**LAMPIRAN**

Lampiran 1. Foto Dokumentasi Penelitian

a). Wawancara dan Pengambilan Data dengan Kepala Tim 1 Ruang Matahari ibu Siti Marwa, SST.



b). Proses Pemeriksaan Berkas Penelitian dan Pengambilan Surat Keterangan Penetilian dengan Kepala Ruangan Matahari Ibu Asna Beatrix, SST.



78

c). Proses Validasi Data dan Sistem yang Digunakan Pada Aplikasi Penjadwalan Tenaga Kebidanan Menggunakan Algoritma Memetika Oleh Ka.Sie. Keperawatan dan Kebidanan Bpk. Husen, S.Kep., M.kes.

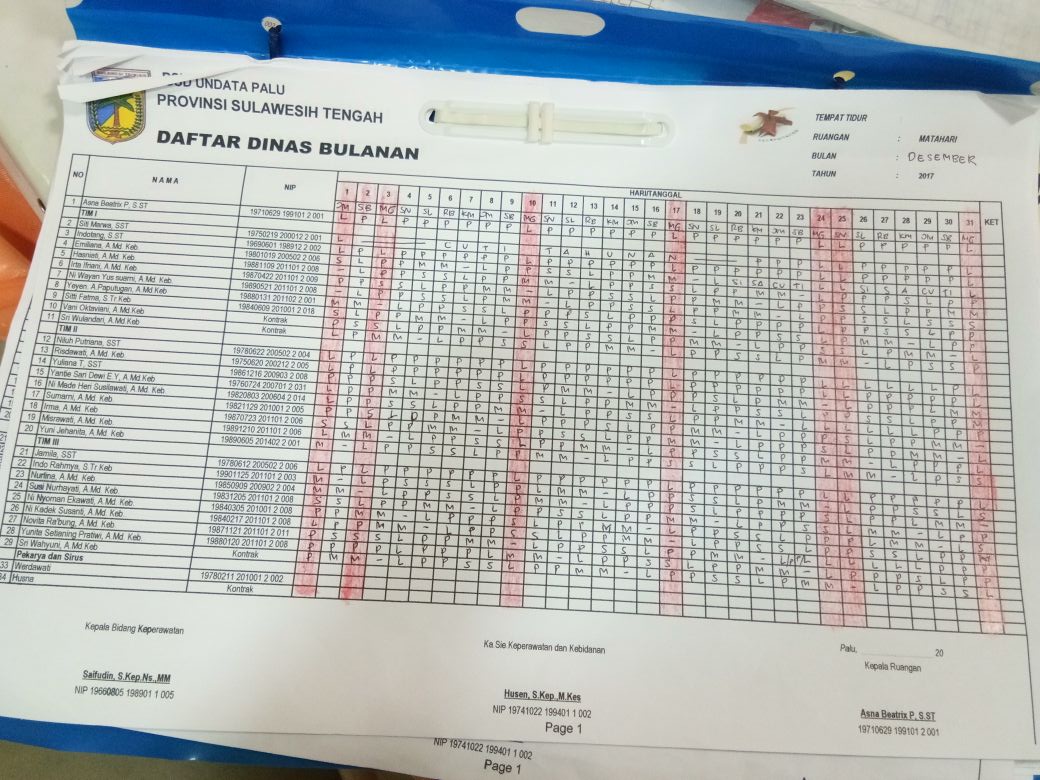




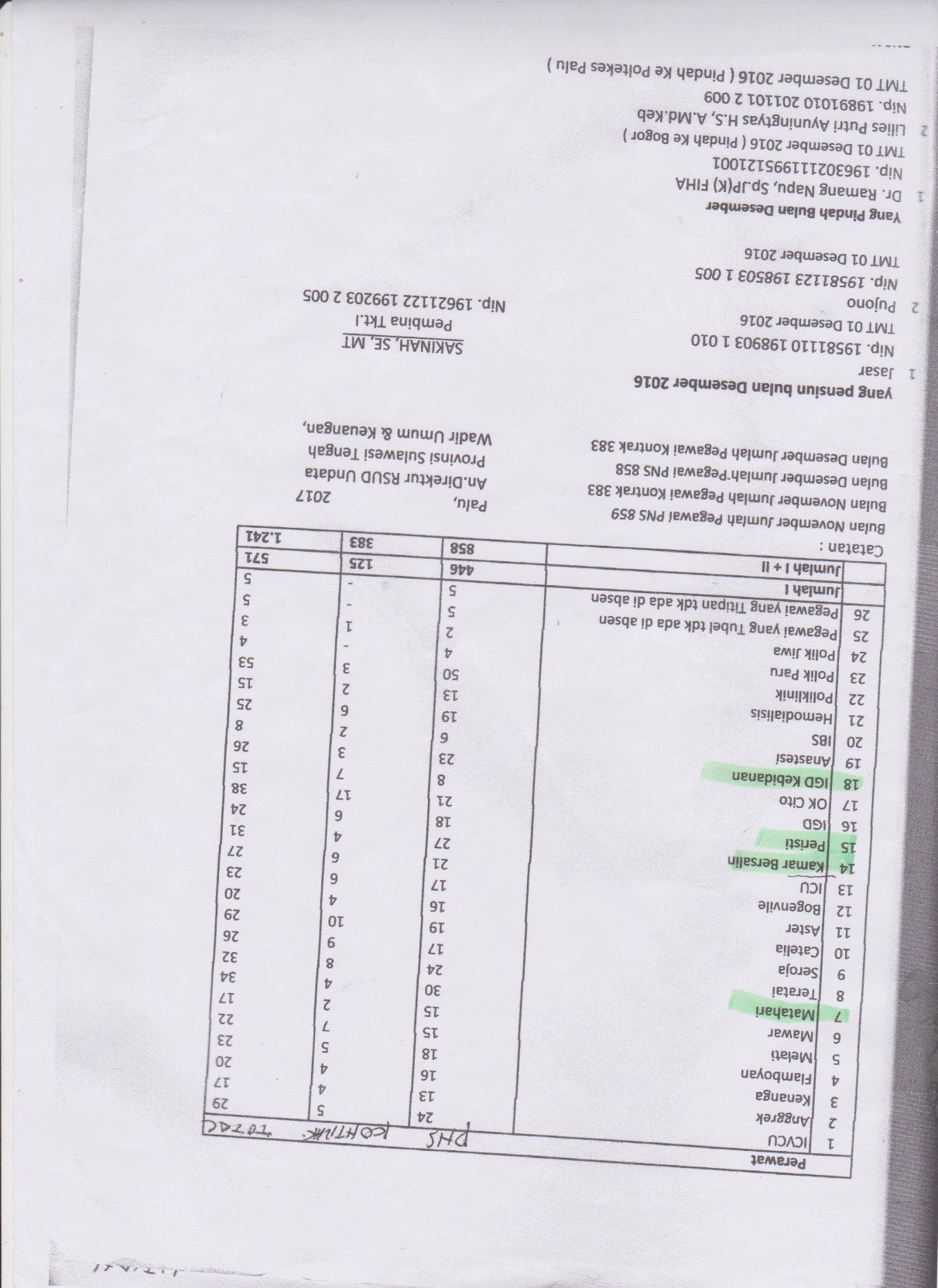
d). Struktur Organisasi Rungan Matahari Kebidanan RSUD Undata Palu



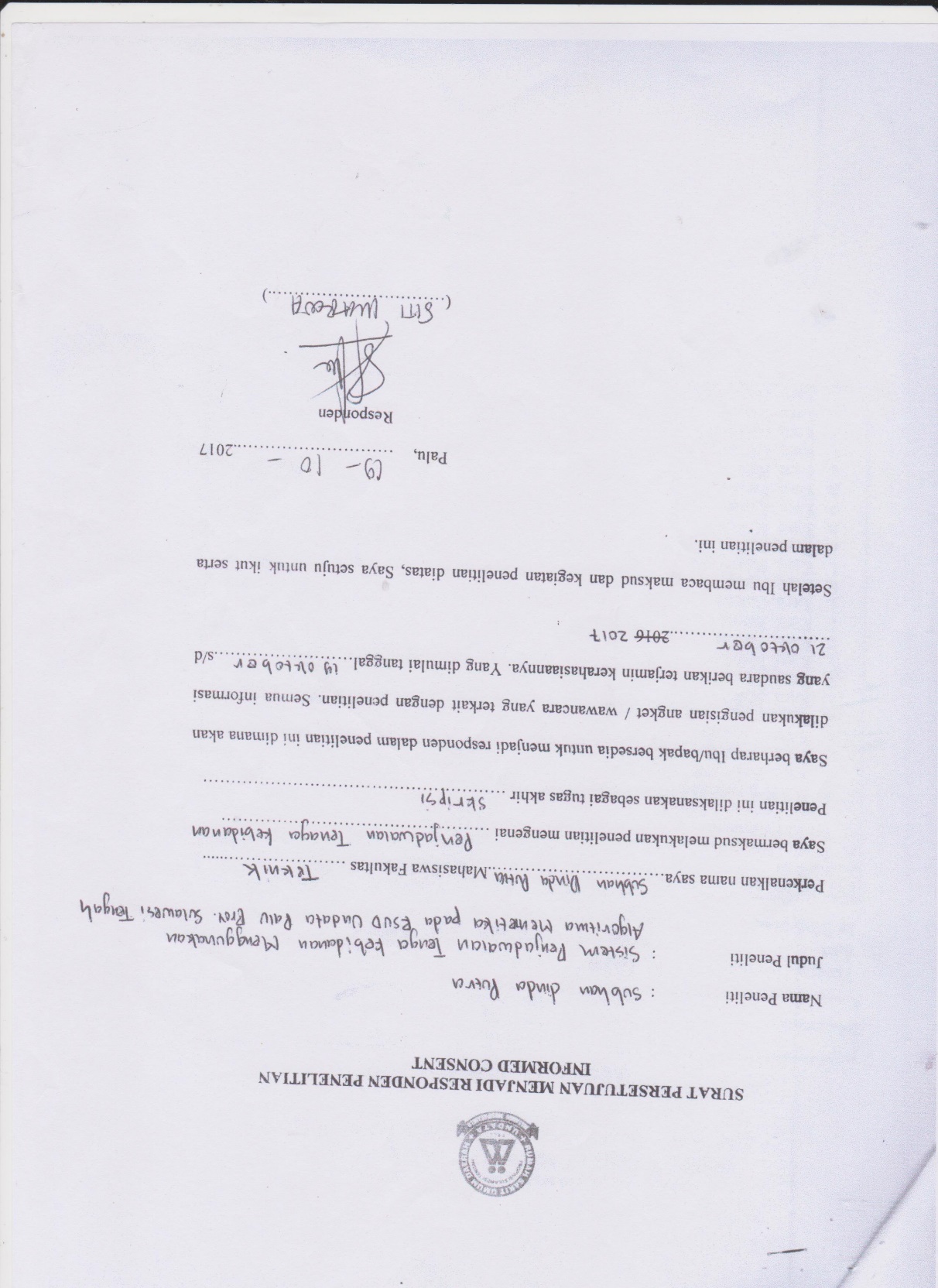
e). Jadwal Periode Desember 2017

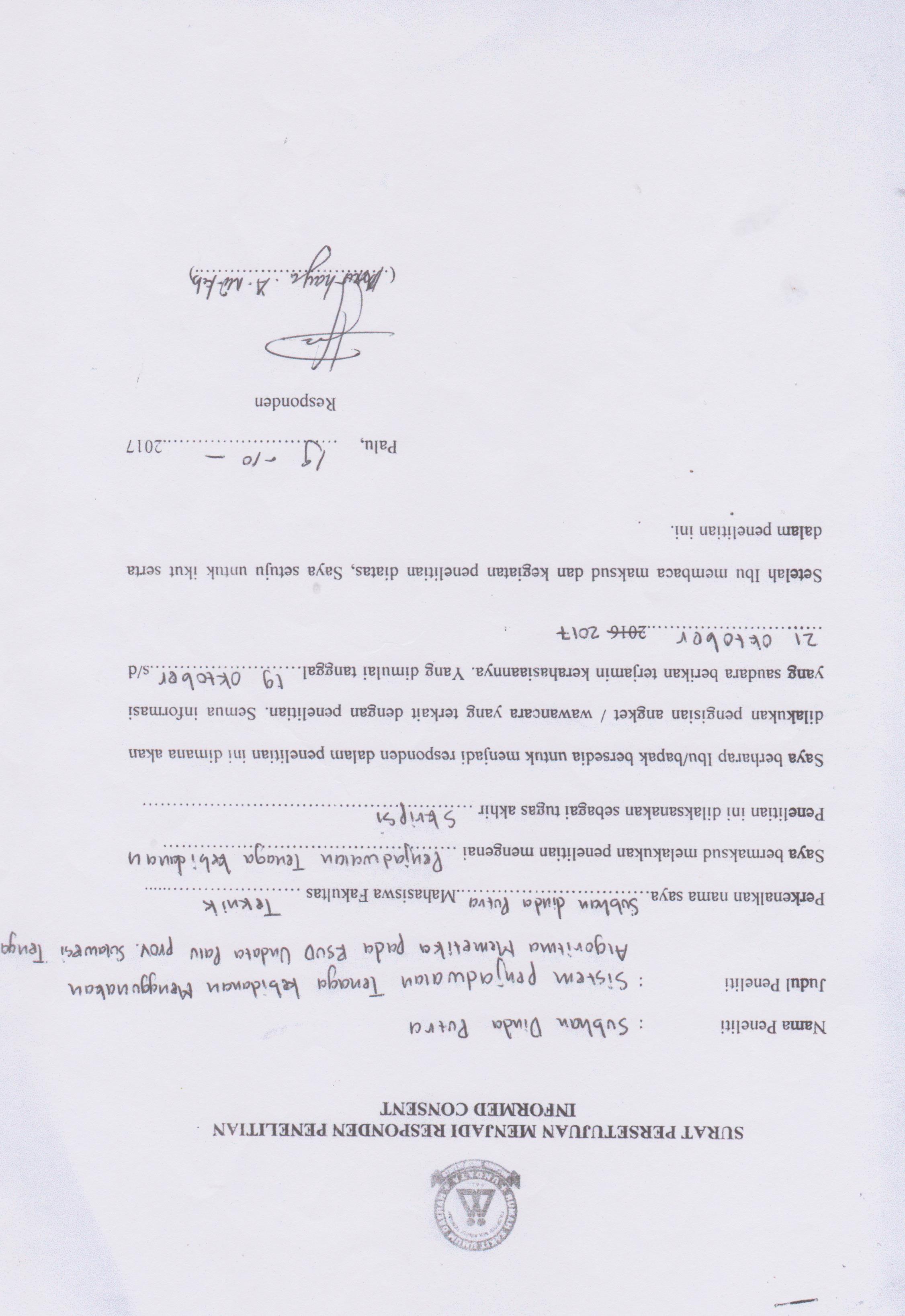


Lampiran 2. Data Ruangan Tenaga Kebidanan

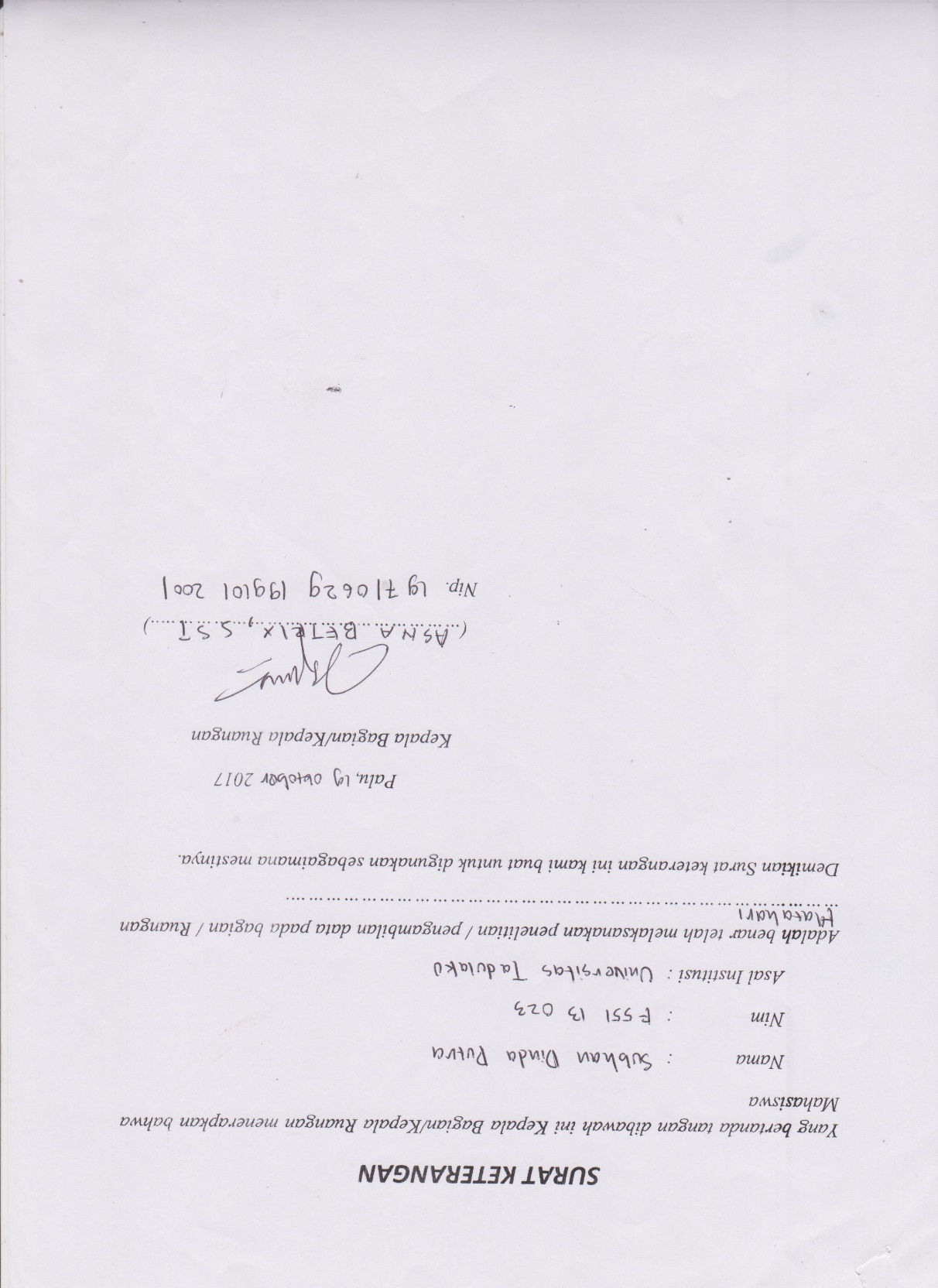


Lampiran 3. Surat Persetujuan Menjadi Responden Penelitian

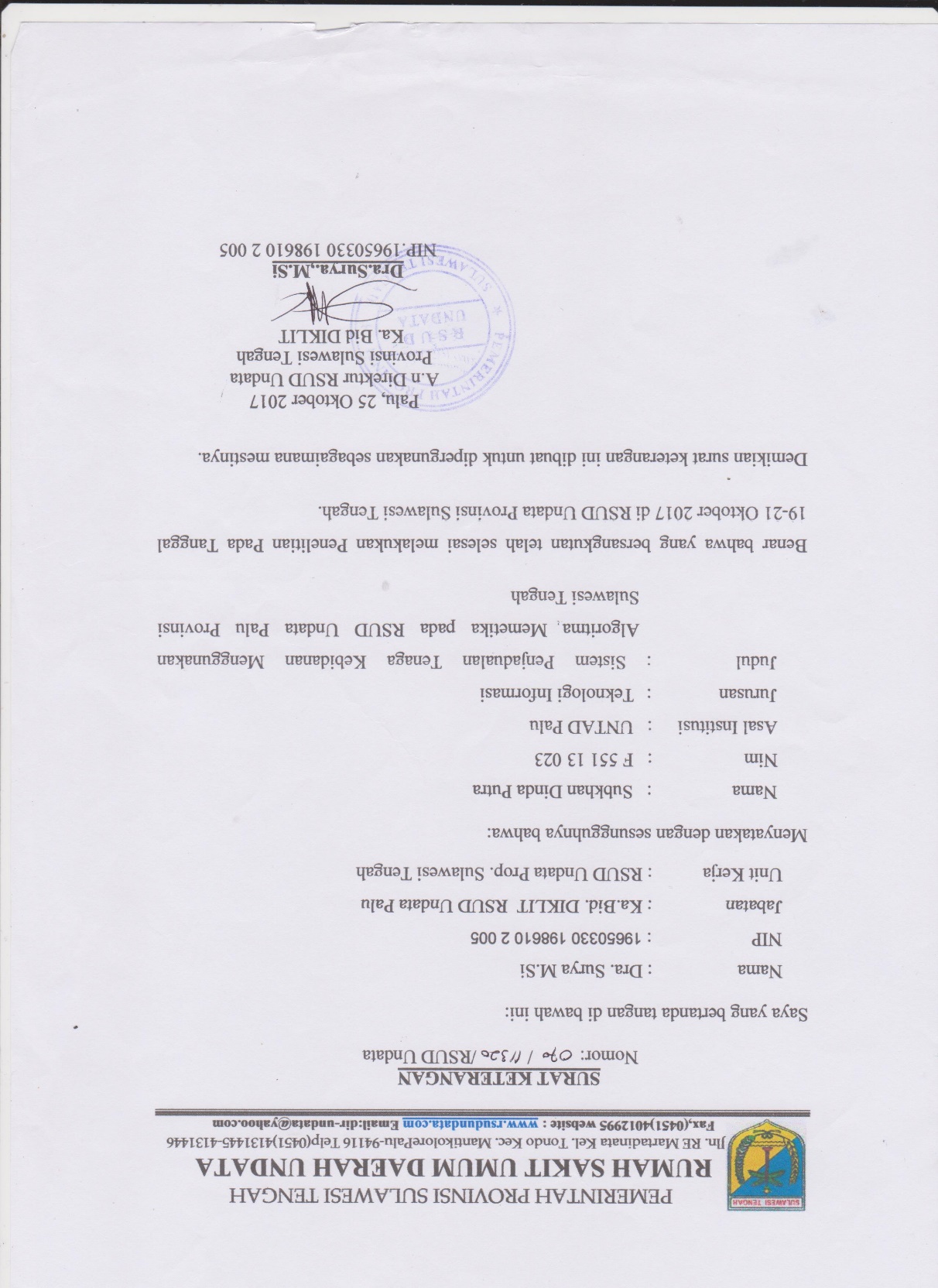




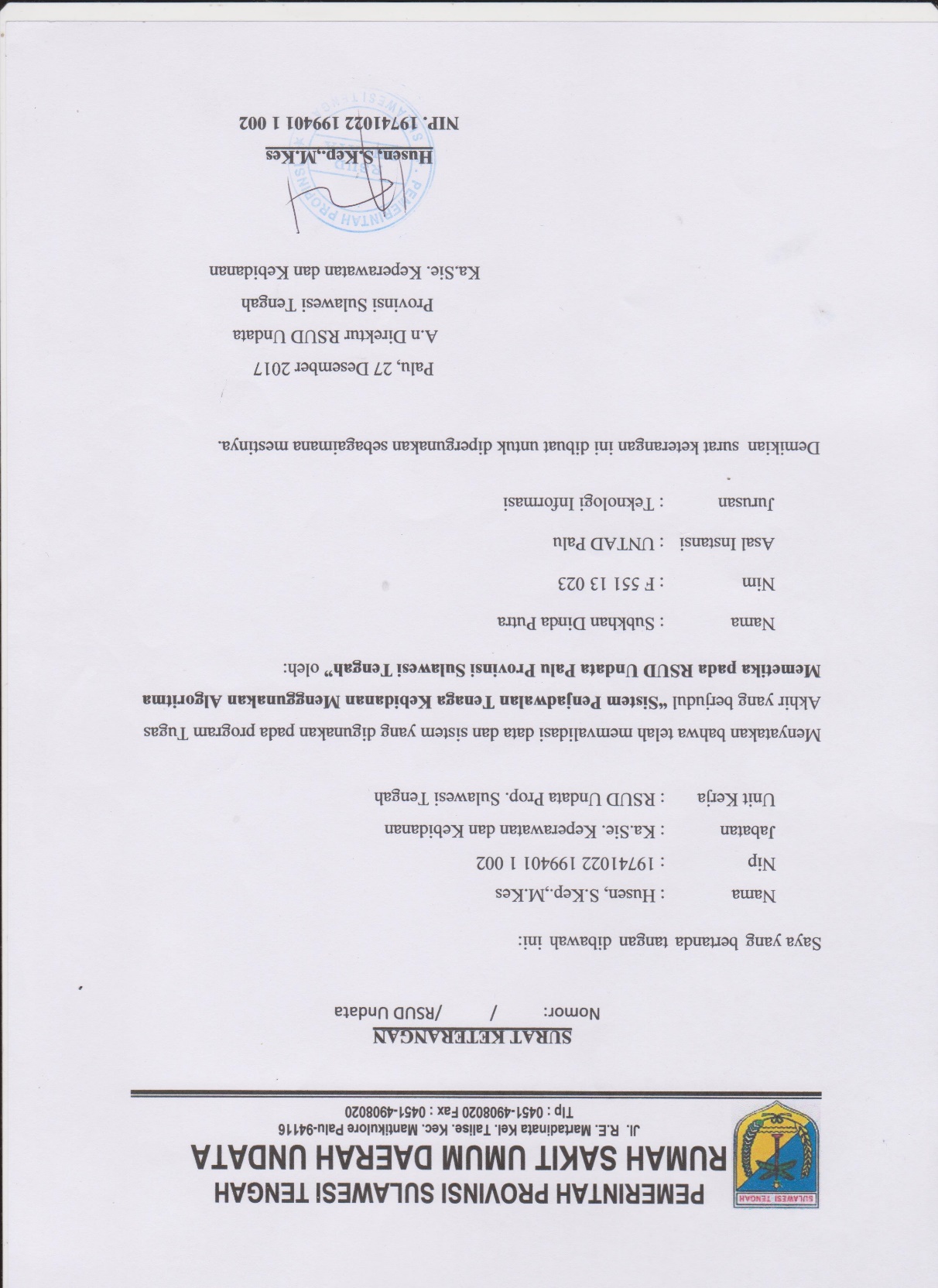
Lampiran 4. Surat Keterangan Melakukan Penelitian di Ruangan Matahari



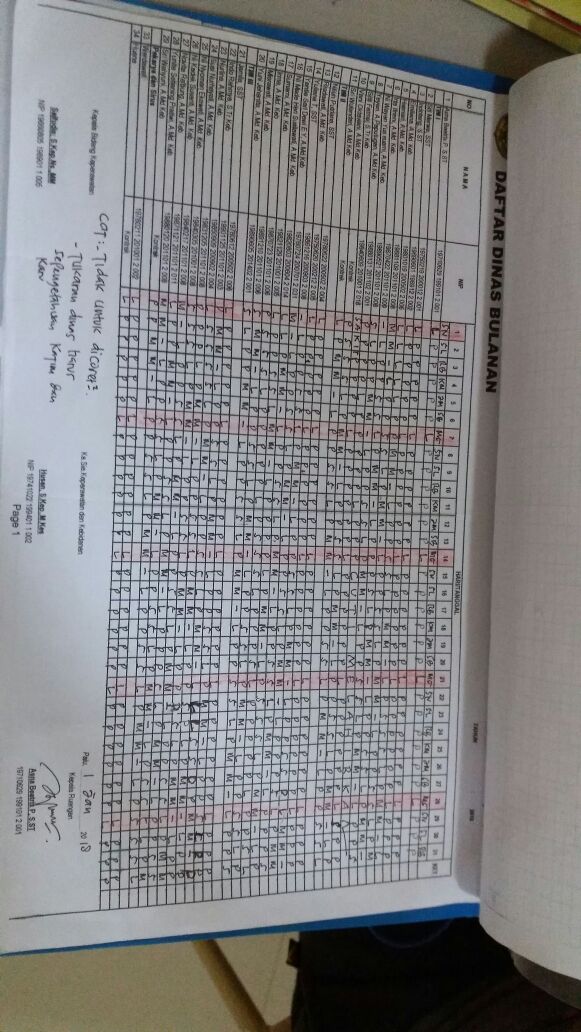
Lampiran 5. Surat Keterangan Melakukan Penelitian di RSUD Undata Palu

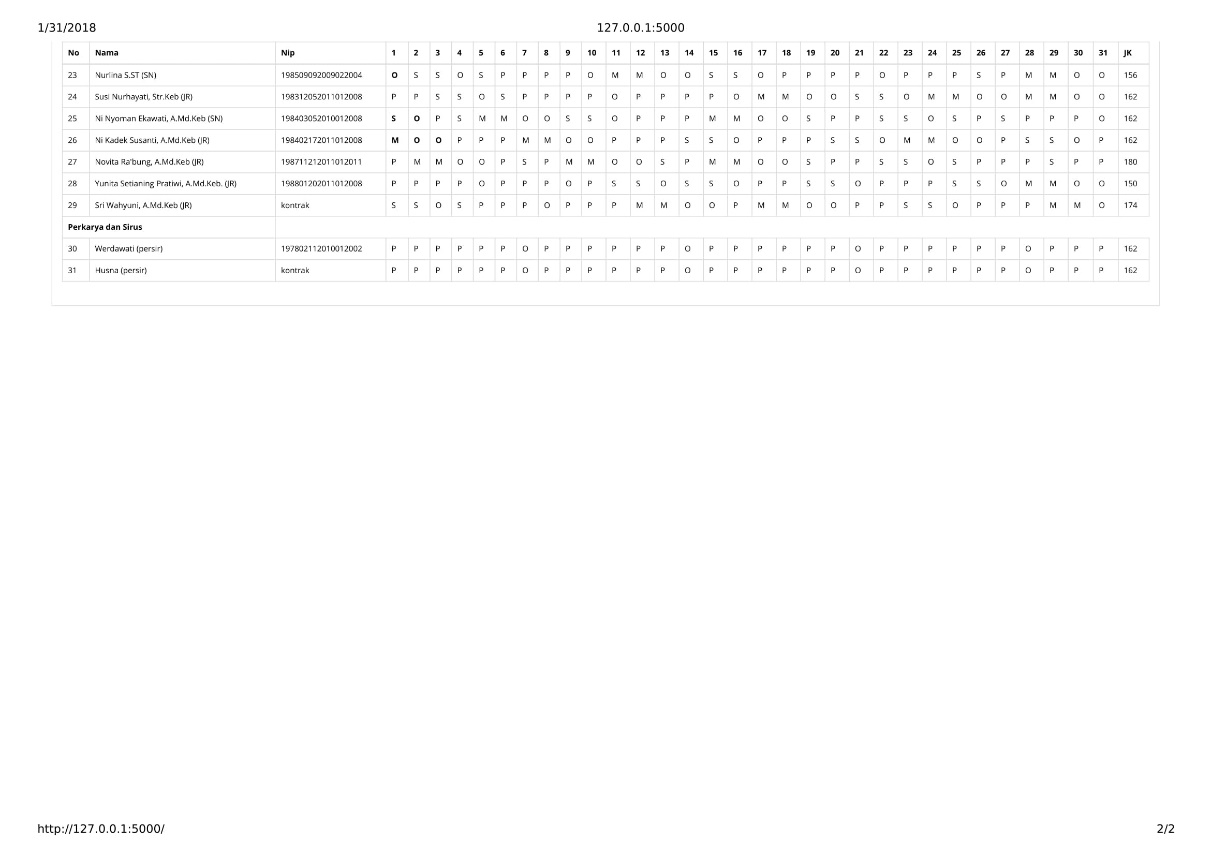
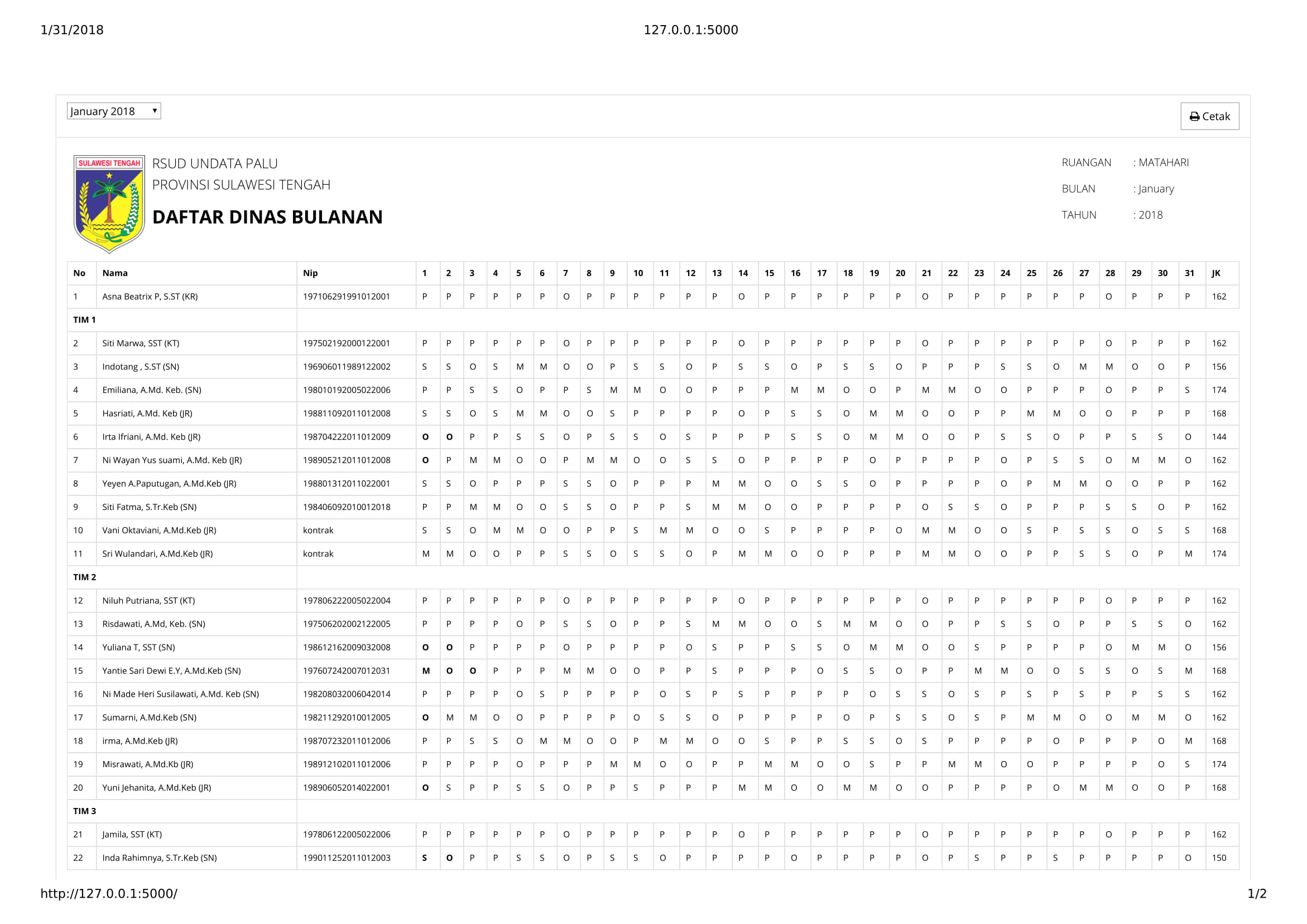


Lampiran 6. Surat Keterangan Validasi Data dan Sistem Aplikasi



Lampiran 7. Perbandingan hasil penjadwalan manual dan hasil penjadwalan sistem

1. Jadwal periode Januari 2018, hasil penjadwalan manual.
2. jadwal periode Januari 2018, hasil penjadwalan sistem.



Lampiran 8. *Source Code* Aplikasi

1. **from** app **import** db
2. **from** models **import** Schedules, Bidan, Periode
3. **import** random, operator, json
4. **import** timeit
5. **import** copy
6. **import** cPickle
7. **from** datetime **import** datetime
9. **class** Memetic():
10. shift = [['P'], ['P', 'P'], ['P', 'P', 'P'], ['P', 'P', 'P', 'O'],
11. ['P', 'P', 'P', 'P', 'O'], ['S'], ['S', 'S', 'O'],
12. ['M', 'M', 'O', 'O']]
13. hard\_penalti = 5
14. soft\_penalti = 1
16. **def** \_\_init\_\_(self, init\_data):
17. self.start\_time = datetime.now()
18. self.get\_pattern\_schedule()
20. **def** detail\_solusi(self):
21. **print** "FITNESS TERTINGGI: %f" % (self.elit\_individu["fitness"])
22. self.single\_fitness(self.elit\_individu["individu"], True)

25. **def** get\_pattern\_schedule(self):
26. current\_bidan\_schedule = Schedules.query \
27. .join(Bidan) \
28. .add\_columns(Bidan.id, Bidan.officer, Schedules.rest\_shift) \
29. .filter(Schedules.periode\_id == self.periode\_id).all()
30. **for** bdn\_sch **in** current\_bidan\_schedule:
31. rest\_shift = bdn\_sch.rest\_shift
32. ''.join(rest\_shift.split())
33. **if** rest\_shift != "CLEAR":
34. rest\_shift = rest\_shift.split(",")
35. self.bidan\_w\_schedule[bdn\_sch.id] = {"officer": bdn\_sch.officer,
36. "rest\_shift": rest\_shift}

39. **def** generate\_random\_shift(self, length=None):
40. **if** length **is** None:
41. length = self.hari
42. shift\_bidan = []
43. **while** True:
44. len\_shift = len(self.shift) - 1
45. rand\_shift = random.randint(0, len\_shift)
46. shift\_bidan = shift\_bidan + self.shift[rand\_shift]
47. **if** len(shift\_bidan) >= length:
48. shift\_bidan = shift\_bidan[0:length]
49. **return** shift\_bidan


53. **def** initial\_populasi(self):
55. static\_shift = []
57. **for** d **in** range(self.hari):
58. static\_shift.append('P')
60. **for** ahad **in** self.ahads:
61. static\_shift[ahad] = 'O'
63. **for** i **in** range(self.populasi):
64. individu = {}
65. **for** bdn **in** Bidan.query.all():
66. **if** bdn.officer == "SN" **or** bdn.officer == "JR":
67. individu[bdn.id] = self.generate\_random\_shift()
68. **else**:
69. self.individu\_static[bdn.id] = static\_shift
71. self.lingkungan\_individu.append(individu)



76. **def** min\_bidan\_improve(self, individu, data):
77. bidan\_id = self.bidan\_w\_schedule.keys()
78. hari = data['hari']
79. need\_shift = data['need\_shift']
80. total\_need = data['total\_need']
81. improve = True
82. **while** improve:
83. rand\_id = random.randint(0, len(bidan\_id) - 1)
84. id = bidan\_id[rand\_id]
85. restshift = self.bidan\_w\_schedule[id]['rest\_shift']
86. officer = self.bidan\_w\_schedule[id]["officer"]
87. **if** officer != "KT" **and** officer != "KR" **and** officer != "persir":
88. **if** data['jenis'] == officer:
89. **if** restshift != "CLEAR":
90. **if** restshift[hari] == "-":
91. shift = individu[id][hari]
92. **else**:
93. shift = None
94. **else**:
95. shift = individu[id][hari]
97. **if** shift **is** **not** None:
98. **if** hari + 1 <= self.hari - 1:
99. nextshift = individu[id][hari + 1]
100. **else**:
101. nextshift = "E"
103. **if** hari + 2 <= self.hari - 1:
104. next2shift = individu[id][hari + 2]
105. **else**:
106. next2shift = "E"
108. **if** hari + 3 <= self.hari - 1:
109. next3shift = individu[id][hari + 3]
110. **else**:
111. next3shift = "E"
113. **if** shift **is** **not** None:
114. **if** shift != need\_shift:
115. individu[id][hari] = need\_shift
116. **if** need\_shift == "M":
117. **if** nextshift != "E":
118. individu[id][hari + 1] = "M"
119. **if** next2shift != "E":
120. individu[id][hari + 2] = "O"
121. **if** next3shift != "E":
122. individu[id][hari + 3] = "O"
123. **if** need\_shift == "S":
124. **if** nextshift != "E":
125. individu[id][hari + 1] = "S"
126. **if** next2shift != "E":
127. individu[id][hari + 2] = "O"

130. total\_need = total\_need - 1
131. **if** total\_need <= 0:
132. improve = False

135. **def** replace\_shift(self, individu\_row, hari, length\_shift):
136. shift\_generate = self.generate\_random\_shift(length\_shift)
137. hari -= len(shift\_generate) - 1
138. **for** s **in** shift\_generate:
139. individu\_row[hari] = s
140. hari += 1

143. **def** day\_off(self, individu, process="fitness", debug=False):
144. pelanggaran\_total = 0
145. **for** id, bdn\_w\_sch **in** self.bidan\_w\_schedule.items():
147. pelanggaran\_off\_siang, pelanggaran\_off\_malam = 0, 0
148. pelanggaran\_off\_day, pelanggaran\_off = 0, 0
149. siang, malam, pagi, off = 0, 0, 0, 0
151. **if** bdn\_w\_sch["officer"] == "SN" **or** bdn\_w\_sch["officer"] == "JR":
152. restshift = bdn\_w\_sch["rest\_shift"]
153. **for** hari **in** range(self.hari):
154. **if** restshift != "CLEAR":
155. **if** restshift[hari] == "-":
156. shift = individu[id][hari]
157. immutable\_shift = None
158. **else**:
159. shift = restshift[hari]
160. immutable\_shift = True
162. **else**:
163. shift = individu[id][hari]
164. immutable\_shift = None
166. **if** shift **is** **not** None:
167. **if** hari + 1 <= self.hari - 1:
168. nextshift = individu[id][hari + 1]
169. **else**:
170. nextshift = "E"
172. **if** hari + 2 <= self.hari - 1:
173. next2shift = individu[id][hari + 2]
174. **else**:
175. next2shift = "E"
177. **if** hari + 3 <= self.hari - 1:
178. next3shift = individu[id][hari + 3]
179. **else**:
180. next3shift = "E"
182. **if** hari - 2 >= 0 **and** hari - 2 <= self.hari - 1:
183. **if** restshift != "CLEAR":
184. **if** restshift[hari - 2] != "-":
185. prev2shift = restshift[hari - 2]
186. **else**:
187. prev2shift = individu[id][hari - 2]
188. **else**:
189. prev2shift = individu[id][hari - 2]
190. **else**:
191. prev2shift = None

194. **if** restshift != "CLEAR":
195. **if** hari + 1 <= self.hari - 1:
196. **if** restshift[hari + 1] != "-":
197. nextrestshift = restshift[hari + 1]
198. **else**:
199. nextrestshift = None
200. **else**:
201. nextrestshift = None
203. **if** hari - 1 >= 0 **and** hari - 1 <= self.hari - 1:
204. **if** restshift[hari - 1] != "-":
205. prevrestshift = restshift[hari - 1]
206. **else**:
207. prevrestshift = None
208. **else**:
209. prevrestshift = None

212. **if** hari + 2 <= self.hari - 1:
213. **if** restshift[hari + 2] != "-":
214. next2restshift = restshift[hari + 2]
215. **else**:
216. next2restshift = None
217. **else**:
218. next2restshift = None
219. **else**:
220. nextrestshift = None
221. next2restshift = None
222. prevrestshift = None

225. **if** shift == "O":
226. siang, malam, pagi = 0, 0, 0
227. off += 1
228. **if** off == 2 **and** immutable\_shift **is** None **and** prev2shift == "P":
229. **if** process == "fitness":
230. pelanggaran\_off += 1
231. **elif** process == "improvement":
232. individu[id][hari] = "S"
233. off = 0
234. **elif** prevrestshift == "O" **and** immutable\_shift **is** None:
235. **if** process == "fitness":
236. pelanggaran\_off += 1
237. **elif** process == "improvement":
238. individu[id][hari] = "P"
239. off = 0
240. **elif** off == 1 **and** hari == 0 **and** immutable\_shift **is** None:
241. **if** process == "fitness":
242. pelanggaran\_off += 1
243. **elif** process == "improvement":
244. individu[id][hari] = "P"
245. off = 0
246. **elif** off > 2:
247. **if** process == "fitness":
248. pelanggaran\_off += 1
249. **elif** process == "improvement":
250. individu[id][hari] = "P"
251. off = 0
252. **elif** shift == "P":
253. siang, malam, off = 0, 0, 0
254. pagi += 1
256. **if** pagi == 4:
257. **if** next2shift == "O":
258. **if** process == "fitness":
259. pelanggaran\_off\_day += 1
260. **elif** process == "improvement":
261. individu[id][hari + 2] = "P"
263. **if** nextrestshift **is** **not** None **and** immutable\_shift **is** None:
264. **if** process == "fitness":
265. pelanggaran\_off\_day += 1
266. **elif** process == "improvement":
267. self.replace\_shift(individu[id], hari, pagi)
268. pagi = 0
269. **elif** nextshift != "O" **and** nextshift != "E":
270. **if** process == "fitness":
271. pelanggaran\_off\_day += 1
272. **elif** process == "improvement":
273. individu[id][hari + 1] = "O"
274. pagi = 0
275. **elif** pagi < 3:
276. **if** nextshift == "O":
277. **if** process == "fitness":
278. pelanggaran\_off\_day += 1
279. **elif** process == "improvement":
280. individu[id][hari + 1] = "P"
282. **elif** shift == "S":
283. malam, off, pagi = 0, 0, 0
284. **if** siang == 0:
285. **if** nextrestshift **is** **not** None **and** next2restshift **is** **not** None
286. **and** immutable\_shift **is** None:
287. **if** nextrestshift == "S" **and** next2restshift == "S":
288. **if** process == "fitness":
289. pelanggaran\_off\_siang += 1
290. **elif** process == "improvement":
291. self.replace\_shift(individu[id], hari, 1)
292. **else**:
293. **if** nextshift == "O":
294. **if** process == "fitness":
295. pelanggaran\_off\_siang += 1
296. **elif** process == "improvement":
297. individu[id][hari + 1] = "P"
298. **else**:
299. siang += 1
300. **elif** siang == 1:
301. **if** nextrestshift **is** **not** None **and** immutable\_shift **is** None:
302. **if** process == "fitness":
303. pelanggaran\_off\_siang += 1
304. **elif** process == "improvement":
305. self.replace\_shift(individu[id], hari, 2)
307. **elif** nextshift != "O" **and** nextshift != "E":
309. **if** process == "fitness":
310. pelanggaran\_off\_siang += 1
311. **elif** process == "improvement":
312. individu[id][hari + 1] = "O"
313. **elif** next2shift == "O":
315. **if** process == "fitness":
316. pelanggaran\_off\_siang += 1
317. **elif** process == "improvement":
318. individu[id][hari + 2] = "P"
319. siang = 0
320. **else**:
321. siang += 1
322. **elif** shift == "M":
323. siang, off, pagi = 0, 0, 0
324. **if** malam == 0 **and** nextshift != "M" **and** nextshift != "E":
325. **if** nextrestshift **is** **not** None **and** immutable\_shift **is** None:
326. **if** process == "fitness":
327. pelanggaran\_off\_malam += 1
328. **elif** process == "improvement":
329. self.replace\_shift(individu[id], hari, 1)
330. **else**:
331. malam += 1
332. **if** process == "fitness":
333. pelanggaran\_off\_malam += 1
334. **elif** process == "improvement":
335. individu[id][hari + 1] = "M"
336. **if** next2shift != "E":
337. individu[id][hari + 2] = "O"
339. **if** next3shift != "E":
340. individu[id][hari + 3] = "O"
342. **elif** malam == 1:
343. **if** nextrestshift **is** **not** None **and** immutable\_shift **is** None:
344. **if** process == "fitness":
345. pelanggaran\_off\_malam += 1
346. **elif** process == "improvement":
347. self.replace\_shift(individu[id], hari, 2)
349. **elif** nextshift != "O" **and** nextshift != "E":
350. **if** process == "fitness":
351. pelanggaran\_off\_malam += 1
352. **elif** process == "improvement":
353. individu[id][hari + 1] = "O"
355. **if** next2restshift **is** **not** None **and** immutable\_shift **is** None:
356. **if** process == "fitness":
357. pelanggaran\_off\_malam += 1
358. **elif** process == "improvement":
359. self.replace\_shift(individu[id], hari, 3)
361. **elif** next2shift != "O" **and** next2shift != "E":
362. **if** process == "fitness":
363. pelanggaran\_off\_malam += 1
364. **elif** process == "improvement":
365. individu[id][hari + 2] = "O"
367. malam = 0
368. **else**:
369. malam += 1
370. **else**:
371. siang, malam, off, pagi = 0, 0, 0, 0
373. pelanggaran\_total += pelanggaran\_off\_malam + pelanggaran\_off\_siang +
374. pelanggaran\_off\_day + pelanggaran\_off
376. **if** debug:
377. **if** pelanggaran\_off > 0 **or** pelanggaran\_off\_day > 0 **or**
378. pelanggaran\_off\_malam > 0 **or** pelanggaran\_off\_siang > 0:
379. **print** "[DAY OFF] PELANGGARAN BIDAN - %d" % (id)
380. **if** pelanggaran\_off\_siang > 0:
381. **print** "---S = %d" % (pelanggaran\_off\_siang)
382. **if** pelanggaran\_off\_malam > 0:
383. **print** "---M = %d" % (pelanggaran\_off\_malam)
384. **if** pelanggaran\_off\_day > 0:
385. **print** "---Off Day = %d" % (pelanggaran\_off\_day)
386. **if** pelanggaran\_off > 0:
387. **print** "Off day kelebihan = %d" % (pelanggaran\_off)
389. **if** debug:
390. **print** "[DAY OFF] Total Pelanggaran: %d" % (pelanggaran\_total)
392. **if** process == "fitness":
393. **return** pelanggaran\_total

396. **def** pairshift\_overflow(self, individu, process="fitness", debug=False):
397. pelanggaran\_total = 0
398. **for** id, bdn\_w\_sch **in** self.bidan\_w\_schedule.items():
399. pelanggaran, pair\_shift\_malam, pair\_shift\_siang = 0, 0, 0
400. pelanggaran\_pola\_malam = 0
401. malam, siang = 0, 0
402. h = 0
403. pair\_patteran\_malam = []
404. **if** bdn\_w\_sch["officer"] == "SN" **or** bdn\_w\_sch["officer"] == "JR":
405. restshift = bdn\_w\_sch["rest\_shift"]
406. **for** hari **in** range(self.hari):
408. **if** restshift != "CLEAR":
409. **if** restshift[hari] == "-":
410. shift = individu[id][hari]
411. **else**:
412. shift = restshift[hari]
413. **else**:
414. shift = individu[id][hari]
416. h += 1
418. **if** shift **is** **not** None:
419. **if** hari + 1 <= self.hari - 1:
420. nextshift = individu[id][hari + 1]
421. **else**:
422. nextshift = "E"
424. **if** shift == "M":
425. **if** nextshift == "M":
426. pair\_patteran\_malam.append([hari, hari+1])
428. malam += 1
429. siang = 0
430. **if** malam == 2:
431. pair\_shift\_malam += 1
432. **if** pair\_shift\_malam > 1 **and** h <= 7:
433. pair\_shift\_malam = 0
434. **if** process == "fitness":
435. pelanggaran += 1
436. **elif** process == "improvement":
437. self.replace\_shift(individu[id], hari, h)
439. malam = 0
440. **elif** shift == "S":
441. siang += 1
442. malam = 0
443. **if** siang == 2:
444. pair\_shift\_siang += 1
445. **if** pair\_shift\_siang > 1 **and** h <= 7:
446. pair\_shift\_siang = 0
447. **if** process == "fitness":
448. pelanggaran += 1
449. **elif** process == "improvement":
450. self.replace\_shift(individu[id], hari, h)
451. siang = 0
452. **else**:
453. malam = 0
454. siang = 0
456. **if** h == 7:
457. pair\_shift\_malam, pair\_shift\_siang= 0, 0
458. malam, siang = 0, 0
459. h = 0
461. **if** len(pair\_patteran\_malam) > 3:
462. **if** process == "fitness":
463. pelanggaran\_pola\_malam += 1
464. **elif** process == "improvement":
465. generate\_shift = self.generate\_random\_shift(2)
466. rand\_index = random.randint(0, len(pair\_patteran\_malam)-1)
467. **for** i,index\_hari **in** enumerate(pair\_patteran\_malam[rand\_index]):
468. **if** restshift != "CLEAR":
469. **if** restshift[index\_hari] == "-":
470. individu[id][index\_hari] = generate\_shift[i]
471. **else**:
472. individu[id][index\_hari] = generate\_shift[i]
474. **if** debug:
475. **if** pelanggaran > 0 **or** pelanggaran\_pola\_malam > 0:
476. **print** "[PAIR SHIFT] PELANGGARAN BIDAN - %d" % (id)
477. **if** pelanggaran > 0:
478. **print** "Pelanggaran pair shift: %d" % (pelanggaran)
479. **if** pelanggaran\_pola\_malam > 0:
480. **print** "Pelanggaran pola malam: %d" % (pelanggaran\_pola\_malam)
482. pelanggaran\_total += pelanggaran + pelanggaran\_pola\_malam
484. **if** debug:
485. **print** "[PAIR SHIFT] TOTAL PELANGGARAN: %d" % (pelanggaran\_total)
487. **if** process == "fitness":
488. **return** pelanggaran\_total

491. **def** working\_hours(self, individu, process="fitness", debug=False):
492. pelanggaran\_total = 0
493. p, s, m = 6,6,12
494. min\_hours = 120
495. max\_hours = 180
496. **for** id, bdn\_w\_sch **in** self.bidan\_w\_schedule.items():
497. pelanggaran = 0
498. **if** bdn\_w\_sch["officer"] == "SN" **or** bdn\_w\_sch["officer"] == "JR":
499. restshift = bdn\_w\_sch["rest\_shift"]
500. total\_jam = 0
501. **for** hari **in** range(self.hari):
502. **if** restshift != "CLEAR":
503. **if** restshift[hari] == "-":
504. shift = individu[id][hari]
505. **else**:
506. shift = restshift[hari]
507. **else**:
508. shift = individu[id][hari]

511. **if** shift == "P":
512. total\_jam = total\_jam + p
513. **elif** shift == "S":
514. total\_jam = total\_jam + s
515. **elif** shift == "M":
516. total\_jam = total\_jam + m
518. # print "[TOTAL\_JAM] Bidan ke - %d adalah (%d)" % (id, total\_jam)
520. **if** total\_jam < min\_hours **or** total\_jam > max\_hours:
521. **if** process == "fitness":
522. pelanggaran = pelanggaran + 1
523. **elif** process == "improvement":
524. individu[id] = self.generate\_random\_shift()

527. pelanggaran\_total += pelanggaran
529. **if** debug:
530. **if** pelanggaran > 0:
531. **print** "[WORKING HOURS] PELANGGARAN BIDAN - %d" % (id)
532. **print** "Pelanggaran: %d" % (pelanggaran)
534. **if** debug:
535. **print** "[WORKING HOURS] TOTAL PELANGGARAN: %d" % (pelanggaran\_total)
537. **if** process == "fitness":
538. **return** pelanggaran\_total


542. **def** fitness(self, debug=False):
543. self.lingkungan\_individu\_fitness = []
544. self.lingkungan\_individu\_fitness\_interval = []
545. total\_fitness = 0
546. **for** individu **in** self.lingkungan\_individu:
547. normalisasi\_fitness = self.single\_fitness(individu)
548. total\_fitness += normalisasi\_fitness
549. self.lingkungan\_individu\_fitness.append(normalisasi\_fitness)

552. **if** debug:
553. **print**(json.dumps(self.lingkungan\_individu\_fitness, indent=4, sort\_keys=False))
555. **for** index **in** range(len(self.lingkungan\_individu\_fitness)):
556. self.lingkungan\_individu\_fitness\_interval.append({"awal": 0, "batas": 0})
557. **if** index-1 < 0:
558. prev\_key = 0
559. **else**:
560. prev\_key = index-1
561. batas\_prev = self.lingkungan\_individu\_fitness\_interval[prev\_key]["batas"]
562. awal = batas\_prev
563. batas = batas\_prev + (self.lingkungan\_individu\_fitness[index]/total\_fitness)
564. self.lingkungan\_individu\_fitness\_interval[index]["awal"] = awal
565. self.lingkungan\_individu\_fitness\_interval[index]["batas"] = batas

568. **def** single\_fitness(self, individu, debug=False):
570. self.temp\_total\_pelanggaran["min\_bidan"] = self.min\_bidan(individu, "fitness",
571. debug)
572. self.temp\_total\_pelanggaran["day\_off"] = self.day\_off(individu, "fitness", debug)
573. self.temp\_total\_pelanggaran["pairshift"] = self.pairshift\_overflow(individu,
574. "fitness", debug)
575. self.temp\_total\_pelanggaran["working\_hours"] = self.working\_hours(individu,
576. "fitness", debug)
578. fitness = 0
579. fitness += self.temp\_total\_pelanggaran["min\_bidan"] \* self.hard\_penalti
580. fitness += self.temp\_total\_pelanggaran["day\_off"] \* self.hard\_penalti
581. fitness += self.temp\_total\_pelanggaran["pairshift"] \* self.soft\_penalti
582. fitness += self.temp\_total\_pelanggaran["working\_hours"] \* self.hard\_penalti
583. normalisasi\_fitness = float(1) / (fitness+1)
584. **return** normalisasi\_fitness

587. **def** roulette\_wheel(self, rand\_number):
588. index = 0
589. **for** fitness **in** self.lingkungan\_individu\_fitness\_interval:
590. **if** rand\_number > fitness["awal"] **and** rand\_number <= fitness["batas"]:
591. **return** index
592. index += 1

595. **def** selection(self):
596. self.parents\_individu = []
597. **for** i **in** range(len(self.lingkungan\_individu)/2):
598. p1 = self.roulette\_wheel(random.uniform(0, 1))
599. p2 = self.roulette\_wheel(random.uniform(0, 1))
600. parent1 = copy.deepcopy(self.lingkungan\_individu[p1])
601. parent2 = copy.deepcopy(self.lingkungan\_individu[p2])
602. self.parents\_individu.append([parent1, parent2])

605. **def** recombination(self):
606. temp\_linkungan\_individu = []
607. **for** parent **in** self.parents\_individu:
608. rand\_val = random.uniform(0, 1)
609. **if** rand\_val <= self.probabilitas\_rekombinasi:
610. #REKOMBINASI ONE POINT COLUMN
611. rand\_col = random.randint(1, self.hari-1)
612. parent1 = {"slice1": [], "slice2": []}
613. parent2 = {"slice1": [], "slice2": []}
614. anak1, anak2 = {}, {}

617. **for** id, individu\_row **in** parent[0].items():
618. parent1["slice1"].append(individu\_row[0:rand\_col])
619. parent1["slice2"].append(individu\_row[rand\_col:])
621. **for** id, individu\_row **in** parent[1].items():
622. parent2["slice1"].append(individu\_row[0:rand\_col])
623. parent2["slice2"].append(individu\_row[rand\_col:])
625. index = 0
626. **for** id, bdn\_w\_sch **in** self.bidan\_w\_schedule.items():
627. **if** bdn\_w\_sch["officer"] == "SN" **or** bdn\_w\_sch["officer"] == "JR":
628. anak1[id] = parent1["slice1"][index] + parent2["slice2"][index]
629. anak2[id] = parent2["slice1"][index] + parent1["slice2"][index]
630. index += 1
632. temp\_linkungan\_individu.append(anak1)
633. temp\_linkungan\_individu.append(anak2)
634. **else**:
635. temp\_linkungan\_individu.append(parent[0])
636. temp\_linkungan\_individu.append(parent[1])
638. **del** self.lingkungan\_individu[:]
639. self.lingkungan\_individu = temp\_linkungan\_individu

642. **def** mutation(self):
644. **for** individu **in** self.lingkungan\_individu:
645. rand\_value = random.uniform(0, 1)
647. **if** rand\_value <= self.probabilitas\_mutasi:
648. individu\_id = individu.keys()
649. **for** i **in** range(2):
650. rand\_id = random.randint(0, len(individu\_id) - 1)
651. id = individu\_id[rand\_id]
652. individu[id] = self.generate\_random\_shift()


656. **def** local\_search(self):
657. **for** index, individu **in** enumerate(self.lingkungan\_individu):
658. rand\_value = random.uniform(0, 1)
660. **if** rand\_value < self.probabilitas\_local\_search:
662. before\_improve = self.single\_fitness(individu)
663. temp\_individu = copy.deepcopy(individu)
665. self.pairshift\_overflow(temp\_individu, "improvement")
667. self.working\_hours(temp\_individu, "improvement")
669. self.min\_bidan(temp\_individu, "improvement")
671. self.day\_off(temp\_individu, "improvement")
673. after\_improve = self.single\_fitness(temp\_individu)
675. **if** after\_improve > before\_improve:
676. **if** after\_improve == 1:
677. self.elit\_individu["fitness"] = after\_improve
678. self.elit\_individu["individu"] = temp\_individu
679. **return**
680. **else**:
681. self.lingkungan\_individu[index] = temp\_individu
683. self.fitness()
684. self.elitist()
686. **def** elitist(self):
687. i, value = max(enumerate(self.lingkungan\_individu\_fitness),
688. key=operator.itemgetter(1))
689. **if** value > self.elit\_individu["fitness"]:
690. self.elit\_individu["fitness"] = value
691. self.elit\_individu["individu"] = copy.deepcopy(self.lingkungan\_individu[i])
693. total\_remove = len(self.lingkungan\_individu) - self.populasi
694. total\_remove += 1
696. **for** i **in** range(total\_remove):
697. index, value = min(enumerate(self.lingkungan\_individu\_fitness),
698. key=operator.itemgetter(1))
699. **del** self.lingkungan\_individu[index]
700. **del** self.lingkungan\_individu\_fitness[index]
702. elit = copy.deepcopy(self.elit\_individu["individu"])
703. self.lingkungan\_individu.append(elit)


707. **def** termination(self, generasi):
709. elit\_fitness = self.single\_fitness(self.elit\_individu["individu"])
710. totalp = 0
711. **for** key, total **in** self.temp\_total\_pelanggaran.items():
712. totalp += total
714. end\_time = datetime.now()
715. elapsed = str(format(end\_time - self.start\_time))
717. **print** "%d. %f" % (generasi, elit\_fitness)
718. **print** "----%d. %f" % (generasi, self.elit\_individu["fitness"])
719. **print** "Total Pelanggaran: %d" % totalp
720. **for** jenis, total **in** self.temp\_total\_pelanggaran.items():
721. **print** "---%s: %d" % (jenis, total)
722. **print** "Waktu: %s" % elapsed
723. **print** " "
725. min\_fitness = min(self.lingkungan\_individu\_fitness)
726. max\_fitness = max(self.lingkungan\_individu\_fitness)
728. f = open('scheduling\_process.txt', 'r')
729. scheduling\_process = f.read()
730. msg = ""
731. **if** scheduling\_process == "false":
732. **print** "PROSESS STOPED FROM CLIENT"
733. **print** " "
734. msg = "Stopped from client"
735. **elif** min\_fitness == max\_fitness:
736. **print** "TERMINASI TERPENUHI - KONVERGENSI FITNESS"
737. **print** " "
738. msg = "Konvergensi fitness"
739. with open("scheduling\_process.txt", "wb") as fo:
740. fo.write("false")
741. **elif** self.elit\_individu["fitness"] == 1:
742. **print** "TERMINASI TERPENUHI - TIDAK ADA PELANGGARAN"
743. **print** " "
744. msg = "Fitness sempurna"
745. **elif** generasi < self.generasi-1:
746. **return** {"stop": False}
747. **elif** generasi == self.generasi-1:
748. **print** "TERMINASI TERPENUHI - MAKSIMAL GENERASI TERCAPAI"
749. **print** " "
750. msg = "Maksimal generasi tercapai"
751. with open("scheduling\_process.txt", "wb") as fo:
752. fo.write("false")
753. **elif** generasi > self.generasi-3:
754. unik = set(self.lingkungan\_individu\_fitness)
755. unik\_value = len(unik)
756. **print** "UNIK FITNESS: %d" % (unik\_value)
757. **return** {"stop": False}
759. result\_individu = {}
760. full\_individu = dict(self.individu\_static.items() +
761. self.elit\_individu["individu"].items())
762. **for** id, myindividu **in** full\_individu.items():
763. rest\_shift = self.bidan\_w\_schedule[id]["rest\_shift"]
764. result\_individu[id] = []
765. **if** rest\_shift != "CLEAR":
766. **for** index, shift **in** enumerate(rest\_shift):
767. **if** shift == "-":
768. result\_individu[id].append(full\_individu[id][index])
769. **else**:
770. result\_individu[id].append(shift)
771. **else**:
772. result\_individu[id] = full\_individu[id]

775. end\_time = datetime.now()
776. elapsed = str(format(end\_time - self.start\_time))

779. data = {"individu": result\_individu, "msg": msg, "generasi": generasi,
780. "elit\_fitness": self.elit\_individu["fitness"],
781. "pelanggaran": self.temp\_total\_pelanggaran, "total\_pelanggaran": totalp, "execution\_time": elapsed}
783. **return** {"stop": True, "data": data}


787. **def** generate\_pattern\_schedule(periode\_date, days):
788. periode\_db = Periode.query.filter(Periode.periode == periode\_date).first()
789. last\_periode = Periode.query.order\_by(Periode.periode.desc()).filter(Periode.periode < periode\_db.periode).first()
790. **if** last\_periode:
791. bidan\_schedule = Bidan.query \
792. .outerjoin(Schedules) \
793. .add\_columns(Schedules.shift, Schedules.rest\_shift, Bidan.officer, Bidan.id) \
794. .filter(Schedules.periode\_id == last\_periode.id).all()
796. bidan\_shift = {}
797. **for** sch **in** bidan\_schedule:
798. **if** sch.shift:
799. bidan\_shift[sch.id] = sch.shift.split(",")
800. **else**:
801. bidan\_shift[sch.id] = None
802. # print(json.dumps(last\_periode.id, indent=4, sort\_keys=False))
803. **for** id, shift **in** bidan\_shift.items():
804. **if** **not** shift:
805. temp = "CLEAR"
806. **else**:
807. index = len(shift) - 1
808. temp\_static = "CLEAR"
809. **if** shift[index] == "P":
810. back = 0
811. pg = 0
812. start = True
813. **while** start:
814. **if** index >= back:
815. **if** shift[index - back] == "P":
816. pg += 1
817. back += 1
818. **else**:
819. start = False
820. **else**:
821. start = False
823. **if** Bidan.query.get(id).officer == "KT" **or** Bidan.query.get(id).officer == "KR" **or** Bidan.query.get(id).officer == "persir":
824. pola\_pagi = 6
825. pola\_pagi = pola\_pagi - pg
826. rest = ["P" **for** i **in** range(pola\_pagi)]
827. rest.append("O")
828. temp\_static = ",".join(rest)
829. **print** temp\_static
830. **else**:
831. **if** pg >= 4:
832. temp = "O"
833. **else**:
834. temp = "CLEAR"
836. **elif** shift[index] == "S":
837. **if** index >= 1:
838. **if** shift[index - 1] == "S":
839. temp = "O"
840. **else**:
841. temp = "S,O"
842. **else**:
843. temp = "S,O"
844. **elif** shift[index] == "M":
845. **if** index >= 1:
846. **if** shift[index - 1] == "M":
847. temp = "O,O"
848. **else**:
849. temp = "M,O,O"
850. **else**:
851. temp = "M,O,O"
852. **elif** shift[index] == "O":
853. **if** index >= 1 **and** index >= 2:
854. **if** shift[index - 1] == "M" **and** shift[index - 2] == "M":
855. temp = "O"
856. **else**:
857. temp = "CLEAR"
858. **else**:
859. temp = "CLEAR"

862. **if** Bidan.query.get(id).officer == "KT" **or** Bidan.query.get(id).officer == "KR" **or** Bidan.query.get(id).officer == "persir":
863. temps = temp\_static
864. **else**:
865. temps = temp

868. **if** temps != "CLEAR":
869. temps\_arr = temps.split(",")
870. none\_rest = ["-" **for** i **in** range(days - len(temps\_arr))]
871. none\_rest\_str = ",".join(none\_rest)
872. temps = temps + "," + none\_rest\_str
874. Schedules.query \
875. .filter((Schedules.periode\_id == periode\_db.id) & (Schedules.bidan\_id == id)) \
876. .update({Schedules.rest\_shift: temps})
878. db.session.commit()
879. **return** True
880. **else**:
881. **return** False