LAMPIRAN A

userController.php (Menampilkan form pemilihan jadwal kursus dan tampilan konfirmasi jadwal kursus)

```
public function showMeetingForm($student_real_name, $enrollment_id, $meeting_number) {
    // Fetch enrollment and course details
    $enrollment = Enrollment::findOrFail($enrollment_id);
    $course = $enrollment->course;

    // Check if this is the last meeting
    $isLastMeeting = $meeting_number == $course->course_length;

    // Find the enrollment data for this student
    $enrollment = enrollment::with(['schedule', 'coursePayment'])->find($enrollment_id);

    // Manipulate and localize this page to Indonesian
    Carbon::setLocale('id');

    // Get the Open Time of the Admin's
    $openTime = \Carbon\Carbon::parse($enrollment->course->admin->open_hours_for_admin);
    // Get the Close Time of the Admin's
    $closeTime = \Carbon\Carbon::parse($enrollment->course->admin->close_hours_for_admin);
    // Get the course duration of the selected schedule
    $courseDuration = $enrollment->course->course_duration;

    // Generate the course time option until the start time is not more than close time
    while ($openTime->lessThan($closeTime)) {
```

```
            // Generate the end time from adding the open time with course duration
            $endOptionTime = $openTime->copy()->addMinutes($courseDuration);

            // When the end time is passed the close time, end the generation
            if ($endOptionTime->greaterThan($closeTime)) {
                break; // Exit the loop if it exceeds
            }

            // Add the slot to available slots
            $availableSlots[] = [
                'start' => $openTime->format('H:i'),
                'end' => $endOptionTime->format('H:i'),
            ];

            // Create new start time by adding the previous start time with course
duration
            $openTime->addMinutes($courseDuration);
        }

        // Return the form for selecting the date and time for the current meeting
        return view('student-page.user-choose-schedule', [
            'pageName' => "Pilih Jadwal Kursus | ",
            'enrollment' => $enrollment,
            'course' => $course,
            'meeting_number' => $meeting_number,
            'availableSlots' => $availableSlots,
            'isLastMeeting' => $isLastMeeting,
        ]);
    }

    public function showConfirmationPage($student_real_name, $enrollment_id) {
```

```
        // Fetch the schedule data from the session
        $enrollment = Enrollment::findOrFail($enrollment_id);
        $course = $enrollment->course;
        $student_real_name = $enrollment->student_real_name;
        $meetings = [];

        for ($i = 1; $i <= $course->course_length; $i++) {
            $meetings[] = [
                'date' => session()->get("meeting_{$i}_date"),
                'time' => session()->get("meeting_{$i}_time"),
            ];
        }

        Carbon::setLocale('id');

        return view('student-page.user-confirm-schedule', [
            'pageName' => "Konfirmasi Jadwal Kursus | ",
            'enrollment' => $enrollment,
            'student_real_name' => $student_real_name,
            'meetings' => $meetings,
        ]);
    }
```

CourseScheduleController.php (Memeriksa tabrakan jadwal dan menyimpan jadwal yang dipilih)

```
public function storeMeetingData(Request $request, $student_real_name, $enrollment_id, $meeting_number) {
        $request->validate([
            'course_date' => 'required|date',
            'course_time' => 'required',
```

```
        ], [
            'course_date.required' => 'Silahkan Pilih Tanggal Kursus',
            'course_time.required' => 'Silahkan Pilih Salah Satu Opsi',
        ]);

        Carbon::setLocale('id'); // This can be set in a service provider or at the start of your controller
        // Get the Enrollment Data
        $enrollmentData = enrollment::findOrFail($enrollment_id);
        // Fetch the instructor_id from Enrollment Data
        $instructor_id = $enrollmentData['instructor_id'];
        // Get the course_length
        $courseLength = $enrollmentData->course->course_length;
        // dd($request);

        // Extract start and end times from course_time
        list($start_time_str, $end_time_str) = explode(' - ', $request->course_time);

        $currentDate = \Carbon\Carbon::parse($request->course_date);

        // Create full datetime for start and end times
        $selectedStartTime = \Carbon\Carbon::parse($currentDate->format('Y-m-d') . ' ' . $start_time_str);
        $selectedEndTime = \Carbon\Carbon::parse($currentDate->format('Y-m-d') . ' ' . $end_time_str);

        // dd($selectedStartTime, $selectedEndTime);

        // Check for earlier date than the previous meeting
        if ($meeting_number > 1) {
```

```
        $previousMeetingDate = session()->get("meeting_" . ($meeting_number -
1) . "_date");
        $previousMeetingDateCarbon =
\Carbon\Carbon::parse($previousMeetingDate);

        if ($currentDate->lt($previousMeetingDateCarbon)) {
            $request->session()->flash(
                'error',
                'Tanggal untuk Pertemuan ' . $meeting_number . ' tidak boleh lebih
awal dari tanggal Pertemuan ' . ($meeting_number - 1) . ' (' .
$previousMeetingDateCarbon->translatedFormat('d F Y') . ').'
            );
            return redirect()->back();
        }
    }

    // Check for conflicts here (but don't store in database yet)
    $existingSchedule = courseSchedule::where(function ($query) use
($instructor_id, $selectedStartTime, $selectedEndTime) {
        $query->where('instructor_id', $instructor_id)
            ->where(function ($q) use ($selectedStartTime, $selectedEndTime) {
                $q->whereBetween('start_time', [$selectedStartTime,
$selectedEndTime])
                    ->orWhereBetween('end_time', [$selectedStartTime,
$selectedEndTime])
                    ->orWhere(function ($q) use ($selectedStartTime,
$selectedEndTime) {
                        $q->where('start_time', '<=', $selectedStartTime)
                            ->where('end_time', '>=', $selectedEndTime);
                    });
            });
```

```
        })->first();

        // When there's a conflicting schedule, return error
        if ($existingSchedule) {
            $request->session()->flash('error', 'Instruktur ' . $existingSchedule->instructor->fullname . ' sudah terjadwal pada pukul ' . $request->course_time . ' di ' . $currentDate->translatedFormat('d F Y') . '. Silakan pilih waktu atau tanggal lain.');
            return redirect()->back();
        }

        // Store the selected meeting data in session
        session()->put("meeting_{$meeting_number}_date", $request->course_date);
        session()->put("meeting_{$meeting_number}_time", $request->course_time);

        // dd(session()->get("meeting_{$meeting_number}_date"), session()->get("meeting_{$meeting_number}_time"));

        // Redirect to the next meeting or confirmation page
        $nextMeetingNumber = $meeting_number + 1;

        if ($nextMeetingNumber > $courseLength) {
            return redirect()->route('schedule.confirmation', [
                'student_real_name' => $student_real_name,
                'enrollment_id' => $enrollment_id,
            ]);
        } else {
            return redirect()->route('schedule.form', [
                'student_real_name' => $student_real_name,
```

```
            'enrollment_id' => $enrollment_id,
            'meeting_number' => $nextMeetingNumber,
        ]);
    }
  }

  public function saveSchedule(Request $request, $student_real_name,
$enrollment_id) {
    // Save all meeting data to the database
    $enrollment = Enrollment::findOrFail($enrollment_id);
    $course = $enrollment->course;

    // Retrieve the submitted meetings from the request
    $meetings = $request->input('meetings');

    // Save each meeting
    foreach ($meetings as $meetingNumber => $meetingData) {
      $newSchedule = new CourseSchedule();
      $newSchedule->enrollment_id = $enrollment_id;
      $newSchedule->course_id = $course->id;
      $newSchedule->instructor_id = $enrollment->instructor_id;

      // Parse date and time
      $newSchedule->start_time = Carbon::parse($meetingData['date'] . ' ' .
$meetingData['start_time']);
      $newSchedule->end_time = Carbon::parse($meetingData['date'] . ' ' .
$meetingData['end_time']);
      $newSchedule->meeting_number = $meetingNumber; // Convert to 1-
based index
      $newSchedule->theoryStatus = 0;
      $newSchedule->quizStatus = 0;
```

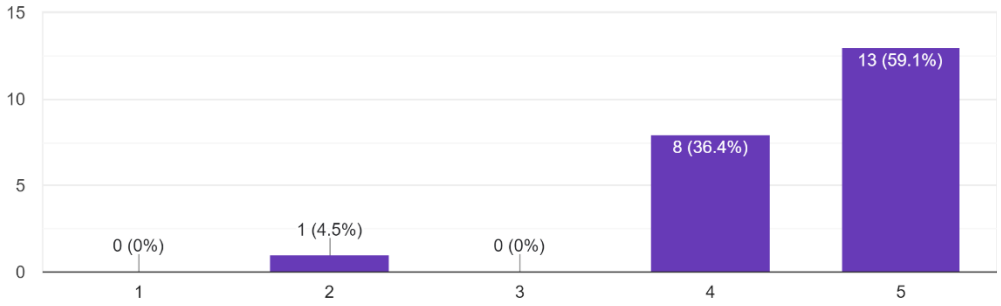
```
            // dd($newSchedule);

            $newSchedule->save();
        }

        // Clear the session
        session()->forget('meeting_*');

        $request->session()->flash('success', 'Jadwal berhasil dibuat. Silahkan hubungi Admin Kursus untuk proses lebih lanjut.');
        return redirect(url('/user-course-progress/' . $student_real_name . '/' . $enrollment_id));
    }
```

LAMPIRAN B

Pertanyaan dan hasil jawaban untuk Faktor *Functional Suitability*

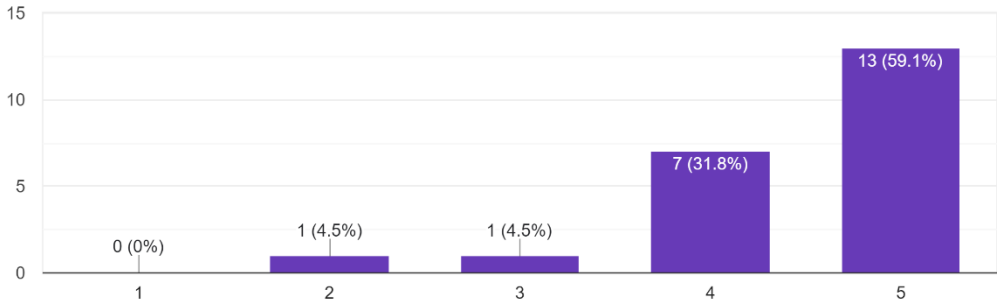
1. Apakah KEMUDI menyediakan semua fungsi yang dibutuhkan oleh Anda untuk mengelola, mendaftar atau melakukan proses kursus mengemudi yang diinginkan?

22 responses



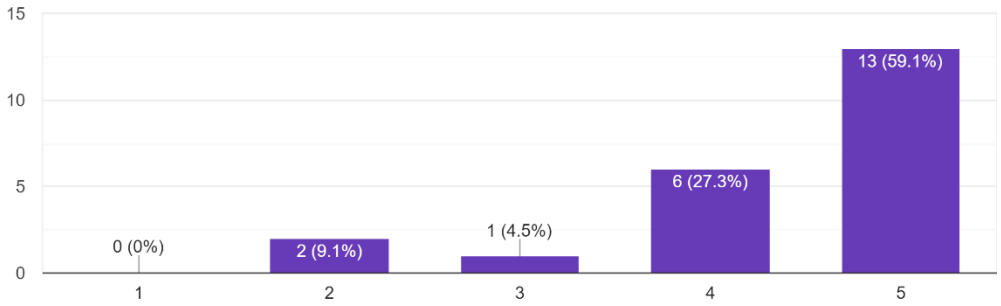
2. Apakah KEMUDI memberikan hasil yang benar dan konsisten sesuai dengan harapan Anda?

22 responses



3. Apakah fungsi-fungsi yang disediakan oleh KEMUDI membantu Anda menyelesaikan mengelola, mendaftar atau melakukan proses kursus mengemudi dengan efisien?

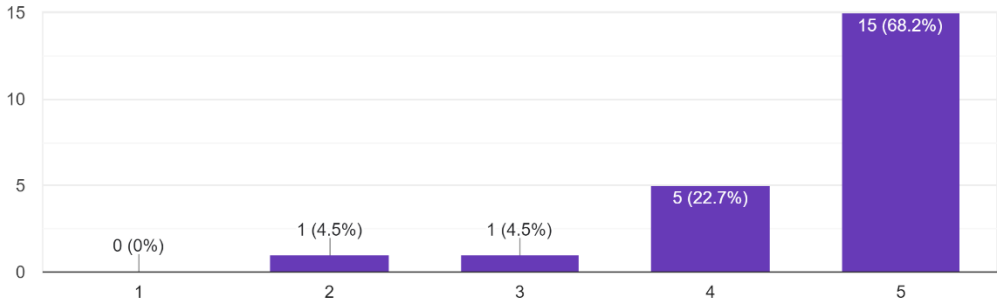
22 responses



Pertanyaan dan hasil jawaban untuk Faktor *Interaction Capability*

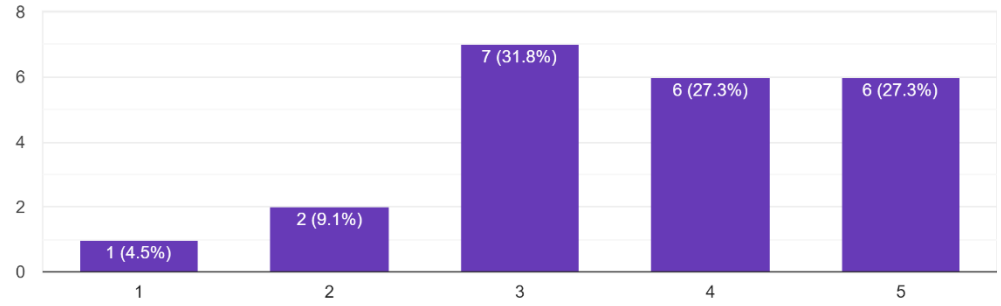
1. Apakah Anda setuju bahwa KEMUDI membantu anda dalam mengelola, mendaftar atau melakukan kursus mengemudi?

22 responses



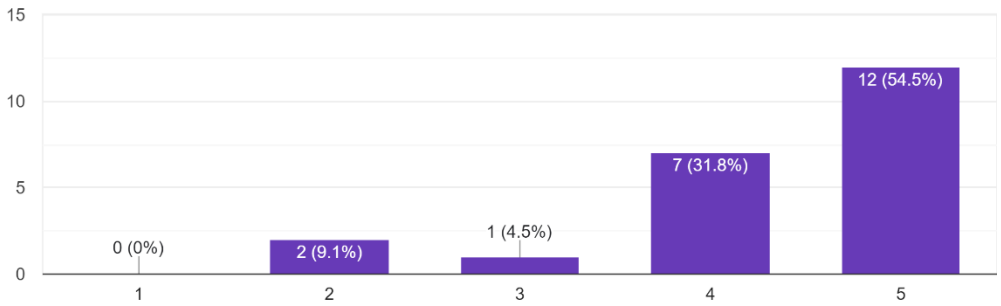
2. Apakah Anda merasa bahwa Anda membutuhkan waktu untuk membiasakan diri dengan KEMUDI?

22 responses



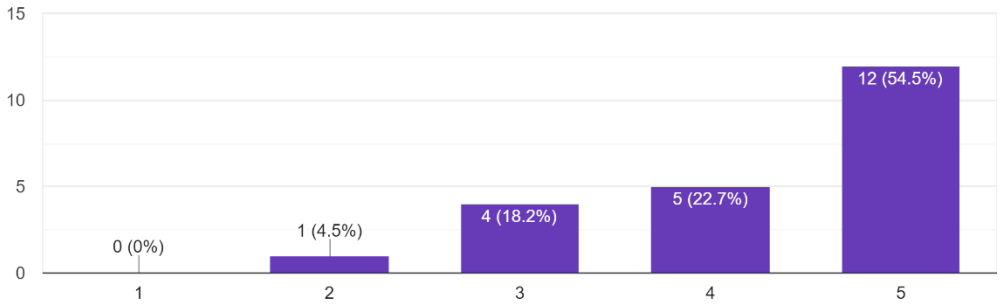
3. Apakah Anda dapat mengoperasikan KEMUDI dengan mudah dan tanpa kesulitan?

22 responses



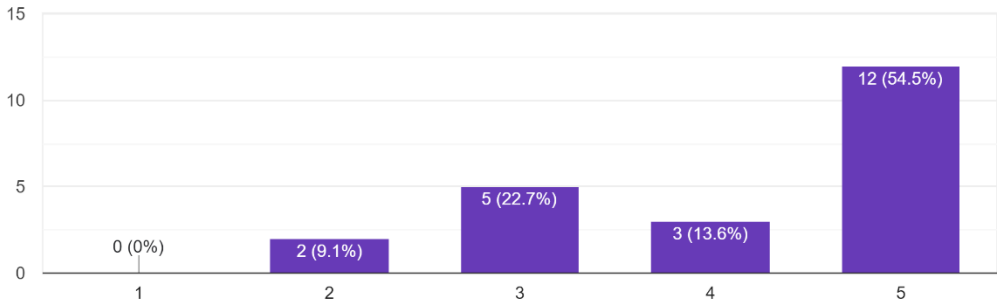
4. Apakah KEMUDI membantu Anda mencegah kesalahan dan memberikan bantuan saat terjadi kesalahan?

22 responses



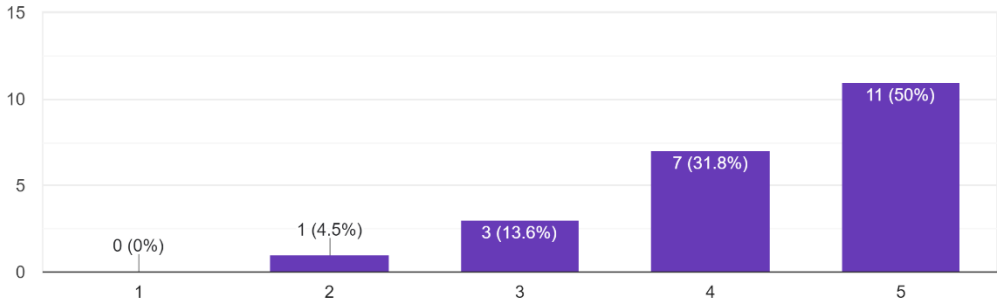
5. Apakah KEMUDI berhasil membuat Anda ingin kembali menggunakannya secara terus menerus?

22 responses



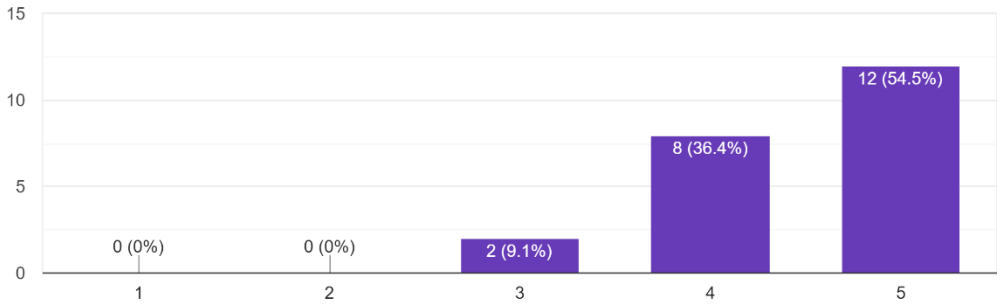
6. Seberapa mudahkah Anda bernavigasi dan memahami berbagai fitur yang ditawarkan oleh KEMUDI, terlepas dari keahlian teknis Anda?

22 responses



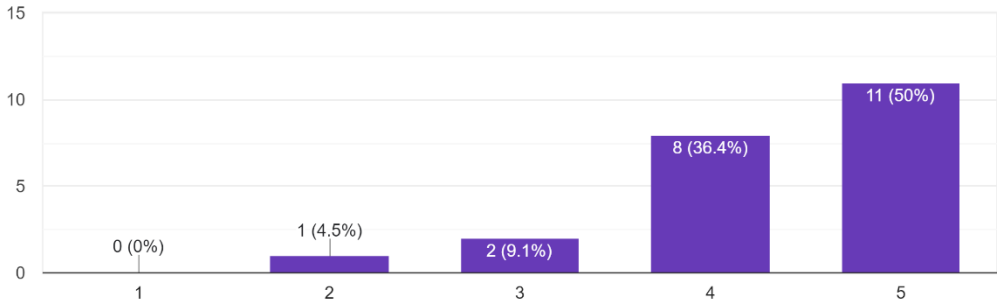
7. Seberapa puaskah Anda dengan ketersediaan dan kualitas bantuan (tutorial, pertanyaan-pertanyaan umum tentang aplikasi, dll.) yang diberikan oleh KEMUDI?

22 responses



8. Apakah label-label, petunjuk, instruksi, dan pesan kesalahan yang ditampilkan oleh KEMUDI sudah jelas dan mudah dipahami?

22 responses



LAMPIRAN C