

Desain Arsitektur Aplikasi Penilaian Sikap Siswa Menggunakan standar *Experience Application Programming Interface (xAPI)*

Aila Gema Safitri^{*1}, Denny Chandra^{*2}

^{*1} Informatika, Universitas Muhammadiyah Bandung, Jalan Palasari 9A, Bandung 40262, Indonesia

^{*2} Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, Bandung 40132, Indonesia

ailagema@umbandung.ac.id, denny.chandra@students.itb.ac.id

Abstract

Nowdays, most of school both public and private has made aspects of student attitudes and skills to be one of the assessment parameters in the report. The teacher evaluates attitude activities by assigning students / parents to record activities in the form of scientific writing, photos and video. The student/parent should reports the activity into a report book of activities by attaching photos, description of time, place and description of activities. However, this raises many obstacles especially in terms of evaluation of learning activities. First, the evaluation process took time longer because it is done manually. Second, the limitations of paper make the documentation of student activities not comprehensive and not realtime. It becomes an opportunity for developing apps to record and analyze all activities. Rapiku (Rapor Afektif dan Psikomotorik Ku) is an mobile app to help teachers and parents monitor and evaluate students activity in a realtime, continuously and accurately. The architecture is designed using Experience Application Programming Interface (xAPI) technology. The App records student activities in the form of photos, videos, hashtags, time and activity descriptions in the form of statements. The statements will be stored in the LRS (Learning Record Store) as the main part of xAPI. The data in the form of statements in the LRS will be processed and analyzed under the learning analytic system. With the standard of xAPI, teacher, school and other educational organizations can aggregate, visualize and analyze educational data mining comprehensively and accurately.

Keywords: xAPI, Tin Can API, learning assessment, learning analytic, educational data mining.

I. PENDAHULUAN

Usia Sekolah Dasar merupakan usia yang tepat untuk diberikan pendidikan dan penanaman karakter. Pembentukan karakter dilakukan melalui pembiasaan kegiatan positif yang dilakukan secara terus menerus sehingga menjadi kebiasaan yang menyenangkan untuk dilakukan. Salah satu komponen penilaian kompetensi inti yang ada pada Kurikulum 2013 adalah penilaian afektif (sikap) dan psikomotorik (keterampilan). Saat ini, hampir di setiap sekolah baik negeri maupun swasta telah menjadikan penilaian sikap dan keterampilan siswa menjadi salah satu parameter penilaian dalam buku rapor. Kegiatan belajar yang dilakukan diluar kelas sebagian besar adalah dalam bentuk aktivitas di rumah ataupun berupa kunjungan ke suatu tempat seperti museum, situs bersejarah, taman dan lainnya.

Umumnya, sekolah masih menggunakan media buku laporan kegiatan dimana siswa/orang tua harus menyertakan bukti fisik kegiatan dalam bentuk foto yang disertai keterangan waktu, tempat dan deskripsi kegiatan. Hal ini memberikan banyak kendala bagi guru dalam melakukan evaluasi karena guru harus memeriksa, merekap dan menilai laporan kegiatan setiap siswa. Dampak yang ditimbulkan adalah proses evaluasi menjadi lama. Kendala lain yang muncul adalah guru

tidak memiliki dokumentasi kegiatan siswa yang lengkap dan *realtime*. Hal ini dikarenakan keterbatasan jumlah foto yang dapat dimasukkan ke dalam buku evaluasi kegiatan.

Arsitektur sistem Rapiku (Rapor Afektif dan Psikomotorik Ku) dikembangkan untuk membantu guru dan orang tua dalam memantau, mengevaluasi dan menilai sikap serta keterampilan siswa atau anak secara *realtime*, kontinyu dan akurat. Dari sisi siswa, pengembangan aplikasi dibuat dengan tujuan memotivasi siswa untuk terus-menerus melakukan kegiatan positif sehingga akhirnya akan tumbuh menjadi suatu kebiasaan atau karakter. Untuk mendukung tujuan tersebut, sistem Rapiku dikembangkan menggunakan teknologi *Experience Application Programming Interface (xAPI)*.

Sistem akan menyimpan rekam jejak kegiatan siswa dalam bentuk foto, video, *hashtags*, waktu dan *statements* (pernyataan) sesuai dengan standar xAPI. Dokumentasi kegiatan akan disimpan di LRS (*Learning Record Store*) yang merupakan komponen utama dari xAPI. Analisa data diambil dari parameter yang direkam oleh LRS dalam bentuk *statements* yang terdiri dari subyek, kata kerja dan objek.

A. *Experience Application Programming Interface (xAPI)*

xAPI disebut juga Tin Can API adalah sebuah spesifikasi perangkat lunak yang berfungsi sebagai perantara/protokol antara konten dan sistem untuk saling berbicara/berinteraksi dalam hal merekam, melacak data dan semua jenis rekam jejak dari pengalaman belajar [6]. Tincan Api/xAPI dapat menangkap data pembelajaran tidak hanya dari *Learning Management System* (LMS), namun dalam bentuk data *array* yang luas baik secara *online* maupun dalam dunia nyata [1].

Standar API sebelumnya yang digunakan dalam industri e-learning adalah *Sharable Content Object Reference Model* (SCORM). SCORM dibuat untuk konten pembelajaran *online* dan *Learning Management System* (LMS) secara bersamaan. Kelemahan dari SCORM adalah hanya terbatas pada komunikasi antara konten pembelajaran *online* dan LMS [6]. Hal ini menyulitkan para praktisi dan vendor industri *e-learning* dalam menilai proses belajar yang sejatinya meliputi banyak aspek dan berlangsung secara informal.

Teori pembelajaran menyatakan bahwa 90% proses belajar terjadi di luar kelas [2]. Dalam pembelajaran *online*, proses belajar sendiri tidak terbatas pada interaksi antara pengguna dengan *Learning Management System* (LMS), namun aktivitas pembelajaran lebih banyak dilakukan diluar LMS seperti interaksi sosial diluar kelas, pengalaman mengikuti seminar, riwayat pekerjaan, kursus *online*, menonton video tutorial, membaca artikel/jurnal, diskusi *peer to peer* dan bentuk pembelajaran informal lainnya yang terjadi secara tidak terstruktur [6]. Sementara pelatihan-pelatihan formal yang didukung oleh SCORM seperti konten pembelajaran *online* hanya menilai 10% dari apa yang benar-benar telah mereka pelajari [6].

Melalui xAPI, semua proses aktifitas belajar dapat direkam secara keseluruhan, ditampilkan dan dianalisa berdasarkan pada pengalaman belajar yang sebenarnya. Saat ini, xAPI telah menjadi standar *de facto* pada beberapa vendor LMS seperti *moodle*, *blackboard*, *sakai* dan sebagainya [3]. Latar belakang munculnya xAPI dikarenakan kebutuhan analisis data pembelajaran yang lebih menyeluruh mencakup banyak aspek dan parameter penilaian.

Berikut beberapa kelebihan dari xAPI/Tin Can API [6]:

1. *Statements freedom* : struktur kalimat menggunakan kata benda, kata kerja dan objek .
2. *History freedom* : Tin Can API dapat membuat beberapa LRS saling berkomunikasi satu sama lain. LRS-LRS tersebut dapat saling membagi

data dan saling menerjemahkan. Peserta didik dapat memiliki “*personal data lockers*” dengan informasi pembelajaran individu yang telah mereka lakukan.

3. *Device freedom : statements* dapat dikirim oleh banyak *device* melalui Tin Can API seperti *mobile phone*, simulasi, games, dsb.
4. *Workflow freedom*: mengecek rekam jejak kegiatan pembelajaran tidak harus dimulai dan berakhir di LMS. Hal ini dapat dimulai dimana saja dan menggunakan alat apapun. Konten tidak terikat pada LMS semata.

Cara kerja xAPI melibatkan tiga komponen utama didalamnya yaitu [4]:

1. *Activity provider* : menghasilkan data dalam format xAPI dan memasukkan data tersebut ke dalam LRS. *Activity provider* adalah sistem dan aplikasi dimana aktivitas belajar terjadi seperti pada konten website maupun portal untuk belajar dan aplikasi belajar lainnya.
2. *Learning Record Store* (LRS): basis data yang melakukan verifikasi apakah input data sesuai dengan spesifikasi xAPI, menyimpan semua data yang diperoleh dari *Activity Provider* atau dari admin sistem yang memiliki akses ke data mentah xAPI untuk analisa.
3. *Activity Consumer* : sama seperti *activity provider* yaitu dalam bentuk sistem dan aplikasi yang memodifikasi *user experience* (UX) berdasarkan data xAPI. Contoh pada antarmuka LMS yang menampilkan tanda “check list” ketika telah menyelesaikan aktivitas belajar, pada *leaderboard* dan gamifikasi berupa pemberian *badge*/lencana.

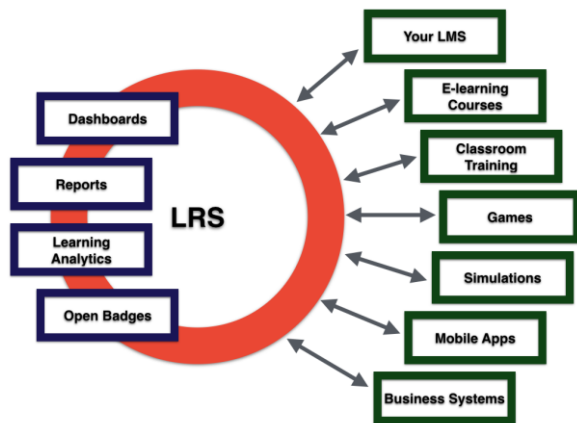
B. *Learning Record Store (LRS)*

Salah satu komponen utama dari xAPI adalah *Learning Record Store* (LRS). LRS adalah database yang menyimpan dan mengambil data yang dihasilkan dalam format xAPI [4]. Setelah nilai variabel direkam, data-data akan disimpan di LRS, kemudian diolah dan dianalisa di bagian analisa pembelajaran [3]. Tugas dari LRS adalah :

1. Melakukan validasi bahwa sistem yang mengirim data adalah sistem yang sudah ter-authorisasi.
2. Memeriksa data yang dikirim oleh xAPI
3. Menyimpan data secara rapih dan terstruktur

4. Menyediakan data ke sistem lain yang ter-authorisasi atau provider ketika diminta.

Dikarenakan xAPI tidak menganjurkan untuk mengambil *learning experience* yang berasal dari LMS, maka detail rekam jejak data dan *statement* dari aktifitas perlu disimpan disuatu tempat dan hal inilah yang menyebabkan kenapa LRS dibutuhkan.



Gambar 1. Arsitektur LRS pada teknologi xAPI yang terhubung ke aplikasi pembelajaran [6].

Pada gambar 1 menunjukkan LRS sebagai sistem yang berdiri sendiri, namun LRS dapat digabungkan ke dalam LMS. Dalam skenario ini, pelaporan dan analisis diserahkan ke LMS dengan menggunakan data dari LRS. LRS yang menjadi bagian dari LMS, masih dapat dihubungkan ke penyedia aktivitas dan LRS lainnya di luar LMS. Seperti pada aplikasi *mobile*, situs pembelajaran dan aktifitas belajar informal lainnya, semuanya dapat diintegrasikan dan digunakan datanya oleh LRS.

Salah satu LRS yang bersifat *open source* dan banyak digunakan yaitu *learning locker* (Learning Locker, 2018). Learning locker dikembangkan oleh HT2 Labs. Beberapa fitur yang terdapat pada *learning locker* yaitu :

1. *Dashboard* yang dapat dikustomisasi, berfungsi untuk memberikan visualisasi hasil analisa data pendidikan dalam bentuk grafik.
2. *Persona*, fitur untuk banyak aktor yang dapat dikelompokkan menjadi satu orang. Tersedia di bagian antar muka dan *statements*. Berfungsi untuk memudahkan *query* dan laporan.
3. *Query Builder*, berfungsi untuk mengelola basis data dari semua nilai query yang ada, sehingga memudahkan pencarian laporan.
4. Peningkatan proses yang lebih cepat pada *endpoint* xAPI.

5. Dapat meneruskan *statements* ke LRS lainnya, atau ke service diluar sistem dengan menggunakan sebuah *webhook*.
6. *Multi-tenant*, multi-LRS. Data telah direstrukturisasi ke dalam organisasi dan penyimpanan, sehingga memungkinkan kontrol dan laporan data yang lebih baik di beberapa LRS didalam lingkungan multi-tenant.
7. Instalasi otomatis. Skrip instalasi lintas platform tersedia untuk mengotomatisasi instalasi *Learning Locker*, dependensi dan konfigurasi server yang diperlukan untuk membangun dan menjalankan server.

C. Statements xAPI

Struktur kalimat yang direkam oleh xAPI disebut *statements*. *Statements* adalah inti spesifikasi xAPI yang memiliki tiga properti utama yang membentuk struktur kalimat yaitu aktor, kata kerja dan objek [3].

Aktor

Aktor mewakili subjek kalimat dapat berupa sebuah *agent* atau grup. *Agent* mewakili seseorang atau sekelompok orang. *Agent* dapat diidentifikasi menjadi empat cara [4] :

1. Mbox yaitu menggunakan email aktor sebagai penanda identitas unik aktor.

```
{
  "objectType": "Agent",
  "name": "John Smith",
  "mbox": "mailto:johnsmith@example.com"
}
```

Gambar 2. Format JSON untuk identifikasi aktor dengan mbox

2. Mbox_sha1sum yaitu menggunakan email aktor yang telah dienkripsi.

```
{
  "objectType": "Agent",
  "name": "John Smith",
  "mbox_sha1sum": "4445904ac65039ef7a91506207f19162ac4dea73"
}
```

Gambar 3. Format JSON untuk identifikasi aktor dengan metode mbox_sha1sum

3. Openid yaitu menggunakan sebuah URL atau link.

```
{
  "objectType": "Agent",
  "name": "John Smith",
  "openid": "http://www.example.com/johnsmith"
}
```

Gambar 4. Format JSON untuk identifikasi aktor dengan metode URL

4. Tipe akun yaitu identifier unik dari aktor beserta sebuah *homepage*.

```
{
  "objectType": "Agent",
  "name": "John Smith",
  "account": {
    "name": "123",
    "homePage": "http://www.example.com/users/"
  }
}
```

Gambar 5. Format JSON untuk identifikasi aktor dengan metode tipe akun.

Kata Kerja

Kata kerja menunjukkan sebuah aksi yang telah dilakukan [5]. Daftar kata kerja yang dipakai oleh xAPI dibuat oleh ADL (*Advanced Distributed Learning*) [8] dan dapat diperluas jika diperlukan. Kata kerja terdiri dari dua properti yaitu :

1. ID, sebagai identitas unik dari kata kerja dan artinya. Pada kondisi tertentu sebuah kata kerja dapat memiliki banyak arti, sehingga untuk membedakan deskripsi kata kerja tersebut digunakan IRI (*Internationalized Resource Identifier*). IRI merupakan *identifier* unik untuk sebuah *resource* pada jaringan internet yang mendukung karakter internasional. IRI adalah bentuk general dari URI (*Uniform Resource Identifier*). Perbedaannya URI hanya mendukung standar encoding ASCII. Contoh dari URI yaitu URL dan no ISBN.
2. *Display*, yang menampilkan kata kerja kedalam format yang dapat dibaca, dapat dipetakan kedalam banyak bahasa.

Objek

Objek adalah apa yang menjadi pengalaman aktor [5]. Elemen objek xAPI terdiri dari :

1. *Activity*

Objek yang di buat sebagai *activity* harus memiliki ID sebagai sebuah penanda unik, tipe objek, dan definisi. Definisi memiliki beberapa properti yaitu

- nama,
- deskripsi,
- tipe (bisa berupa sebuah video, *assessment*, simulasi dan lainnya),
- *more Info* berupa sebuah link mengenai penjelasan lebih detil mengenai *activity*,

- *extensions* yang digunakan untuk identifikasi *activity* lebih detil seperti dimana *activity* terjadi melalui koordinat *longitude* dan *latitude*.
- *Properti interaction* seperti data pada SCORM.

2. *Agent/group*.

Contoh objek dalam bentuk *agent* "John Smith 'followed' Ben". Ben disini sebagai objek berupa *agent*.

3. *Sub-statements* yang merupakan anak kalimat dari kalimat utama.

4. *Statements Ref* yang merupakan referensi dari *statement* sebelumnya. Biasanya dipakai jika ingin membatalkan *statements* awal, maka digunakan statemen ref untuk merujuk ke *statement* sebelumnya.

Pernyataan xAPI diurai dengan menggunakan format JSON - Notasi Objek *JavaScript*. Berikut contoh format statemen xAPI :

```
{
  "actor": {
    "objectType": "Agent",
    "name": "John Smith",
    "account": {
      "name": "123",
      "homePage": "http://www.example.com/users/"
    }
  },
  "verb": {
    "id": "http://adlnet.gov/expapi/verbs/completed",
    "display": {
      "en-GB": "completed"
    }
  },
  "object": {
    "objectType": "Activity",
    "id": "http://www.example.com/activities/1"
  }
}
```

Gambar 6. Contoh Format Statements xAPI

Pada gambar 6, keterangan aktor John Smith sebagai subjek memiliki id unik 123 di <http://www.example.com/users/>. Sementara kata kerja "completed" memiliki definisi yang dapat ditemukan di <http://adlnet.gov/expapi/verbs/completed>. Objek "activity" dapat ditemukan di <http://www.example.com/activities/1>. Tujuan dibuat alamat ini sebagai penanda unik dari kata kerja "completed" dan objek "activity" yang digunakan didalam *statements* "John Smith Completed Activity".

II. METODE

Arsitektur Rapiku dirancang menggunakan teknologi *Experience Application Programming Interface (xAPI)*. Komponen utama dari xAPI yaitu LRS (*Learning Record Store*) yang merupakan sebuah kelas dari sistem xAPI, berfungsi untuk menyimpan semua rekam jejak data hasil aktivitas belajar secara menyeluruh, mencakup banyak aspek dan parameter penilaian. LRS menyimpan data tentang pengalaman belajar dari berbagai sumber, yang dikenal sebagai *Activity Providers*. LRS menggunakan data ini di *dashboard*, laporan, analisis pembelajaran dan bahkan dapat memberi penghargaan kepada siswa. Data pada LRS juga dapat digunakan oleh sistem lain yang telah terintegrasi oleh aplikasi Rapiku.

Tools yang digunakan untuk merancang dan mengujicoba *statement xAPI* terdiri dari *learning locker* dan *postman*.

1. Learning Locker

Learning locker adalah salah satu LRS xAPI yang tersedia secara *open source*. Arsitektur *learning locker* terdiri dari antarmuka *browser*, antarmuka HTTP, layanan xAPI dan *worker* [9].

- Antarmuka *browser* ditulis dalam *JavaScript*, menggunakan *react* untuk membuat tampilan dan *redux* untuk mengelola *state* pada *react*. Antarmuka *browser* menggunakan antarmuka HTTP untuk mengambil dan mengubah model di *Learning Locker*. Semua model secara otomatis disimpan dalam 3 detik setelah diubah.
- Antarmuka HTTP ditulis dalam *JavaScript*, menggunakan *express*, *restify*, *mongo*, dan *mongoose*. *Express* digunakan untuk menyediakan rute HTTP, *restify* digunakan di atas *express* untuk menyediakan rute RESTful untuk masing-masing model di *Learning Locker*. *Mongoose* digunakan di atas *Mongo* untuk mengelola model dalam basis data *mongo* yang digunakan oleh *learning locker*.
- Workers* juga ditulis dalam *JavaScript*. Selain itu menggunakan *Redis* dan secara opsional *SQS* melalui antrian driver. *Workers* memanfaatkan antrian untuk memproses pekerjaan yang berjalan lama. *Workers* yang *multiple instance* dapat digunakan dalam *cluster* untuk memproses antrian secara paralel di banyak mesin dan prosesor. Fungsi lain dari *workers* untuk mengelola *cache* pembangun kueri, mengekstraksi persona dari *statement*, mengeksport laporan melalui CSV, mengimpor persona melalui CSV.

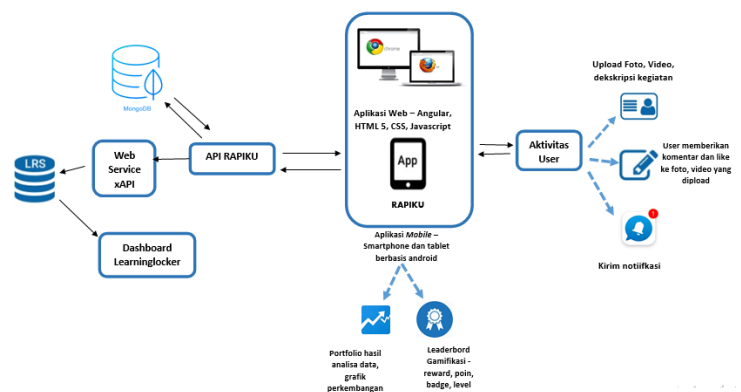
- Postman*, digunakan untuk menguji *statement API* apakah sudah terkirim ke *learning locker* atau masih ada error.

Skenario ujicoba dilakukan pada dua komputer dengan keterangan komputer A sebagai server lokal yang telah diinstall *learning locker* dan komputer B sebagai klien yang bertugas mengirimkan *statement xAPI* melalui *postman*. Jika format *statement xAPI* berhasil dikirim dari komputer B maka akan muncul di *dashboard learning locker* pada komputer A dalam bentuk kalimat.

III. HASIL DAN PEMBAHASAN

a. Ujicoba Statement xAPI

Pada desain arsitektur Rapiku, implementasi LRS yang digunakan adalah *Learning Locker* [7]. Sementara *tool* yang digunakan untuk menguji *statement API* apakah sudah terkirim ke *learning locker* yaitu *postman*. Berikut gambaran arsitektur sistem Rapiku yang diintegrasikan ke LRS *Learning Locker*:



Gambar 7. Arsitektur Rapiku integrasi ke LRS

Berikut penjelasan arsitektur sistem:

Ketika siswa melakukan aktivitas belajar diluar kelas atau dilapangan, siswa atau orang tua akan mengunggah dokumentasi kegiatan dalam bentuk foto atau video menggunakan aplikasi *mobile* Rapiku. Untuk menyimpan data-data aktivitas tersebut, dibuatlah API Rapiku yang akan menjadi perantara untuk menyimpan data ke basis data. Data-data dalam format jpg, png atau jpeg untuk foto, dan mp4 untuk video disimpan ke basis data *monggo* beserta keterangan waktu (jam tanggal dan hari). Sementara, format *statement xAPI* disimpan di LRS *learning locker* melalui *web service xAPI* yang dikirimkan dari API Rapiku.

Data-data kegiatan disimpan dalam dua jenis basis data yaitu *monggo Db* dan LRS *learning locker*.

Pada LRS *learning locker*, terdapat *dashboard* yang dapat menampilkan hasil analisa data dalam bentuk grafik. *Statement* yang telah dikirim sebelumnya dalam format JSON diubah menjadi bentuk kalimat yang dapat dipahami. Berikut contoh tampilan *statement JSON* yang telah berubah menjadi kalimat pada *learning locker*.



Gambar 8. Tampilan *Statement JSON* yang telah diubah menjadi bentuk kalimat pada *learning locker*

```
{
  "actor": {
    "objectType": "Agent",
    "name": "Ihsan",
    "mbox": "ihsanbm@gmail.com"
  },
  "verb": {
    "id": "http://adlnet.gov/expapi/verbs/completed",
    "display": {
      "id-GB": "completed"
    }
  },
  "object": {
    "objectType": "Activity",
    "id": "http://www.example.com/activities/1",
    "definition": {
      "name": {
        "id-GB": "completed"
      },
      "description": {
        "id-GB": "completed"
      },
      "type": "http://adlnet.gov/expapi/activities/question",
      "moreInfo": "http://www.example.com/activities/1/more",
      "extensions": {
        "http://www.example.com/ext": true
      }
    }
  }
}
```

Penjelasan mengenai format JSON diatas sebagai berikut :

1. Pada *statements*, *actor* Ihsan adalah tipe *agent* yang diidentifikasi secara unik menggunakan properti *mbox* yang merupakan email dari pengguna.

2. Format verb terdiri dari dua properti yaitu *id* dan *display*. *Id* sebagai identifikasi unik dari kata kerja yang digunakan. Properti *id* menjadi pembeda arti dari suatu kata kerja, dimana beberapa kata kerja ada yang memiliki bunyi dan tulisan yang sama namun memiliki arti atau kontek yang berbeda. *Value* properti *Id* berupa link URL yang bersifat unik. Properti *display* menampilkan *id verb* menjadi format yang dapat dibaca oleh pengguna dalam bentuk teks dengan standar bahasa Inggris (en). Pada contoh, verb yang digunakan adalah *completed* dimana verb ini diambil dari link <http://adlnet.gov/expapi/verbs/completed> yaitu semacam kamus berisi standar kata kerja dan jenis aktivitas yang sering digunakan di dalam *statement xAPI*.
3. Jenis objek yang digunakan pada contoh adalah *activity*. Objek dapat diperluas dengan memberikan definisi yang berasal dari properti *activity*. Pada contoh, properti objek yang digunakan yaitu *id* dengan definisi *activity "completed question"*.

b. Desain Purwarupa Sistem

Berikut purwarupa dari aplikasi Rapiku yang telah dirancang.



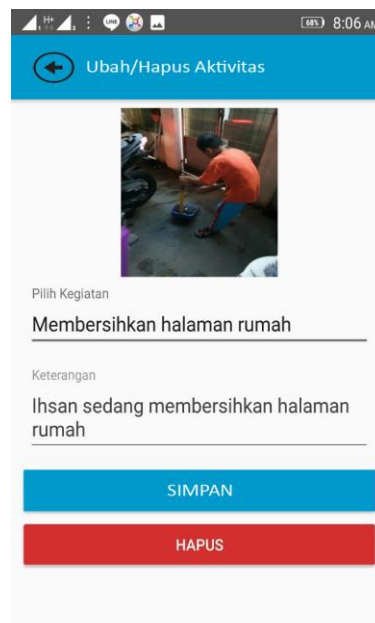
Gambar 9. Leaderboard Aplikasi Mobile

Leaderboard berfungsi untuk menampilkan hasil analisa aktivitas siswa berdasarkan kriteria aspek sikap yang akan ditampilkan yaitu sopan santun, disiplin, tanggung jawab, jujur dan mandiri. Pada gambar 9, data yang ditampilkan adalah enam besar siswa yang memiliki aspek disiplin paling baik. Ranking *leaderboard* diurut berdasarkan jumlah foto terbanyak yang diunggah berdasarkan kategori disiplin. Parameter yang direkam selain jumlah foto yaitu waktu (tanggal, jam), kriteria kegiatan.



Gambar 10. Laporan Sikap Disiplin

Gambar 10 menunjukkan rancangan laporan akhir dari analisa kegiatan sikap disiplin siswa. Analisa data dibuat setiap bulan selama satu semester. Selain itu, nilai yang muncul adalah hasil pengolahan data jumlah foto dikalikan dengan poin yang diberikan untuk kategori disiplin.



Gambar 11. Upload Foto Kegiatan di Luar Kelas

Pada gambar 11, siswa melakukan kegiatan di rumah yang menjadi tugas dalam penilaian sikap. Orang tua mengunggah foto kegiatan disertai keterangan deskripsi kegiatan.



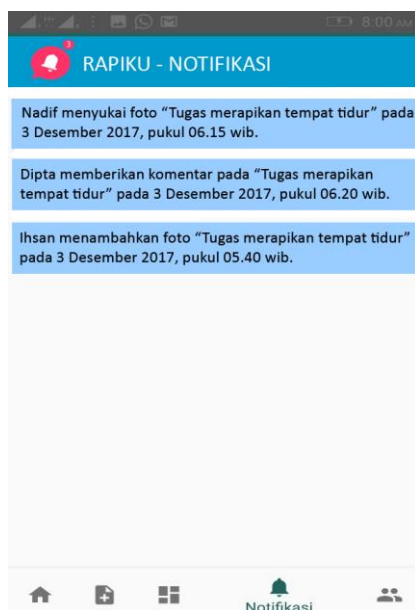
Gambar 12. Rekam Jejak Kegiatan di Luar Kelas

Rekam jejak kegiatan siswa di luar kelas dapat dilihat pada *timeline* masing-masing siswa. Guru dan orang tua dapat memantau aktivitas siswa berdasarkan kategori kegiatan yang dilakukan.



Gambar 13. Halaman Profile Log Kegiatan

Gambar 13 menunjukkan purwarupa tampilan *profile* siswa yang disertai dengan album kegiatan secara keseluruhan.



Gambar 14. Bentuk Statements xAPI sebagai Log Kegiatan Siswa

Gambar 14 adalah desain halaman notifikasi yang muncul ketika ada aktivitas seperti unggah foto, *like* dan komentar. Bentuk kalimat yang muncul pada notifikasi adalah sama seperti format *statement* xAPI yang terdiri dari komponen aktor, kata kerja dan objek.

IV. KESIMPULAN

Perancangan arsitektur sistem Rapiku telah dilakukan dalam dua tahap. Pertama, merancang API dari sistem dalam format *statement* xAPI dengan menggunakan *tools learning locker*. Pada tahap ini penulis telah menentukan variabel hasil kegiatan pembelajaran yang direkam dan menjadi format standar *statement* xAPI yang direkam di LRS. Variabel-variabel tersebut adalah nama (aktor), kata kerja, objek, keterangan waktu (tanggal, jam), jenis kegiatan. Variabel-variabel tersebut akan dianalisa dan menjadi hasil evaluasi kegiatan afektif siswa. Kedua, merancang antarmuka aplikasi *mobile* berdasarkan fungsi aplikasi dan kebutuhan pengguna. Pada tahap selanjutnya akan dikembangkan versi 1 dari aplikasi *mobile android* yang akan siap diujicoba, penambahan variabel lokasi kegiatan yang diambil dari map berdasarkan posisi *latitude* dan *longitude*.

REFERENSI

- [1] A. Borra and C. T. Esguerra, "Game for Learning : An Application of Tin Can API and Learning Record Store (LRS)," vol. 4, 2016.
- [2] D. A. Blackman *et al.*, "THE 70 : 20 : 10 MODEL FOR LEARNING AND DEVELOPMENT : AN EFFECTIVE MODEL FOR CAPABILITY DEVELOPMENT ? bm iss ion ac ce p te d fo r t he Ac ad f M ag t A Authors nn ua l M ee The 70 : 20 : 10 model for learning and development : an effective model for capability development ? tin," no. August, 2016.
- [3] J. C. Vidal, T. Rabelo, and M. Lama, "Semantic description of the experience API specification," *Proc. - IEEE 15th Int. Conf. Adv. Learn. Technol. Adv. Technol. Support. Open Access to Form. Informal Learn. ICALT 2015*, pp. 268–269, 2015.
- [4] T. H. E. P. Masters and G. To, "Guide To," pp. 1–16.
- [5] Mario Manso Vázquez, Manuel Caeiro Rodríguez, Martín Llamas Nistal. *Development of a xAPI Application Profile for Self Regulated Learning*. IEEE Global Engineering Education Conference (EDUCON). 2015
- [6] <https://experienceapi.com/overview/> diakses tanggal 19 Januari 2018.
- [7] <https://learninglocker.net/>. Diakses tanggal 23 Februari 2018.
- [8] <https://www.adlnet.gov/> . Diakses tanggal 1 April 2018.
- [9] <http://docs.learninglocker.net/overview-architecture/> .Diakses tanggal 1 April 2018.

