeser (*drag*) potongan gambar yang berada di dalam CAPTCHA ke posisi potongan gambar lainnya yang mereka inginkan.

4.4.2 Pembangkit Gambar dan *Session*

Pada bagian ini pembuatan *script* untuk membangkitkan gambar CAPTCHA dimulai. Secara garis besar *script* ini terbagi menjadi beberapa bagian utama, yaitu:

a. Bagian yang membuat gambar baru berdasarkan sumber gambar lain. Pada bagian ini sebuah baris *script* PHP yang menggunakan fungsi pustaka grafis GD imagecreatefromjpeg akan membuat sebuah gambar baru dari sebuah *file* gambar sumber yang diberikan dari *script* lain, gambar baru yang terbuat nantinya akan identik dengan gambar sumber. Gambar baru ini selanjutnya akan dipotong-potong menjadi potongan gambar yang lebih kecil secara beraturan.

b. Bagian yang memecah gambar baru menjadi potongan-potongan gambar lain yang lebih kecil. Bagian ini bertugas untuk memotong-motong gambar baru yang telah dibuat pada bagian pertama di atas. Jumlah potongan gambar yang dihasilkan bisa ditentukan dengan cara mengubah-ubah sebuah variabel col dan row pada parameter fungsi ini. *File* potongan gambar yang dihasilkan diberi nama dengan posisi kolom dan posisi row potongan gambar tersebut secara benar pada gambar utuhnya.

c. Bagian yang menyimpan kembali potongan-potongan gambar ke dalam sebuah *session*. Bagian ini bertugas untuk menyimpan kembali potongan-potongan gambar yang telah dihasilkan ke dalam sebuah *session* yang telah disiapkan.

d. Bagian yang mengirimkan *array* nama-nama *file* potongan gambar ke *browser* untuk ditampilkan. Bagian ini adalah bagian terakhir dari *script*. Bagian ini hanya mengirimkan sebuah *array* yang berisi data-data berkaitan dengan tampilan CAPTCHA data-data tersebut terdiri dari lebar asli gambar sumber, tinggi asli gambar sumber dan *array* nama-nama *file* potongan gambar. Lebar dan tinggi asli gambar sumber diperlukan supaya CAPTCHA ditampilkan di *browser* secara benar dan ukurannya sama dengan *file* sumber gambar.

4.4.3 Pengacakan dan Pembentukan *Puzzle*

Pada bagian ini dikembangkan beberapa baris kode yang akan menampilkan gambar yang dihasilkan oleh script pembangkit potongan gambar tersebut dengan urutan acak pada *browser*. Potongan-potongan gambar tersebut ditampilkan di dalam sebuah kotak yang memiliki ukuran panjang dan lebar sama persis dengan ukuran asli dari gambar utuh potongan-potongan gambar tersebut, sehingga jika user berhasil menyusun potongan-potongan gambar tersebut dengan benar maka akan terlihat gambar utuh dari potongan-potongan gambar tersebut.

Pada bagian pengembalian gambar berisi nama file dari potongan-potongan gambar yang tersimpan di sebuah folder di *web server*. Bagian tersebut berisi array potongan gambar yang masih berurutan dari sisi kiri atas ke kanan dan selanjutnya ke bawah, maka sebuah *script* PHP akan mengacak urutan array potongan gambar tersebut yang selanjutnya akan ditampilkan pada *browser*. Potongan gambar tersebut ditampilkan pada *browser* dengan di bungkus oleh tag html li atau di tampilkan seperti list html. Selanjutnya pada tag html li inilah beberapa event html5 natif *drag and drop* akan dipasang.

Di dalam tag html li tersebut, selain ada potongan gambar, juga akan dibuatkan sebuah *hidden field* yang berisi enkripsi dari nama file tersebut, hal ini dimaksudkan supaya susunan gambar bisa ikut ter-*submit* bersama dengan data *form* yang lain. Jadi ketika user menyusun potongan gambar, maka sebenarnya dia sedang menyusun tag html li yang di dalamnya mengandung potongan gambar dan *hidden field* yang berisi enkripsi nama file potongan gambar tersebut.

4.4.4 Pembuatan *Client Side Script Event*

Pada bagian ini pembuatan *client side script* untuk menangani *event drag and drop* dimulai. *Script* yang dikembangkan dibuat dengan bantuan javascript framework jquery, jquery dipilih karena kemudahannya sehingga pembuatan *script* bisa lebih cepat.

Prinsip dasarnya cukup sederhana, yaitu dengan memberikan beberapa event *drag and drop* pada semua tag html li yang menampilkan potongan gambar, untuk selanjutnya akan didefinisikan *script* yang berkaitan dengan respon apa

yang akan diberikan ketika salah satu event sedang terjadi pada salah satu potongan gambar. Beberapa event html5 natif *drag and drop* tersebut yaitu:

a. *Dragstart*. Event ini terjadi segera setelah pengguna menggeser potongan gambar. Pada saat event ini terjadi maka potongan gambar akan diberi efek tampilan mengecil dengan cara manambahkan kelas css_dragging yang manandakan bahwa potongan gambar sedang digeser.
b. *Dragenter*. Event ini terjadi ketika pointer *mouse* bergerak memasuki elemen target *drop* ketika proses *drag* sedang terjadi.
c. *Dragover*. Event ini terjadi segera setelah pointer *mouse* memasuki elemen target *drop* ketika proses drag sedang terjadi. Pada saat event ini terjadi maka elemen target *drop* akan diberi efek tampilan membesar dengan cara menambahkan kelas css hovered yang manandakan bahwa potongan gambar yang sedang digeser bisa dilepaskan(*drop*) pada elemen target potongan gambar tersebut.
d. *Dragleave*. Event ini terjadi ketika pointer *mouse* meninggalkan elemen target *drop* ketika proses drag sedang terjadi. Pada saat event ini terjadi maka efek tampilan membesar pada elemen target *drop* akan dihilangkan dengan cara menghilangkan kelas hovered yang sebelumnya dipasang pada elemen target *drop* tersebut.
e. *Drag*. Event ini terjadi selama pointer *mouse* bergerak pada proses *drag*.
f. *Drop*. Event ini terjadi pada elemen target *drop* ketika pengguna melepaskan klik kiri mouse(*drop*) selama proses *drag* terjadi. Pada saat event ini terjadi maka posisin potongan gambar yang sedang digeser dengan potongan gambar elemen target *drop* akan saling ditukar.
g. *Dragend*. Event ini terjadi ketika pengguna melepaskan klik kiri *mouse* pada saat proses drag terjadi. *Script* yang dibuat untuk sekumpulan event html5 natif *drag and drop* tersebut di kemas di dalam sebuah jquery plugin supaya mudah untuk memanggil dan mengimplementasikannya pada potongan gambar CAPTCHA.

4.4.5 Integrasi CAPTCHA

Pada bagian ini akan dilakukan integrasi dari CAPTCHA yang telah dihasilkan dengan sebuah *form* komentar pada sebuah aplikasi *web*. Sedangkan aplikasi *web* nya sendiri sudah dikembangkan sebelum penelitian ini, pada tahap ini hanya dilakukan integrasi CAPTCHA nya saja. Aplikasi *web* tersebut adalah sebuah aplikasi sederhana yang digunakan untuk percobaan *submit* data. Aplikasi ini dikembangkan dengan menggunakan php sederhana.

Integrasi antara CAPTCHA dengan aplikasi ini dilakukan hanya dengan cara memanggil fungsi CAPTCHA yang bertugas untuk menampilkan CAPTCHA pada browser.

4.5 Langkah Proteksi CAPTCHA

Langkah langkah proteksi yang digunakan *Jigsaw puzzle* CAPTCHA dan validasinya terhadap serangan dan potensi serangan masalah dengan menggunakan CAPTCHA sebagai berikut:

1. Penggunaan skema *mouse intervention*. Fungsi *keyboard* dialihkan kepada *mouse*, sehingga *input* CAPTCHA tidak bisa dengan sistem paksa atau *brute force*, tetapi harus dilakukan manual dari penyusunan hingga gambar tersusun sempurna.
2. Penggunaan gambar sebagai pengganti teks. CAPTCHA konvensional biasanya menggunakan teks dalam pelaksanaannya. Saat ini pemecahannya telah banyak ditemukan yaitu dengan menggunakan OCR, sehingga pada penelitian ini digunakan gambar sebagai pengganti teks, tujuannya agar OCR tidak bisa membaca karena tidak ada karakter pada gambar CAPTCHA.
3. Validasi *session*. Gambar yang diolah menjadi *puzzle* dipotong berdasarkan posisi kolom dan baris, kemudian hasilnya akan disimpan di *session*. Saat pengguna berhasil menyelesaikan *puzzle*, maka hasil posisi potongan akan dibandingkan dengan *session*.

BAB V
IMPLEMENTASI DAN PENGUJIAN

5.1 Implementasi

Implementasi *Jigsaw Puzzle* CAPTCHA dilakukan berdasarkan hasil analisa dan perancangan *Jigsaw Puzzle* CAPTCHA pada Bab Analisa dan Perancangan, dengan menggunakan bahasa pemrograman PHP, HTML5 dan Javascript, sehingga akan diketahui apakah CAPTCHA ini benar-benar dapat mencapai tujuan yang diinginkan.

5.1.1 Lingkungan Implementasi

Lingkungan implementasi untuk pengembangan *Jigsaw* CAPTCHA ini mencakup *Hardware* (perangkat lunak komputer) dan *Software* (perangkat lunak). Berikut adalah rincian lingkungan implementasi yang digunakan:

1. Hardware
   a. Processor : Intel Atom Processor 1.5GHz
   b. Memory : DDR2 2GB
   c. Harddisk : 160 GB
2. Software
   a. Sistem operasi : Windows XP Professional SP2
   b. Web Server : Apache 2.0
   c. Web Browser : Google Crome Version 26.0.1410.43 /

5.1.2 Hasil Implementasi

CAPTCHA yang dihasilkan adalah dalam berbentuk *plugin* php, dimana CAPTCHA bisa dipasangkan pada aplikasi *web* lain yang berbasis php selain dari aplikasi yang digunakan pada penelitian ini. Beberapa parameter harus disesuaikan ketika melakukan integrasi dengan sebuah aplikasi *web* diantaranya seperti perameter lokasi folder tempat penyimpanan file gambar sumber, lokasi folder tempat penyimpanan *temporary file* potongan gambar, dan lainnya.

Implementasi *Jigsaw Puzzle* CAPTCHA sebagai berikut:

1. File Index.php. File ini menampilkan *textbox* dan CAPTCHA.
2. File Paper.php. File inti yang berisikan langkah kerja pembentukan puzzle, sistem kerja CAPTCHA, validasi CAPTCHA dan canvas sebagai media untuk menampilkan CAPTCHA.
3. File gambar dengan format PNG berdimensi 128x128 *pixel*.
4. File jquery.js sebagai library javascript.

Hasil implementasi tampilan *form Jigsaw puzzle* CAPTCHA sebagai berikut:

Gambar 5.1 *Form Jigsaw Puzzle* CAPTCHA

5.2 Pengujian

Pengujian dilakukan untuk mengetahui apakah *Jigsaw puzzle* CAPTCHA dapat mencapai tujuan yang diinginkan dan menemukan kesalahan-kesalahan yang mungkin terjadi pada bagian implementasi.

5.2.1 Pengujian *Blackbox*

CAPTCHA yang telah dikembangkan selanjutnya akan diuji dengan menggunakan metode *blackbox*. Tahap pengujian dilakukan dengan tujuan untuk

menjamin sistem yang dibuat sesuai dengan hasil analisis dan perancangan serta menghasilkan kesimpulan apakah sistem tersebut sesuai dengan yang diharapkan. Pengujian dibagi dua yaitu pengujian menampilkan *Jigsaw Puzzle* CAPTCHA dan pengujian sistem CAPTCHA. Berikut hasil pengujian menampilkan *Jigsaw Puzzle* CAPTCHA pada tabel 5.1.

Tabel 5.1 Pengujian Menampilkan CAPTCHA

Deskripsi	**Prosedur Pengujian**	***Input***	**Keluaran yang diharapkan**	**Kriteria evaluasi hasil**	**Hasil yang didapatkan**
Pengujian menampilkan CAPTCHA	1. Masukkan alamat index.php pada *browser* 2. Apabila alamat benar, maka tampil *form* dan CAPTCHA	-	Tampilnya *form* dan CAPTCHA	Tampilnya *form* dan CAPTCHA	Tampilnya *form* dan CAPTCHA

Selanjutnya Pengujian sistem dilakukan setelah CAPTCHA berhasil diintegrasikan pada *form* registrasi. Terdapat dua skenario pada pengujian ini, yaitu:

1. Skenario Normal

   Skenario ini digunakan untuk menguji CAPTCHA dengan memasukkan data registrasi yang valid dan menyusun CAPTCHA dengan benar. Hasil yang diharapkan dari pengujian ini adalah data-data registrasi bisa tersimpan ke *database* dan *user* bisa registrasi dengan sukses. Hasil pengujian dengan menggunakan skenario normal ini ditampilkan pada gambar 5.2.

Gambar 5.2 Hasil pengujian skenario normal

2. Skenario Alternatif

   Skenario ini digunakan untuk menguji CAPTCHA dengan memasukkan data registrasi yang tidak valid dan menyusun CAPTCHA dengan salah. Hasil yang diharapkan dari pengujian ini adalah data-data registrasi tidak bisa tersimpan ke *database* dan registrasi *user* gagal karena adanya penguncian tombol *submit* yang tidak ditampilkan sebelum CAPTCHA berhasil diselesaikan.

Tabel 5.2 Skenario Pengujian Sistem CAPTCHA

Deskripsi	**Prosedur Pengujian**	**Masukan**	**Keluaran yang diharapkan**	**Kriteria evaluasi hasil**	**Hasil yang didapatkan**
Pengujian skenario blackbox	Susun *Puzzle* untuk melakukan *submit*	*Puzzle* benar	Tampil notifikasi dan tombol *submit*	Tampil notifikasi dan tombol *submit*	Tampil notifikasi dan tombol *submit*
		Puzzle belum selesai	Tombol *submit* tidak ditampilkan	Tombol *submit* tidak ditampilkan	Tombol *submit* tidak ditampilkan

Setelah dilakukan pengujian dengan menggunakan dua skenario di atas, maka masing-masing skenario menghasilkan hasil yang sesuai dengan harapan dari tiap-tiap skenario.

5.2.2 Pengujian Dimensi Gambar

Pengujian ini difokuskan kepada penggunaan ukuran gambar sebagai bagian utama dari CAPTCHA yang dikembangkan. Pada pengujian ini, CAPTCHA diberikan situasi dengan mencoba berbagai ukuran gambar yang kemudian akan diimplementasikan oleh sistem CAPTCHA. Hasil dari pengujian ini akan memberikan informasi tentang dampak dari pengaturan berbagai ukuran gambar terhadap CAPTCHA yang dikembangkan. Ukuran Matriks yang digunakan dibatasi hanya menggunakan matriks 2x2 dan dimensi gambar yang digunakan dalam pengujian ini dibatasi menjadi 4 ukuran yaitu: 64x64 *pixel*, 128x128 *pixel*, 192x192 *pixel* dan 256x256 *pixel*. Berikut ini hasil pengujian dimensi gambar.

Gambar 5.3 Pengujian menggunakan gambar berdimensi 64x64 *pixel*

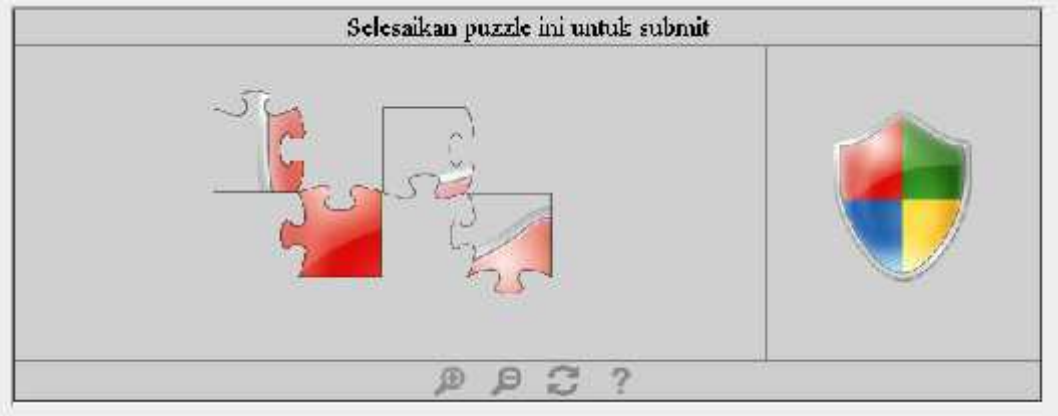
Pada penggunaan gambar berdimensi 64x64 *pixel*, terlihat bahwa gambar akan disesuaikan dengan dimensi yang *default* pada *puzzle*. Gambar ditampilkan secara utuh tetapi ada penambahan ruang kosong agar ukuran potongan gambar membentuk *puzzle* yang sempurna.

Gambar 5.4 Pengujian menggunakan gambar berdimensi 128x128 *pixel*

Pada penggunaan gambar berdimensi 128x128 *pixel*, terlihat bahwa gambar akan dibentuk menjadi potongan *puzzle* dengan sempurna. Oleh sebab itu gambar yang direkomendasikan pada pengembangan CAPTCHA pada penelitian ini adalah menggunakan gambar berdimensi 128x128.

Gambar 5.5 Pengujian menggunakan gambar berdimensi 192x192 *pixel*

Pada penggunaan gambar berdimensi 192x192, terlihat bahwa gambar akan disesuaikan dengan dimensi *default* pada *puzzle*, sehingga gambar akan dipotong dan hanya menampilkan sebagian gambar. Gambar yang ditampilkan tidak sempurna seperti pada bagian *preview*.



Gambar 5.6 Pengujian menggunakan gambar berdimensi 256x256 *pixel*

Pada penggunaan gambar berdimensi 256x256, terlihat bahwa gambar akan disesuaikan dengan dimensi *default* pada *puzzle*, sehingga gambar akan dipotong dan hanya menampilkan bagian dari ujung kiri atas gambar. Gambar yang ditampilkan tidak sempurna seperti pada bagian *preview*.

Dari hasil pengujian diatas dapat disimpulkan bahwa ukuran gambar mempengaruhi hasil tampilan pada potongan *puzzle*. Pada gambar berdimensi 64x64 *pixel*, gambar ditampilkan secara utuh namun ada penambahan ruang kosong agar ukuran *puzzle* sesuai dengan format yang telah ditentukan. Penggunaan ruang kosong ini tidak sesuai dengan gambar yang ditampilkan pada bagian *preview*. Pada gambar berdimensi 128x128*pixel*, gambar ditampilkan utuh dan sempurna seperti pada bagian *preview*. Pada gambar berdimensi 192x192 *pixel* dan 256x256 *pixel*, gambar yang ditampilkan tidak utuh dan dipotong mulai dari sudut kiri atas hingga membentuk format standar *puzzle* pada CAPTCHA. Sehingga ukuran gambar yang direkomendasikan pada CAPTCHA yang dikembangkan pada penelitian ini adalah 128x128.

5.2.3 Pengujian *User Acceptance Test*

Pengujian ini difokuskan kepada *feedback user* terhadap CAPTCHA yang telah dikembangkan. Dari hasil pengujian akan mendapatkan penilaian pengguna dari faktor kenyamanan dan kesulitan yang dihadapi saat melakukan penyelesaian

CAPTCHA. Responden pengujian dipilih sebanyak 10 orang dengan syarat yang telah ditentukan yaitu responden harus terbiasa menggunakan perangkat komputer dan telah mengenal CAPTCHA. Beberapa tujuan yang ingin dicapai pada pengujian ini sebagai berikut:

1. Rata-rata waktu yang diperlukan pengguna untuk menyelesaikan *puzzle* untuk beberapa jenis ukuran matriks. Kondisi ukuran matriks yang akan diujikan adalah: 1x2, 2x2, 2x3, dan 3x3.
2. Ukuran matriks yang direkomendasikan oleh pengguna.
3. Perbandingan kenyamanan dan kesulitan penggunaan antara CAPTCHA yang dikembangkan dengan CAPTCHA yang telah ada.
4. Kesulitan yang didapatkan oleh pengguna pada CAPTCHA yang dikembangkan.

Pengujian dilakukan pada *web browser* dan tanggapan pengguna didapatkan dari sebuah *form User Acceptance Test* yang terdapat pada lampiran C. Hasil dari pengujian adalah sebagai berikut:

1. Rata-rata waktu (dalam detik) untuk menyelesaikan CAPTCHA sangat beragam tergantung ukuran matriks yang digunakan. Hasil pengujian tercantum pada tabel 5.3.

Tabel 5.3 Rata-rata waktu pengujian berbagai ukuran matriks CAPTCHA

Matriks	**1x2**	**2x2**	**2x3**	**3x3**
Waktu (detik)	2.06	8.49	14.48	41.06

2. Ukuran matriks yang direkomendasikan adalah 2x2 sebanyak 70% dari responden.
3. Faktor kenyamanan menunjukkan 100% responden lebih nyaman menggunakan CAPTCHA ini dibandingkan CAPTCHA konvensional yang berbasis teks.

Dari butir uji diatas yang berjumlah 10 Pengujian dapat diketahui bahwa tingkat keberhasilan dalam penyelesaian CAPTCHA sebesar 100%. Kelemahan yang terjadi umumnya karena pengguna membutuhkan waktu pada saat awal pengenalan CAPTCHA ini. Namun untuk percobaan selanjutnya pengguna dapat

menyelesaikan dalam waktu yang relatif singkat. Masalah utama terjadi saat responden menerima matriks dengan ukuran lebih besar sehingga ukuran potongan *puzzle* semakin kecil dan rumit. Hal ini mengakibatkan responden mengalami kesulitan untuk meletakkan potongan *puzzle* pada posisi yang benar. Masalah lainnya terjadi ketika *background* transparan menjadi salah satu bagian dari potongan *puzzle*. Namun dengan pengaturan potongan *puzzle* yang bergelombang dapat membantu responden untuk meletakkan potongan pada posisi yang sebenarnya.

5.2.4 Pengujian *Security*

Berdasarkan penjelasan pada bab II, bahwa terdapat empat jenis resiko yang berkaitan dengan keamanan, empat hal tersebut adalah OCR *attack*, *dictionary attack*, *database attack* dan *simple brute force attack*. OCR *attack* yang dimaksud adalah penggunaan dari aplikasi OCR untuk mengidentifikasi karakter-karakter yang ditampilkan pada CAPTCHA, hal ini berbeda dengan CAPTCHA yang dihasilkan dengan skema *mouse intervention*, di mana CAPTCHA yang dihasilkan adalah CAPTCHA *image based* yang pada pengoperasiannya tidak berlandaskan pada karakter-karakter melainkan pada identifikasi sebuah gambar utuh dari sekumpulan potongan yang tak berurut. Hal tersebut menyebabkan CAPTCHA yang dihasilkan tidak rentan terhadap OCR *attack*. Sedangkan *dictionary attack* yang dimaksud adalah penggunaan dari kamus untuk membobol CAPTCHA, hal ini tidak dimungkinkan karena CAPTCHA yang dihasilkan sendiri tidak menggunakan sebuah kamus dalam pengoperasiannya.

Tabel 5.4 Pengujian *Security*

Skema Serangan	Tahan Terhadap Serangan
OCR	Yes
Database Attack	Yes
Dictionary Attack	Yes

Berbeda dengan kedua resiko keamanan di atas, pengembangan menggunakan HTML5 menyebabkan struktur dari kode HTML mudah diketahui hal ini menyebabkan cukup mudah untuk mengetahui proses pengecekkan CAPTCHA yakni dengan pengecekkan *hidden field* yang berisi data-data urutan potongan gambar yang benar, maka dengan mengetahui struktur kode html dan proses pengecekkan, dimungkinkan untuk dilakukannya *brute force* terhadap urutan dari potongan gambar yang benar. Hal tersebut diantisipasi dengan cara selain dilakukan pengecekkan terhadap urutan potongan gambar yang benar juga dilakukan dengan cara penghapusan terhadap data *session* urutan potongan gambar yang benar yang tersimpan pada *server* setiap kali pengaksesan. Hal ini bertujuan supaya urutan potongan gambar yang benar pada *session* berubah-ubah setiap kali pengaksesan.

Berdasarkan paragraf di atas, selanjutnya dilakukan pengujian untuk mensimulasikan *simple brute force attack*. Pengujian tersebut dilakukan dengan cara membuat *script* yang mampu memaksakan mengirim data registrasi bersamaan tanpa menyelesaikan Jigsaw Puzzle CAPTCHA. Namun berdasarkan hasil pengujian dinyatakan bahwa serangan *simple brute force* dapat diatasi dengan langkah proteksi seperti yang telah dijelaskan pada bab IV.

Tabel 5.5 Pengujian *simple brute force*

Deskripsi	**Prosedur Pengujian**	***Input***	**Keluaran yang diharapkan**	**Kriteria evaluasi hasil**	**Hasil yang didapatkan**
Pengujian simple *Brute force*	Menambahkan paksa tombol *submit*	*Puzzle* belum selesai	Tampil notifikasi CAPTCHA belum diselesaikan	Tampil notifikasi CAPTCHA belum diselesaikan	Tampil notifikasi CAPTCHA belum diselesaikan

5.3 Analisa Hasil Pengujian

Pada akhirnya pengembangan ini menghasilkan Sebuah CAPTCHA Baru dengan skema *mouse intervention* menggunakan HTML5. CAPTCHA yang telah

dikembangkan adalah berasal dari sebuah gambar utuh yang dipotong-potong segi empat secara teratur, potongan-potongan gambar tersebut kemudian ditampilkan kepada pengguna pada sebuah kotak yang memiliki ukuran sama dengan ukuran gambar utuhnya dan ditampilkan kepada urutan acak. Selanjutnya untuk melalui tes CAPTCHA ini, pengguna cukup menyusun potongan-potongan gambar yang ditampilkan dengan cara *drag and drop* pada potongan gambar tersebut. Pengguna bisa melakukan *submit* hanya ketika *puzzle* berhasil diselesaikan. CAPTCHA yang telah berhasil dikembangkan kemudian diintegrasikan dengan sebuah aplikasi *web*.

Setelah melalui tahapan penelitian, pengembangan dan pengujian, terdapat beberapa kelebihan pada CAPTCHA tersebut. Berikut adalah kelebihan dari CAPTCHA yang dikembangkan pada penelitian ini:

1. Sebagaimana dengan yang dijelaskan oleh Trier Giang dan Fei Lui (2006) bahwa CAPTCHA dengan skema *mouse intervention* tidak rentan terhadap OCR *attack, Dictionary attack, Database attack* dan *Simple Brute Force*.
2. Dari hasil pengujian dimensi gambar dapat disimpulkan bahwa ukuran gambar mempengaruhi hasil tampilan pada potongan *puzzle*. Ukuran gambar yang direkomendasikan adalah 128x128 *pixel*.
3. Penerapan teknologi HTML5 terhadap CAPTCHA dengan skema *mouse intervention* menghilangkan kebutuhan *third party plugi*n pada sisi *browser*.
4. Hasil pengujian UAT diperoleh 100% responden mengungkapkan metode CAPTCHA ini lebih menarik dan nyaman digunakan dibandingkan CAPTCHA konvensional yang berbasis teks. Hasil pengujian UAT juga menyatakan penggunaan matriks 2x2 lebih direkomendasikan oleh responden. Hasil ini didapatkan dengan pertimbangan kenyamanan penggunaan dan waktu penyelesaiannya.

BAB VI
PENUTUP

6.1 Kesimpulan

Kesimpulan yang dapat diambil dalam Tugas Akhir ini sebagai berikut:

1. Pengembangan CAPTCHA dengan skema *mouse intervention* dilakukan dengan metode pengembangan perangkat lunak *Rapid Application Development* telah menghasilkan sebuah CAPTCHA baru yang tidak rentan terhadap OCR *attack*, *dictionary attack*, dan *database attack*, serta aman terhadap *simple brute force attack.*
2. Penggunaan *Jigsaw Puzzle* pada CAPTCHA berbasis *image* dapat memberikan kenyamanan dari sisi pengguna dalam menyelesaikan persoalan CAPTCHA.
3. Dari hasil Pengujian UAT didapatkan hasil bahwa ukuran matriks yang direkomendasikan oleh responden adalah 2x2. Hasil ini didapatkan dengan pertimbangan kenyamanan penggunaan dan waktu penyelesaiannya.

6.2 Saran

Saran-saran untuk pengembangan *Puzzle* CAPTCHA, sebagai berikut:

1. Pada penelitian ini ditemukan bahwa penggunaan berbagai dimensi ukuran gambar dan ukuran matriks mempengaruhi tampilan pada *Jigsaw Puzzle* CAPTCHA. Sehingga penulis menyarankan untuk diadakannya penelitian berikutnya yang membahas tentang peningkatan fleksibilitas ukuran dan dimensi pada *Jigsaw Puzzle* CAPTCHA.
2. Penelitian CAPTCHA ini hanya didesain untuk penggunaan *web browser* pada perangkat komputer, hal ini mengakibatkan tampilan CAPTCHA pada perangkat lain seperti *handphone*, tablet, ataupun perangkat *mobile* lainnya belum dibahas. Sehingga tampilan serta pengoperasiannya pun belum sama dengan CAPTCHA pada

komputer. Maka dari itu diperlukan penelitian lebih lanjut yang mengkaji kompatibilitas dari CAPTCHA ini pada berbagai perangkat baik perangkat komputer maupun *mobile*.

3. CAPTCHA pada penelitian ini belum dikembangkan ke arah penggunaan API Key seperti pada aplikasi reCAPTCHA. konsep *web service* yang digunakan reCAPTCHA pun bisa diteliti dan selanjutnya diterapkan pada CAPTCHA dengan skema *mouse intervention* pada penelitian ini.

DAFTAR PUSTAKA

Ahn, L., et al. “Telling Humans and Computers Apart Automatically”. hlm. 57-60, 2004.

Banday M. Tariq Banday, dan N. A. Shah. “A Study of CAPTCHAs for Securing Web Services “,IJSDIA *International Journal of Secure Digital Information Age*, Vol. 1. No. 2, December 2009 .

CAPTCHA Official Site. CAPTCHA [Online]. Tersedia:http://www.captcha.net. 14 April 2010.

Desai, A dan Patadia, P. “Drag and Drop: A Better Approach to CAPTCHA” India:A.D. Patel Institue of Technology. 2009.

G. Moly, N. Jones, C. Harkless, and R. Potter, "Distortion Estimation Techniques in Solving Visual CAPTCHAs," in IEEE Computer Society conference on Computer Vision and Pattern Recognition(CVPR '04), hlm 23-28, 2004.

Giang, T & Liu, F. “*Mouse Intervention CAPTCHA”* Proceedings of iiWAS2006 , hlm 155-162. 2006.

H. S. Baird, D. Lopresti, B. D. Davison, and W. M. Pottenger, "Robust Document Image Understanding Technologies," in 1st ACM Workshop on Hardcopy document processing, Washington DC, USA, hlm 9-14, 2004.

Heriyanto. (2011). “Pengembangan perangkat lunak sebagai alat pembuktian solusi metode peta karnaugh untuk menunjang matakuliah elektronika digital”. Skripsi pada jurusan Fisika FMIPA Universitas Negeri Malang : tidak diterbitkan

Jigsaw Puzzle [Online], http://www.omochatoys.com/mainan-edukatif/496-puzzle.html, November 2012.

Michael Southwell dan Chris Snyder, “*Pro PHP Security*”, Apress Publisher, 2005.

Moni Naor. *Verification of a human in the loop or identification via the turing test*. Unpublish manuscript, Sept 1996.

Mori, G., dan Malik, J. (2003) "Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA,". IEEE CS Society Conference on Computer Vision and Pattern Recognition

(CVPR'03), Madison, WI, hlm 134-141, 2003

Payal Nitisha, Nidhi Chaudhary, dan Parma Nand Astya. "JigCAPTCHA: An Advanced Image-Based CAPTCHA Integrated with Jigsaw Piece Puzzle using AJAX" , *International Journal of Soft Computing and Engineering* (IJSCE), ISSN: 2231-2307, Volume-2, Issue-5, November 2012.

Rick Lehtinen, "*Computer Security Basic, 2nd Edition*", O'Reilly, juni 2006

Saini, Baljit Singh. "THE JIGSAW CAPTCHA", *International Journal Of Research In Computer Application & Management*, Vol No. 2, Issue No. 6, Juni 2012.

Sarijo. (2011). *Metode Penelitian dan Pengembangan(Research and Development/R&D)*. Tersedia : "http://greensarijo.blogspot.com/2011/07/metode-penelitian-dan-pengembangan.html". [30 Oktober 2012]

Setiawan Eko Budi. "Optimalisasi Keamanan Website menggunakan CAPTCHA-AD video", Jurnal Komputer dan Informatika (KOMPUTA) Edisi. I Volume. 1, Maret 2012.

Setiawan Ade, "Rapid Application Development", Universitas Gunadarma, 2011

Shah N.A., Banday M.T. (2008). "Drag and Drop Image CAPTCHA ," .Sprouts: Working Papers on Information Systems, 8(46). http://sprouts.aisnet.org/8-46,
Proceedings of 4th J&K Science Coongress 12th to 114th Nov, 2008.

Soni Rituraj, dan Devendra Tiwari. "Improved CAPTCHA Method", *International Journal of Computer Applications*, Volume 1 – No. 25 , 2010.

Tanvee , Moin Mahmud, Mir Tafseer Nayeem, dan Md. Mahmudul Hasan Rafee. "Move & Select: 2-Layer CAPTCHA Based on Cognitive Psychology for Securing Web Services", International Journal of Video & Image Processing and Network Security IJVIPNS-IJENS Vol: 11 No: 05 , Oktober 2011