BAB I
PENDAHULUAN

1.1.Latar Belakang Masalah

Automatic grading system adalah sistem penilaian atau assessment secara otomatis. Automatic grading system dapat membantu menilai sebuah source code. Ada beberapa *tools* yang dapat melakukan penilaian otomatis yang telah tersedia diantaranya CodingBat, betterprogrammer, Practice-It, Web-CAT dan Marmoset. Salah satu keuntungan dari *tools* tersebut adalah dapat mendapatkan timbal balik secara langsung (*realtime*) tentang hasil penilaian dari *source code* yang telah diajukan oleh pelajar, *tools* ini juga menyediakan tempat penyimpanan data pengguna agar dapat menyimpan data pekerjaan yang telah dikerjakan.

Sistem *automatic grading* dapat digunakan untuk membantu dosen dalam menilai sebuah *source code* yang dibuat oleh mahasiswanya. Tujuan penilaian ini adalah untuk mengetahui kemampuan mahasiswa dalam membuat program. Penilaian dilakukan dengan melihat kriteria penilaian. Kriteria penilaian yang ada mencakup dasar-dasar dalam belajar pemrograman seperti kebenaran program ,*code coverage* ,dan kesalahan sintak. Pada setiap kriteria penilaian dibutuhkan bobot penilaian untuk membedakan nilai terpenting dari *source code* yang dinilai. Sistem ini menyediakan beberapa kriteria penilaian dengan bobot peniaian yang di isi langsung oleh dosen sebelum melakukan proses penilaian .

Dalam bidang informatika, pembuatan *source code* adalah salah satu bagian terpenting dalam mempelajari bahasa pemrograman. Untuk mengetahui kebenaran *logic* dan keefisienan penulisan sebuah *source code* yang dibuat maka dibutuhkan penilaian. Penilaian ini dilakukan oleh instruktur atau dosen (pembuat soal) yang akan memberikan nilai dari sebuah *source code* yang telah dibuat oleh mahasiswa. Penilaian biasa mengoreksi benar/salahnya keluaran yang diberikan, kesalahan sintak dan keefisienaan penulisan *source code*. Penilaian ini merujuk dari masalah yang telah didefinisikan. Dalam melakukan penilaian biasanya didapatkan kendala baik sebelum melakukan penilaian maupun saat melakukan penilaian. Biasanya penilaian terhadap sebuah *source code* tidak dapat dilakukan dengan singkat(setelah pengumpulan *source code* langsung dinilai). Apalagi jika terdapat banyak variasi penulisan *source code*. Dapat diambil contoh dalam lingkup jurusan Teknik

Informatika di POLBAN. Penilaian beberapa dosen mata kuliah DDP(Dasar-Dasar Pemrograman) terhadap sebuah *source code* yang telah dibuat mahasiswa tidak dapat segera dilakukan ketika mahasiswa telah menyelesaikan pembuatan *source code* nya. Berdasarkan hasil wawancara kami terhadap salah satu dosen DDP di jurusan Teknik Komputer dan Informatika POLBAN dikatakan bahwa mahasiswa mendapatkan hasil tugas mereka rata-rata lebih dari 1 minggu setelah pengumpulan tugas *source code*nya bahkan terkadang tidak ada penilaian yang dilakukan dosen terhadap *source code* tersebut sehingga mahasiswa tidak dapat mengetahui sejauh mana nilai kebenaran dari *source code* yang telah dibuatnya. Berdasarkan pengalaman dari salah satu dosen JTK POLBAN yang mengajar mata kuliah DDP ada beberapa hal yang membuat proses pengembalian hasil pemeriksaan menjadi lama. Adapun hal-hal yang menjadi penyebab lamanya waktu pengembalian hasil penilaian diantaranya :

1. Kesibukan dari seorang dosen (pemeriksa)
2. Jadwal dosen yang telah ditentukan waktu memeriksa tiap tugas
3. Pemeriksaan berkali-kali yang dilakukan dosen agar menghindari kesalahan dalam memeriksa
4. Menilai terhadap *coding* koefisiennya

Selain itu juga, penilaian yang dilakukan dosen terkadang hanya berbentuk angka dari *range* 0-100. Terkadang tidak ada penjelasan mengenai hasil penilaian dan bentuk *source code* yang sempurna (dengan *range* nilai 100). Seperti yang di jelaskan oleh salah satu dosen mata kuliah DDP di JTK POLBAN bahwa dalam melakukan penilaian terdapat kriteria penilaian yang harus ditentukan . Berikut merupakan kriteria penilaian yang dilihat dosen DDP dalam penilaian source code.

a. Test Case
b. Kesalahan sintak
c. Komentar atau deskripsi program
d. Kemiripan antar *source code*
e. Kesesuaian proses dengan struktur program yang telah ditentukan

f. Batas waktu pengumpulan *source code*

Automated grade merupakan salah satu alternatif cara menyelesaikan masalah tersebut. Umumnya, aplikasi automated grade menilai 3 aspek dari program/ *source code*. Pertama, menilai *validitas source code* (mengecek benar salah dari hasil eksekusi *source code*). Kedua, menilai kelengkapan dari *source code*. Ketiga, menilai kualitas dan indentasi *source code* ("Teaching Software Testing: Automatic Grading Meets Test-first Coding",Stephen H. Edwards,2003).

1.2.Perumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, rumusan masalah pada sistem ini adalah :

1. Banyaknya kriteria penilaian *source code* sehingga dibutuhkan suatu *tools* yang dapat membantu dosen dalam memberikan bobot penilaian terhadap kriteria penilaian yang ada
2. Dosen membutuhkan waktu lebih dari 1 minggu untuk mengembalikan hasil pemeriksaan *source code* mahasiswa sehingga dibutuhkan *tools* untuk mempercepat waktu pemeriksaan *source code* berbahasa Java.

1.3.Tujuan dan Manfaat

Berdasarkan rumusan masalah yang telah di uraikan maka tujuan dan manfaat dari aplikasi ini adalah :

1. Aplikasi ini membantu dosen untuk mengefisiensikan waktu penilaian sehingga dapat memberikan *feedback* yang lebih cepat kepada mahasiswa
2. Aplikasi ini mampu membantu dosen untuk memberikan bobot penilaian terhadap kriteria penilaian *source code*.

1.4.Batasan masalah

Adapun batasan masalah yang merujuk dari latar belakang diatas yaitu :

1. Kriteria penilaian yang digunakan adalah *correctness* dan *code coverage*.

2. Penentuan bobot penilaian berupa presentase terhadap kriteria penilaian *correctness* dan *code coverage*

1.5.Lingkup Sistem

Sistem yang akan dikembangkan ditujukan untuk digunakan dosen-dosen dari jurusan di Teknik komputer dan informatika (tahun ajaran tingkat 1) Politeknik Negeri Bandung (POLBAN) dengan domain Dasar-Dasar Pemrograman (DDP) bahasa pemrograman Java dengan cakupan sistem sebagai berikut :

1. Menilai *source code* untuk bahasa pemrograman Java
2. Dapat menampilkan hasil keluaran berupa kesalahan sintak setelah proses pemeriksaan *source code*.
3. Dapat menilai *correctness* program dengan memberikan hasil keluaran nilai berupa angka.
4. Dapat menilai *code coverage* program dengan memberikan hasil keluaran berupa nilai *method, conditional* dan *statement*.
5. Dapat menampilkan hasil keluaran berupa nilai total (*range* dari 0-100), Nilai total ini didapat setelah mengakumulasikan nilai dari tiap kriteria penilaian yang telah ditentukan bobotnya.

1.6.Metodologi

Untuk Metodologi Pengembangan Software, metode yang digunakan adalah Waterfall. Metode *waterfall* adalah sebuah metode pengembangan perangkat lunak dimana proses pengerjaannya dilakukan secara sekuen. Berikut merupakan langkah-langkah pengerjaan dalam metodologi *waterfall* :

1. Analisis

   Tujuan dari analisis ini yaitu untuk memperoleh informasi tentang proses bisnis dan kebutuhan aplikasi. Hal yang dianalisis dan evaluasi adalah sebagai berikut:

   a. Waktu penilaian yang dilakukan dosen Dasar-Dasar Pemrograman terhadap *source code* yang telah dibuat oleh mahasiswa

b. Adanya kriteria penilaian menyangkut mata kuliah Dasar-Dasar Pemograman
c. Penentuan bobot kriteria penilaian
d. Mengeksplorasi aplikasi sejenis *automatic grading* seperti WebCAT, Marmoset dan Codingbat
e. Kejelasan nilai yang diberikan oleh dosen terhadap *source code* yang dinilai

2. Perancangan

Pada tahapan ini dilakukan perancangan aplikasi *automatic grading* yang disesuaikan dengan kebutuhan yang telah didefinisikan di Bab analisis. Hal-hal yang dilakukan pada tahap perancangan adalah ::

a. Menentukan batasan otomasi
b. Merancang modul-modul yang digunakan yang berasal dari Framework marmoset.
c. Merancang Aliran Data dan Informasi menggunakan pendekatan Object Oriented.
d. Merancang Data menggunakan tools entity relationship, dan skema relasi.
e. Merancang Tampilan Aplikasi menggunakan tools mock-up dari aplikasi yang akan dikembangkan.

3. Coding

Pada tahapan ini dilakukan implementasi yang sesuai dengan rancangan yang telah dibuat. Hal-hal yang digunakan untuk implementasi adalah sebagai berikut:

a. Bahasa Pemrograman Java.
b. *Open source* marmoset
c. OS Linux

4. Testing

Pada tahap ini dilakukan perancangan pengujian yang akan dilakukan untuk aplikasi. Testing yang akan dikerjakan bertujuan untuk menguji proses dan semua requirement yangdibuat.