Jurnal Strategi
Volume 2 Nomor 1 Mei 2020

Layanan Web Kalender Fakultas Teknologi Informasi Universitas Kristen Maranatha dengan Constraint Satisfaction Problem

Jimmy Purnawan[#1], Hapnes Toba[*2]

[#]*Program Studi S1 Teknik Informatika, Universitas Kristen Maranatha*
Jl. Prof. drg. Suria Sumantri, MPH no 65, Bandung

[1]jimmypurnawan7@gmail.com

[2]hapnestoba@it.maranatha.edu

Abstract — ***Faculty of Information Technology Maranatha Christian University organizes many activities on a regular basis and incidental. One of the major problems is how to avoid conflicting schedules. This research proposes a web calendar application which able to display available time schedules among the participants of an event. The development of the web calendar is based on flexibility by using API. The implementation shows that the calendar system is able to recommend time availabilities of a conflicting timetable by generating minimum remaining values. Our evaluation results show that the web-based calendar application was successfully implemented and interact properly with Google Calendar.***

Keywords— ***API ,Back-End Web, Google Calendar, CSP, MySQL, PHP***

I. PENDAHULUAN

Fakultas Teknologi Informasi Universitas Kristen Maranatha (FIT UKM) memiliki banyak kegiatan dan event yang diselanggarakan setiap harinya, baik acara yang diadakan oleh unit kegiatan, program studi, fakultas, universitas maupun dari pihak luar. Terdapat berbagai kendala yang menyebabkan FIT UKM kesulitan dalam pengaturan kegiatan sehingga tidak terkoordinir dengan baik, misalnya terkait dengan bentrok penjadwalan pada saat sebuah event direncanakan.

Seringkali terjadi pula hambatan komunikasi antara pembuat suatu event dan peserta acara. Banyak sivitas akademika yang ingin mengikuti sebuah acara tetapi tidak mengetahui jadwal acaranya dengan jelas sehingga seringkali terlewat. Masalah lain yang timbul adalah kesulitan dalam menentukan waktu untuk pertemuan antar beberapa orang dikarenakan jadwal tiap orang yang berbeda-beda. Sebuah solusi yang diusulkan melalui riset ini adalah pembuatan aplikasi web kalender untuk dapat menampilkan waktu yang tersedia bagi sekumpulan orang pada saat merencanakan sebuah event.

Di masa yang akan datang tidak dapat diprediksi platform yang sedang marak digunakan, maka dari itu dibutuhkan sebuah web API yang dapat dengan mudah berpindah platform jika diperlukan. Untuk mengatasi hal ini diperlukan web API yang bisa menangani semua kegiatan dalam sebuah kalender yang berisikan jadwal setiap kegiatan yang ada dari setiap program studi, fakultas, universitas maupun pihak luar. Dengan penggunaan web API ini diharapkan dapat membuat aplikasi ini menjadi fleksibel dalam perkembangannya. Aplikasi dirancang untuk dapat mencari waktu kosong dari beberapa orang yang akan disertakan dalam sebuah acara. Diharapkan dengan dibuatnya web API ini dapat mengatasi permasalahan yang ada, khususnya pada bagian back-end (sisi server).

II. KAJIAN LITERATUR

A. Back-End

Back-end adalah bagian belakang dari sebuah website yang tidak terlihat oleh pengguna, yang berisi kumpulan algoritma untuk menjalankan fungsi-fungsi pada sebuah website [2]. Bagian ini pada umumnya terhubung dengan database dan juga Application Programing Interface (API).

Setiap back-end dari situs web terdiri dari tiga bagian: server, database, dan aplikasi. Seorang pemrogram back-end bertugas menulis kode yang memungkinkan ketiga komponen ini berinteraksi dan bekerja sama untuk melakukan fungsi dan menyampaikan informasi yang dapat digunakan ataupun dilihat oleh pengguna.

Bahasa pemrograman yang biasa digunakan pada back-end untuk pembuatan aplikasi web adalah PHP, Java dan Python. Untuk sistem database terdapat misalnya Microsoft SQL Server, MySQL, dan Oracle.

B. PHP

PHP adalah suatu bahasa pemrograman web yang disisipkan ke dalam bahasa HyperText Markup Language (HTML) [3]. Tujuan dari bahasa pemrograman ini adalah untuk membuat web berinteraksi secara dinamis dan cepat. PHP digunakan pada program server-side atau back-end yang tidak terlihat pada sisi klien. PHP juga membuat pengaksesan database oleh web menjadi mudah.

C. MySQL

MySQL adalah salah satu perangkat lunak Relational Database Management System (RDBMS) yang multiuser dan multithread dan menggunakan bahasa Structured Query Languange (SQL) [4]. MySQL tersedia secara gratis atau open source dengan lisensi GNU General Public License (GPL). Meskipun gratis, MySQL dapat menangani data yang cukup besar dan perfomanya pun dapat diandalkan.

D. API

API merupakan sebuah pendekatan yang membuat data dari website dapat dicerna oleh komputer [5]. Melalui API, komputer dapat melihat ataupun merubah data lintas sistem, dan tidak bergantung pada platform yang digunakan. Ketika dua buah sistem (misalnya berbasis website, desktop, ataupun smartphone) terhubung oleh API, maka sistem-sistem tersebut dapat diintegrasikan. Saat sebuah integrasi terjadi, ada dua sisi yang perlu diperhatikan. Yang pertama adalah sisi server. Ini adalah sisi dimana API berada. Sisi server membantu untuk mengingatkan bahwa API ini adalah hanya sebuah program yang berjalan pada server.

Sisi yang lain adalah klien. Sisi ini adalah program terpisah yang mengetahui data apa saja yang tersedia melalui API dan dapat memanipulasi data, biasanya atas permintaan dari pengguna. Sebuah contoh yang baik adalah aplikasi smartphone yang tersinkronisasi dengan website. Ketika tombol refresh aplikasi ditekan, sisi klien berbicara ke server melalui API dan mengambil info terbaru. Prinsip yang sama berlaku untuk situs web yang terintegrasi. Ketika salah satu situs menarik di data dari yang lain, situs yang menyediakan data bertindak sebagai server, dan situs mengambil data adalah klien.

E. Google Calendar

Google Calendar adalah sebuah aplikasi web berbasis API untuk mengelola waktu yang dibuat oleh Google. Untuk penggunaannya tidak memerlukan biaya atau gratis. Dengan Google Calendar API, pengguna dapat mengatur jadwal dan membagikan jadwal mereka pada pengguna lainnya [6]. Terdapat pula reminder yang dapat dikirim melalui ponsel yang terintegrasi dengan akun Google pengguna untuk mengingatkan jadwal dari pengguna. Dibutuhkan sebuah akun Google untuk menggunakan API ini.

F. Constraint Satisfaction Problem

Constraint Satisfaction Problem adalah sebuah pendekatan untuk menyelesaikan suatu masalah dengan tujuan menemukan keadaan atau objek yang memenuhi sejumlah persyaratan atau kriteria yang ditentukan [7].

Komponen-komponen yang terdapat pada CSP adalah :

1. Variabel
   Variabel merupakan penampung dapat diisi berbagai nilai,
2. Domain
   Merupakan kumpulan nilai legal yang dapat diisi ke variabel.
3. Constraint / Batasan
   Constraint yang merupakan suatu aturan atau batasan yang ditentukan untuk mengatur nilai yang dapat diisikan ke dalam sebuah variabel, atau kombinasi variabel.
4. Assignment
   Pemberian nilai pada suatu variabel.
5. Solusi
   Pemberian nilai-nilai pada setiap variabel yang memenuhi semua constraint yang telah ditetapkan untuk persoalan, sehingga nilai-nilai variabel tersebut merupakan solusi dari persoalan.

Salah satu pendekatan untuk mencari solusi dalam CSP adalah dengan menemukan suatu nilai yang paling mungkin digunakan untuk mengisi sebuah constraint dikenal sebagai minimum remaining values (MRV). Melalui pendekatan ini, jika terdapat beberapa nilai dalam sebuah domain yang masih mungkin untuk diisikan pada suatu variabel, maka akan dicari kandidat solusi dengan menghitung jumlah constraint minimal yang tersisa terhadap keseluruhan variabel.

G. Penelitian Sejenis

Dalam Tabel 1 disampaikan beberapa penelitian sejenis dengan penerapan API dari Google Calendar. Kelebihan utama yang ditawarkan dalam penelitian ini adalah adanya mekanisme pengecekan jadwal kosong dari berbagai akun yang dilibatkan pada saat pembuatan jadwal.

TABEL I

PENELITIAN SEJENIS

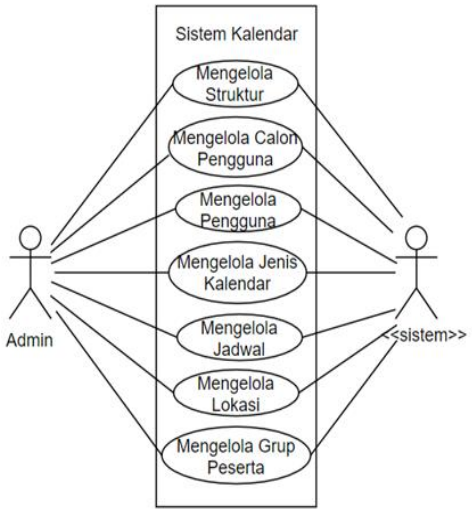
Parameter/ Penulis	**Judul**	**Teknologi/ Metode**	**Bahasa Pemrograman**	**Platfrom / Hasil**
Imelda Sartika Wulan Rosari dan Christine Dewi (2016)	Perancangan Aplikasi Bimbingan Tugas Akhir Memanfaatkan Google Calendar	*CodeIgniter, Rapid Application Development*	PHP	*Website*
Gunawan Ariyanto (2016)	Aplikasi kalender untuk mengelola jadwal kegiatan di UMS menggunakan Google *Application* API	*Django Framework, Google Application API*	Python	*Website*

III. ANALISIS DAN RANCANGAN SISTEM

A. Rancangan Pemanfaatan Web Kalender

Use case aplikasi ditinjau dari sisi back-end memiliki dua aktor utama, yaitu: admin dan sistem. Dalam aplikasi ini, aktor sistem dan juga admin dapat melakukan aksi yang sama, yaitu: mengelola

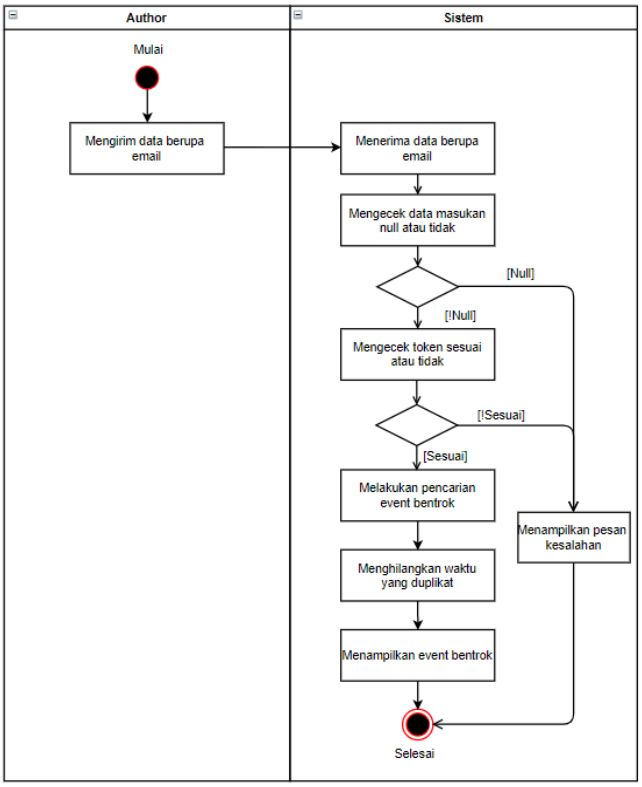
struktur, mengelola calon pengguna, mengelola pengguna, mengelola jenis kalender, mengelola jadwal, mengelola lokasi dan mengelola grup peserta. Use case tersebut dapat dilihat pada Gambar 1.



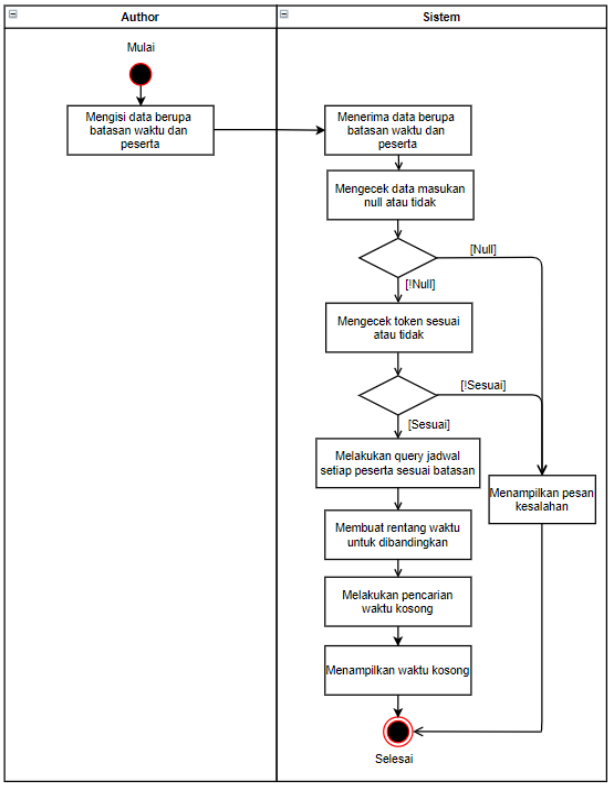
Gambar 1. *Use Case* Web Kalender

B. Proses Menampilkan Jadwal Bentrok

Pada Gambar 2 disampaikan proses untuk menampilkan jadwal bentrok. Pada saat seorang *author* membuka halaman jadwal bentrok maka *author* akan mengirim data berupa *email* kepada sistem dan sistem akan menerimanya serta melakukan pengecekan apakah pesan *valid* atau tidak, bila tidak *valid* (kondisi *null*), maka sistem akan menampilkan pesan kesalahan dan jika *valid* (kondisi *!null*) maka akan dilanjutkan dengan proses pengecekan *token*. Jika *token* yang dimasukkan benar, maka sistem akan melakukan *query* untuk mencari data jadwal bentrok dalam *database*. Kemudian sistem akan menampilkan jadwal yang bentrok. Sedangkan jika *token* terdeteksi salah, maka sistem akan menampilkan pesan kesalahan.



Gambar 2. *Activity Diagram* Menampilkan Jadwal Bentrok



Gambar 3. *Activity Diagram* Pencarian Waktu Kosong

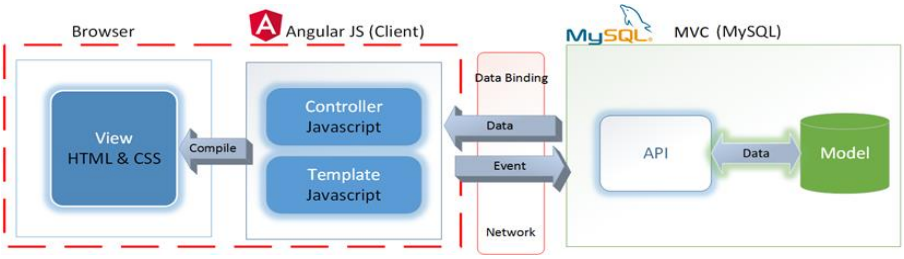
C. Proses Pencarian Waktu Kosong

Pada Gambar 3 terdapat proses pencarian waktu kosong. Dimulai saat *author* mengisi data berupa batasan waktu dan peserta yang terlibat untuk dijadwalkan, kemudian sistem menerima batasan berupa waktu mulai dan selesai. Setelah itu, sistem akan melakukan pengecekan terhadap data yang diterima apakah bernilai tidak *valid* (*null*). Jika *null*, sistem akan menampilkan pesan kesalahan. Jika *valid*, maka sistem akan melakukan pengecekan *token*. Jika *token* yang dimasukkan benar, maka sistem akan melakukan *query* untuk membandingkan jadwal setiap peserta sesuai batasan waktu yang ditentukan. Lalu sistem akan membuat rentang waktu untuk dibandingkan, kemudian dilanjutkan dengan melakukan pencarian waktu kosong dengan menggunakan pendekatan *minimum remaining values*.

IV. IMPLEMENTASI

A. Arsitektur Aplikasi

Aplikasi web yang dikembangkan memiliki arsitektur seperti terlihat pada Gambar 4. Aplikasi ini dikembangkan dalam bahasa pemrograman PHP sebagai back-end dan MySQL sebagai model-nya, dengan konsep model-view-controller (MVC). Sebelum data di ampilkan pada browser, data harus melewati berbagai tahap terlebih dahulu. Dimulai dari perintah atau event yang dikirim oleh controller untuk meminta data, yang diterima oleh API dan dilanjutkan untuk diproses ke bagian datamodel yaitu MySQL. Setelah itu barulah dikirim kembali sesuai permintaan ke bagian controller dengan menggunakan template antarmuka yang diolah terlebih dahulu sebelum ditampilkan pada halaman view.



Gambar 4. Arsitektur Aplikasi

B. Implementasi Fitur Pencarian Jadwal Bentrok

Fitur pencarian jadwal bentrok adalah sebuah fitur untuk menampilkan data jadwal dari seseorang yang terjadi dalam rentang waktu yang sama. Langkah-langkah untuk mendapat jadwal yang bentrok adalah sebagai berikut:

1. Pertama melakukan *query* ke dalam *database* dengan memilih terlebih dahulu semua data jadwal dari pengguna.

```
1.  $query = "SELECT event.*,calendar.* FROM event
2.  JOIN calendar ON calendar.idCalendar=event.Calendar_idCalendar
3.  JOIN participant ON event.idEvent=participant.Event_idEvent
4.  WHERE Email='$email' ORDER BY StartTime DESC";
```

Gambar 5. Query Jadwal Pengguna

2. Dilakukan pengecekan terhadap setiap jadwal apakah bentrok atau tidak satu dengan lainnya. Jika bentrok maka data jadwal akan dimasukkan ke dalam *array* kumpulan jadwal bentrok.

```
$conflict=[];
$notconflict=[];
$n=0;
for($i=0;$i<count($output);$i++){
    $cek=FALSE;
    $ada=TRUE;
    for($j=$i+1;$j<count($output);$j++){
        $s1=$output[$i]['StartTime'];
        $e1=$output[$i]['EndTime'];
        $s2=$output[$j]['StartTime'];
        $e2=$output[$j]['EndTime'];
        $start1 = strtotime($s1);
        $end1 = strtotime($e1);
        $start2 = strtotime($s2);
        $end2 = strtotime($e2);

        if($start2<$end1 && $end2>$start1&&(($start2>=$start1&&
           $end2>=$end1)||($end2<=$end1 && $start2<=$start1 )||
           ($start2>=$start1 &&  $end2<=$end1)))
        {
            array_push($conflict,$output[$j]);
            $cek=TRUE;
        }
        else{
            array_push($notconflict,$output[$j]);
        }
    }
    if($cek){
        array_push($conflict,$output[$i]);
        $ada=FALSE;
        $n++;
        array_push($conflict,$n);
     }
}
```

Gambar 6. Cek Jadwal Bentrok

3. Sebelum dikirim, *array* kumpulan jadwal bentrok akan melakukan pengecekan apabila terdapat elemen yang bernilai sama.

```
function super_unique($array,$key){
    $temp_array = [];
    foreach ($array as &$v) {
        if (!isset($temp_array[$v[$key]]))
        $temp_array[$v[$key]] =& $v;
    }
    $array = array_values($temp_array);
    return $array;
}
```

Gambar 7. Proses Membuang Duplikasi Data Jadwal

C. Implementasi Fitur Pencarian Jadwal Kosong

Fitur pencarian waktu kosong adalah sebuah fitur untuk menampilkan waktu kosong dari rentang waktu yang ditentukan, yaitu dalam ‘satu hari’, dan dilakukan terhadap semua peserta sebuah kegiatan. Proses kerja algoritma pencarian waktu kosong adalah sebagai berikut:

1. Pertama-tama algoritma menerima masukkan berupa rentang waktu dari waktu yang dibutuhkan dan juga para peserta suatu kegiatan serta menyimpan ke dalam variabel.

```
$data = json_decode(file_get_contents("php://input"));
$starttime=$data->StartTime;
$endtime=$data->EndTime;
$listemail=$data->ListEmail;
$priority = $data->Priority;
```

Gambar 8. Terima Masukkan Jadwal

2. Selanjutnya dilakukan *query* data jadwal dari dalam *database* untuk setiap peserta kegiatan yang dipilih dalam rentang waktu tersebut dalam format 'hh-hh'.

```
$query="SELECT CONCAT(DATE_FORMAT(StartTime, '%k.%i' ),
CONCAT('-',DATE_FORMAT(EndTime, '%k.%i' ))) AS event FROM event
JOIN user_has_event ON event.idEvent=user_has_event.Event_idEvent
JOIN user ON user.NIK=user_has_event.User_NIK JOIN structure ON
structure.idStructure=user.Structure_idStructure WHERE StartTime<='$endt'
&&EndTime>='$startt' && ((StartTime>='$startt' &&  EndTime>='$endt')OR
(EndTime<='$endt' && StartTime<='$startt' ) OR (StartTime>='$startt' &&
EndTime<='$endt'))  && Email='$email' && Priority>='$priority'";
```

Gambar 9. Query Jadwal Tiap Peserta

3. Membuat sebuah *array* yang berisikan dari waktu-waktu di antara rentang waktu yang telah ditentukan di awal.

```
$allTimeSlots = [];
for($j=$hs;$j<$he;$j++){
    array_push($allTimeSlots,(strval($j).".00-".strval($j+0.3)."0"));
    array_push($allTimeSlots,(strval($j+0.3)."0-".strval($j+1).".00"));
}
```

Gambar 10. Membuat Rentang Waktu

4. Memecah-mecah waktu dari tiap jadwal menjadi per-satu jam dan menampungnya ke dalam sebuah *array*.

```
function parseSlot($list_of_slots) {
    $result = [];
    $aa=0;
    for($j=0;$j<count($list_of_slots);$j++){
        list($a, $b) = explode('-', $list_of_slots[$j]['event']);
        $start_time=(int)$a;
        $start_timet=(double)$a;
        $end_timet=(double)$b;
        $k = 0;
```

```
10.         while(($start_time + $k) < $end_timet){
11.             if($k==0){
12.                 if(fmod($start_timet,1)>=0.3){
13.                     array_push($result,(strval($start_time+$k+0.3)."0-"
14.                     .strval($start_time+$k+1).".00"));
15.                 }
16.                 else{
17.                     array_push($result,(strval($start_time+$k).".00-"
18.                     .strval($start_time+$k+0.3)."0"));
19.                     array_push($result,(strval($start_time+$k+0.3)."0-"
20.                     .strval($start_time+$k+1).".00"));
21.                 }
22.             }
23.             else if($k==$end_timet-$start_time-1){
24.                 if(fmod($end_timet,1)>=0.3){
25.                     array_push($result,(strval($start_time+$k).".00-"
26.                     .strval($start_time+$k+0.3)."0"));
27.                     array_push($result,(strval($start_time+$k+0.3)."0-"
28.                     .strval($start_time+$k+1).".00"));
29.                     array_push($result,(strval($start_time+$k+1).".00-"
30.                     .strval($start_time+$k+1.3)."0"));
31.                 }
32.                 else{
33.                     array_push($result,(strval($start_time+$k).".00-"
34.                     .strval($start_time+$k+0.3)."0"));
35.                     array_push($result,(strval($start_time+$k+0.3)."0-"
36.                     .strval($start_time+$k+1).".00"));
37.                 }
38.             }
39.             else{
40.                 array_push($result,(strval($start_time+$k).".00-"
41.                 .strval($start_time+$k+0.3)."0"));
42.                 array_push($result,(strval($start_time+$k+0.3)."0-"
43.                 .strval($start_time+$k+1).".00"));
44.             }
45.             $k += 1;
46.         }
47.     }
48.     return $result;
49. }
50.
```

Gambar 11. Pemecahan Waktu Tiap Jadwal

5. Kemudian dibandingkan *array* yang berisi rentang waktu dengan *array* yang berisi pecahan waktu dari setiap jadwal. Jika terdapat elemen dari *array* rentang waktu yang ‘tidak ada’ dalam pecahan waktu setiap jadwalnya, maka elemen tersebut akan ditampung ke dalam sebuah *array* yang berisi kumpulan waktu kosong.

```
$badslots=[];
$commonFreeSlots = [];
for($j=0;$j<count($allTimeSlots);$j++){
    $timeSlotOk = True;
    for($k=0;$k<count($allPeople);$k++){
        $person_free_slots = parseSlot($allPeople[$k]);
        if(in_array($allTimeSlots[$j],$person_free_slots)){
            $timeSlotOk = False;
            break;
            }
        }
    if($timeSlotOk){
        array_push($commonFreeSlots,$allTimeSlots[$j]);
    }
    else{
        array_push($badslots,$allTimeSlots[$j]);
    }
    $timeSlotOk = True;
}
```

Gambar 12. Membandingkan Waktu Jadwal dengan Rentang Waktu

D. Implementasi CSP

Dalam sistem kalender ini digunakan *Constraint Satisfaction Problem* pada fitur pencarian jadwal kosong. Pendekatan yang digunakan disini adalah MRV yang mencari *domain* nilai dengan jumlah *constraints* seminimal mungkin. MRV dapat memilih dan menentukan pilihan dari beberapa kandidat, yang secara heuristik memilih kandidat dengan jumlah ekspansi (*constraints*) paling sedikit.

MRV memprioritaskan variabel yang tidak ditetapkan secara dinamis berdasarkan jumlah nilai yang tersedia dalam domain variabel. Komponen-komponen yang digunakan dalam implementasi adalah :

1. Variabel
   Variabel yang digunakan berupa:
   • $arrayTime, sebagai penampung data jadwal dari setiap peserta.
   • $startt, sebagai penampung batas waktu mulai.
   • $endt, sebagai penampung batas waktu selesai.
   • $email, sebagai penampung nilai email yang dari setiap peserta.
   • $priority, sebagai penampung nilai priority dari struktur yang dimiliki setiap participant.

2. *Constraint* / Batasan
   Constraint yang digunakan berupa:
   • StartTime<='$endt', dengan StartTime pada suatu jadwal lebih kecil sama dengan $endt.
   • EndTime>='$startt', dengan EndTime pada suatu jadwal lebih kecil sama dengan $startt.
   • ((StartTime>='$startt' && EndTime>='$endt') OR
   (EndTime<='$endt' && StartTime<='$startt') OR
   (StartTime>='$startt' && EndTime<='$endt')), dengan StartTime pada suatu jadwal lebih besar sama dengan $startt dan EndTime lebih besar sama dengan $endt atau EndTime lebih kecil sama dengan $endt dan StartTime lebih kecil sama dengan $startt atau StartTime lebih besar sama dengan $startt dan EndTime lebih kecil sama dengan $startt.
   • Email='$email', dengan email pengguna yang login sama dengan $email.
   • Priority>='$priority', dengan prioritas dari setiap peserta dibandingkan dengan prioritas dari pengguna yang sedang login. Prioritas disini bergantung pada tingkat jabatan dalam sebuah organisasi. Misalnya seorang ketua lebih tinggi dibanding wakilnya.

3. Solusi
   Solusi didapat ketika variabel $arrayTime memenuhi batasan yang ada.
   Dalam hal ini diasumsikan bahwa variabel atau nilai pertama yang sesuai dengan *constraint* yang ada adalah nilai yang paling optimal atau dalam hal ini adalah rentang waktu yang paling optimal untuk dimasukkan sebagai hasil penemuan jadwal.

E. Sinkronisasi Google Calendar

Pada sistem ini terdapat fitur untuk melakukan sinkronisasi dengan Google Calendar untuk setiap pengguna yang terlibat dalam penjadwalan. Sinkronisasi terjadi pada saat sistem menarik data jadwal yang dimiliki pengguna di Google Calendar. Hal ini dibatasi dengan 10 jadwal yang belum berlangsung. Setelah data jadwal berhasil ditarik maka akan diproses untuk dicek apakah data jadwal tersebut telah ada pada *database* sistem. Jika belum ada maka data jadwal akan ditambahkan ke dalam sistem, namun jika sudah ada maka data pada Google Calendar akan di-*update* sesuai dengan data jadwal pada *database*. Ini dilakukan karena data yang terdapat dalam aplikasi diberlakukan sebagai master data dan dianggap sebagai data yang paling benar pada aplikasi.

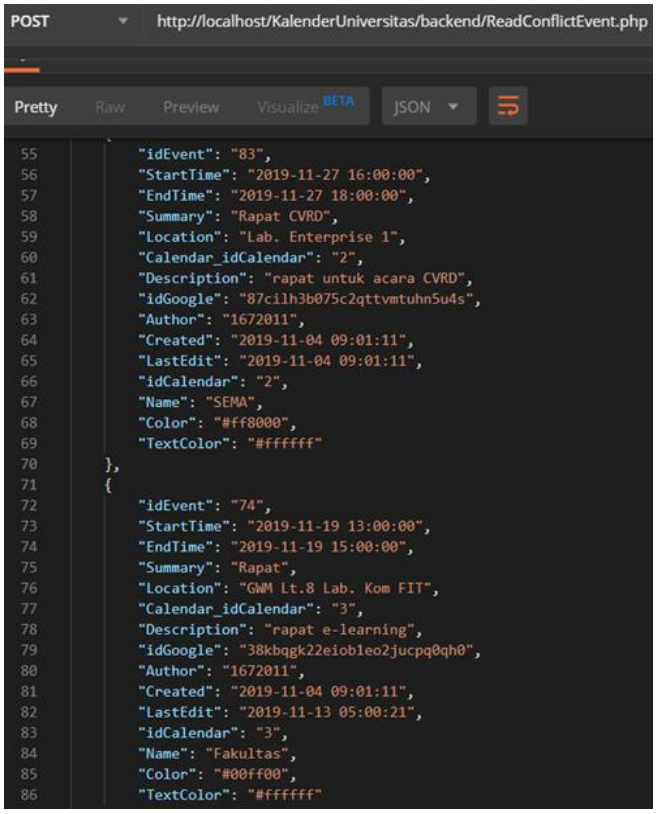
V. PENGUJIAN

A. White Box Testing

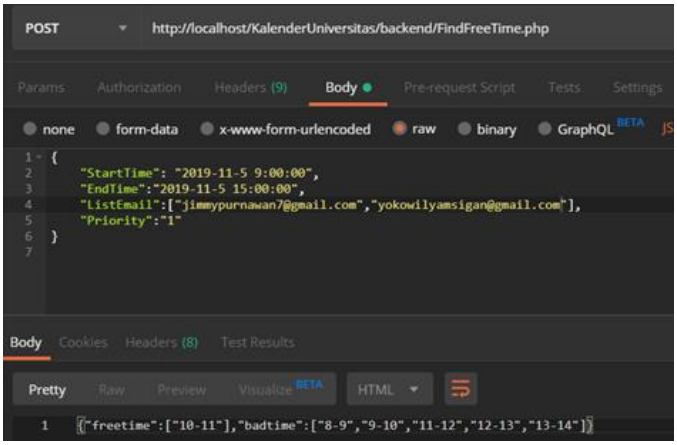
Pengujian web kalender FIT UKM dilakukan dengan menggunakan metode white box testing. Untuk kebutuhan ini digunakan software Postman sebagai alat pengujian untuk API yang dibuat.

1) Fitur Menampilkan Jadwal Bentrok

Pada Gambar 13 diperlihatkan hasil pengujian terhadap fitur untuk menampilkan jadwal bentrok dari pengguna. Setiap data jadwal bentrok akan dikembalikan atau dikirim dalam format json.



Gambar 13. Hasil Pengujian Menampilkan Jadwal Bentrok



Gambar 14. Hasil Pengujian Mencari Waktu Kosong

2) Fitur Mencari Waktu Kosong

Pada Gambar 14 diperlihatkan hasil pengujian terhadap fitur menampilkan waktu kosong dengan empat parameter, yaitu StartTime, EndTime, ListEmail para peserta, dan Priority yang akan dibandingkan dengan prioritas pembuat jadwal. Setiap data waktu kosong akan dikembalikan atau dikirim dalam format JSON.

B. PHP Unit Testing

PHP *Unit Testing* digunakan untuk melakukan pengujian terhadap fungsi-fungsi utama pada aplikasi ini. Pengujian dilakukan untuk dua skenario besar. Pertama, apabila fungsi mengembalikan sebuah nilai *null* atau tidak. Dan kedua adalah disaat fungsi tidak mendapat nilai masukan yang diperlukan.

1) Fitur Menampilkan Jadwal Bentrok

- Hasil Pengujian Test 1

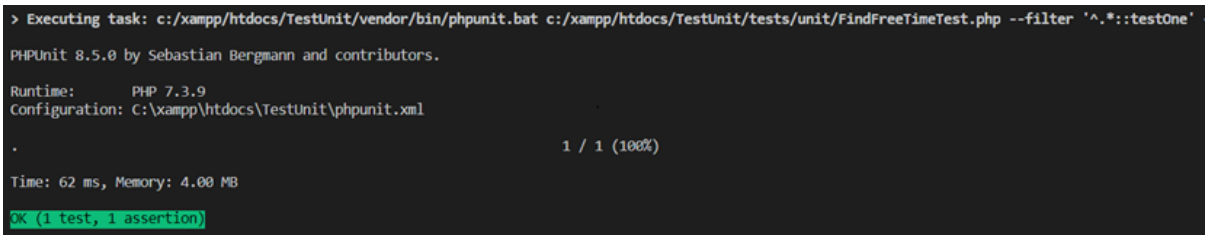
Gambar 15. Hasil Unit Testing Menampilkan Jadwal Bentrok 1

- Hasil Pengujian Test 2

Gambar 16. Hasil Unit Testing Menampilkan Jadwal Bentrok 2

2) Fitur Mencari Waktu Kosong

- Hasil Pengujian Test 1



Gambar 17. Hasil Unit Testing Mencari Waktu Kosong 1

- Hasil Pengujian Test 2

Gambar 17. Hasil Unit Testing Mencari Waktu Kosong 1

VI. KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan, maka dapat diambil kesimpulan bahwa aplikasi web untuk pengelolaan kalender kegiatan FIT UKM telah berhasil dibangun dengan menggunakan API. Selain itu berhasil dibuat pula interaksi dengan Google Calendar API melalui proses sinkronisasi.

Berdasarkan pada hasil pengujian sistem, aplikasi web telah berhasil mengetahui jadwal yang bentrok dan berhasil memberikan saran waktu kosong yang optimal dengan pendekatan Minimum Remaining Values. Dalam hal ini, aplikasi mampu melakukan pencarian jadwal kosong dalam jangka waktu yang ditentukan, yaitu dalam periode satu hari, untuk semua peserta yang diikutsertakan dalam sebuah penjadwalan kegiatan.

DAFTAR PUSTAKA

[1] P. Roger S. Pressman, dalam Software Engineering A Practitioner's Approach Seventh Edition, New York, McGraw-Hill, 2010, p. 930.
[2] Welling, Luke, dan Laura Thomson. PHP and MySQL Web Development. Addison-Wesley, 2017.
[3] Purwanto, yudi. Pemrograman Web dengan PHP, Elex Media Komputindo, Jakarta, 2001.
[4] Anhar. Panduan menguasai PHP & Mysql secara otodidak. Jakarta: Mediakita, 2010.
[5] Cooksey Brian.An Introduction to APIs, 2014.
[6] Google Calendar [Online]. Available: https://www.google.com/calendar/about/ [diakses 17 April 2019].
[7] [7] Sandag, Green A.Constraint Satisfaction Problem.2010.