IMPLEMENTASI *CONSTRAINT SATISFACTION PROBLEM* PADA APLIKASI PENJADWALAN PERKULIAHAN BERBASIS WEB DI JURUSAN ILMU KOMPUTER UNIVERSITAS LAMPUNG

(Skripsi)

Oleh

JIHAN HAYA MUFIALDO
2117051095

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

IMPLEMENTASI *CONSTRAINT SATISFACTION PROBLEM* PADA APLIKASI PENJADWALAN PERKULIAHAN BERBASIS WEB DI JURUSAN ILMU KOMPUTER UNIVERSITAS LAMPUNG

Oleh

JIHAN HAYA MUFIALDO

Skripsi

Sebagai Salah Satu Syarat untuk Memperoleh Gelar
SARJANA KOMPUTER

Pada

Program Studi S1 Ilmu Komputer
Jurusan Ilmu Komputer

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

ABSTRAK

IMPLEMENTASI *CONSTRAINT SATISFACTION PROBLEM* PADA APLIKASI PENJADWALAN PERKULIAHAN BERBASIS WEB DI JURUSAN ILMU KOMPUTER UNIVERSITAS LAMPUNG

Oleh

JIHAN HAYA MUFIALDO

Masalah penjadwalan perkuliahan merupakan tantangan kompleks yang telah banyak dikaji dalam bidang *Operations Research* (OR) dan *Artificial Intelligence* (AI). Salah satu pendekatan yang efektif untuk menyelesaikan permasalahan ini adalah *Constraint Satisfaction Problem* (CSP), dengan *Constraint Programming* sebagai salah satu paradigma pemrogramannya. Penelitian ini bertujuan untuk mengembangkan sistem penjadwalan perkuliahan berbasis CSP menggunakan *solver* CP-SAT dari Google OR-Tools. Penelitian ini dilakukan di Jurusan Ilmu Komputer Universitas Lampung pada Semester Ganjil Tahun Ajaran 2024/2025, dengan menggunakan data perkuliahan dari periode akademik 2023 Ganjil dan 2024 Genap. *Constraint* dalam model disesuaikan dengan aturan penyusunan jadwal yang berlaku, diformulasikan ke dalam model matematika, dan diimplementasikan dalam kode pemrograman Python. Model kemudian diselesaikan menggunakan *solver* CP-SAT, dan hasilnya dianalisis berdasarkan aspek validitas, kelayakan, optimalitas, serta waktu komputasi. Hasil pengujian menunjukkan bahwa model mampu menghasilkan jadwal yang memenuhi seluruh *constraint* tanpa pelanggaran. Waktu komputasi rata-rata untuk menghasilkan jadwal perkuliahan semester 2023 Ganjil dengan 264 kelas adalah 51,58 detik, sedangkan untuk semester 2024 Genap dengan 206 kelas adalah 31,18 detik. Dengan hasil ini, model yang dikembangkan terbukti andal dalam menghasilkan jadwal perkuliahan yang valid dan optimal, serta berpotensi untuk dikembangkan lebih lanjut guna meningkatkan fleksibilitas dan integrasi dalam sistem penjadwalan akademik.

Kata kunci: *Constraint Satisfaction Problem* (CSP), *Constraint Programming,* CP-SAT, OR-Tools, Penjadwalan Perkuliahan.

ABSTRACT

IMPLEMENTATION OF CONSTRAINT SATISFACTION PROBLEM IN A WEB-BASED COURSE SCHEDULING APPLICATION AT THE DEPARTMENT OF COMPUTER SCIENCE UNIVERSITY OF LAMPUNG

By

JIHAN HAYA MUFIALDO

The problem of course scheduling is a complex challenge that has been widely studied in the fields of Operations Research (OR) and Artificial Intelligence (AI). One of the effective approaches to solving this problem is the Constraint Satisfaction Problem (CSP), with Constraint Programming as one of its programming paradigms. This study aims to develop a CSP-based course scheduling system using the CP-SAT solver from Google OR-Tools. The research was conducted at the Department of Computer Science, University of Lampung, for the Odd Semester of the 2024/2025 Academic Year, utilizing course scheduling data from the 2023 Odd and 2024 Even academic periods. The constraints in the model were adapted to comply with the scheduling regulations in the department, formulated into a mathematical model, and implemented in Python programming code. The model was then solved using the CP-SAT solver, and the results were analyzed based on validity, feasibility, optimality, and computational time. The evaluation results show that the model successfully generates schedules that satisfy all constraints without violations. The average computation time to generate the course schedule for the 2023 Odd Semester, consisting of 264 classes, was 51.58 seconds, while for the 2024 Even Semester, with 206 classes, it was 31.18 seconds. These results demonstrate that the developed model is reliable in producing valid and optimal course schedules and has the potential for further development to enhance flexibility and integration within academic scheduling systems.

Keywords: Constraint Satisfaction Problem (CSP), Constraint Programming, Course Scheduling, CP-SAT, OR-Tools

Judul Skripsi	:	**IMPLEMENTASI *CONSTRAINT SATISFACTION PROBLEM* PADA APLIKASI PENJADWALAN PERKULIAHAN BERBASIS WEB DI JURUSAN ILMU KOMPUTER UNIVERSITAS LAMPUNG**
Nama Mahasiswa	:	*Jihan Haya Mufialdo*
Nomor Pokok Mahasiswa	:	2117051095
Program Studi	:	S1 Ilmu Komputer
Fakultas	:	Matematika dan Ilmu Pengetahuan Alam

MENYETUJUI

1. Komisi Pembimbing

Febi Eka Febriansyah, M.T.
NIP. 19800219 200604 1 001

2. Ketua Jurusan Ilmu Komputer

Dwi Sakethi, S.Si., M.Kom.
NIP. 19680611199802 1 001

MENGESAHKAN

1. Tim Penguji

Ketua : **Febi Eka Febriansyah, M.T.** .....

Penguji Pembahas I : **Didik Kurniawan, S.Si., M.T.** .....

Penguji Pembahas II : **Wartariyus, S.Kom., M.T.I.** .....

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Dr. Eng. Heri Satria, S.Si., M.Si.
NIP. 19711001 200501 1 002

Tanggal Lulus Ujian Skripsi: **19 Maret 2025**

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Jihan Haya Mufialdo

NPM : 2117051095

Menyatakan bahwa skripsi saya yang berjudul "**Implementasi *Constraint Satisfaction Problem* pada Aplikasi Penjadwalan Perkuliahan Berbasis Web di Jurusan Ilmu Komputer Universitas Lampung**" merupakan karya saya sendiri dan bukan karya orang lain. Semua tulisan yang tertuang di skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti skripsi saya merupakan hasil penjiplakan atau dibuat orang lain, maka saya bersedia menerima sanksi berupa pencabutan gelar yang telah saya terima.

Bandar Lampung, 19 Mei 2025

Penulis,

Jihan Haya Mufialdo

NPM. 2117051095

RIWAYAT HIDUP

Penulis dilahirkan di Pekanbaru pada tanggal 29 Januari 2003 sebagai anak pertama dari dua bersaudara, dari Bapak Yanra Mufialdo, S.Sos. dan Ibu Santi Dewi. Pendidikan yang ditempuh penulis diantaranya, menyelesaikan pendidikan dasar di SD Negeri 36 Pekanbaru pada tahun 2015, pendidikan menengah pertama di SMP Negeri 1 Pekanbaru pada tahun 2018, dan pendidikan menengah atas di SMA IT Insan Cendekia Payakumbuh pada tahun 2021. Perjalanan pendidikan penulis dilanjutkan dengan terdaftar sebagai mahasiswa Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung melalui jalur Seleksi Bersama Masuk Perguruan Tinggi (SBMPTN) pada tahun 2021.

Selama menjadi mahasiswa, penulis aktif mengikuti beberapa kegiatan antara lain:

1. Asisten Praktikum mata kuliah Dasar-Dasar Pemrograman dan Pemrograman Terstruktur pada tahun 2022 dan 2023.
2. Sekretaris Bidang Keilmuan Himpunan Mahasiswa Jurusan Ilmu Komputer (HIMAKOM) Universitas Lampung pada tahun 2023.
3. Anggota Tim Minat Bakat Robotik Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung sebagai anggota Tim Robot SAR pada tahun 2023.
4. MBKM Riset Independen di Jurusan Ilmu Komputer Universitas Lampung sebagai anggota penelitian dengan topik Penerapan *Convolutional Neural Networ*k untuk Pengenalan Wajah dalam Sistem Kehadiran Mahasiswa pada tahun 2023.
5. MBKM Magang Bersertifikat di Bakrie Center Foundation melalui program MSIB *batch* 6 dan 7 sebagai *IT Database* dan *IT Development* pada tahun 2024.

MOTTO

“Boleh jadi kamu membenci sesuatu, padahal ia amat baik bagimu, dan boleh jadi (pula) kamu menyukai sesuatu, padahal ia amat buruk bagimu; Allah mengetahui, sedang kamu tidak mengetahui.”
(Q.S. Al-Baqarah [2]: 216)

"... dan Allah adalah sebaik-baik perencana."
(QS. Ali Imran [3]: 54)

“You lose the right to complain about outcomes you never worked for.”
(Jihan H.M.)

“What is right is not always popular, and what is popular is not always right.”
(Albert Einstein)

PERSEMBAHAN

Alhamdulillahirabbil'alamin

Puji syukur kehadirat Allah Subhanahu Wa Ta'ala atas segala rahmat dan karunia-Nya sehingga skripsi ini dapat diselesaikan dengan sebaik-baiknya. Shalawat beriring salam selalu tercurahkan kepada junjungan Nabi Agung Muhammad Shallallahu 'Alaihi Wasallam.

Aku persembahkan karya ini kepada:

Kedua Orang Tuaku Tersayang
dan
Keluargaku Tercinta

Atas segala doa yang tak henti dipanjatkan, cinta yang tak pernah habis dibagikan, serta pengorbanan dan ketulusan dalam membimbing setiap langkah hidupku. Terima kasih telah menjadi sandaran, penyemangat, dan tempat pulang yang paling tulus sepanjang perjalanan ini.

Seluruh Keluarga Besar Ilmu Komputer 2021

Atas kebersamaan, semangat, dan dukungan yang tak ternilai sepanjang perjalanan perkuliahan ini.

Almamater Tercinta, Universitas Lampung dan Jurusan Ilmu Komputer

Tempat bernaung dan menimba ilmu untuk bekal kehidupan dunia dan akhirat.

SANWACANA

Puji syukur atas segala rahmat Allah SWT. Yang diberikan kepada penulis sehingga dapat menyelesaikan skripsi berjudul "IMPLEMENTASI *CONSTRAINT SATISFACTION PROBLEM* PADA APLIKASI PENJADWALAN PERKULIAHAN BERBASIS WEB DI JURUSAN ILMU KOMPUTER UNIVERSITAS LAMPUNG". Tidak lupa shalawat dan salam senantiasa dicurahkan kepada Nabi Muhammad SAW, suri teladan yang telah menunjukkan jalan yang benar kepada seluruh umatnya.

Pada kesempatan ini, penulis menyampaikan terima kasih kepada semua pihak yang telah memberikan bantuan dan dukungan, baik dalam penyusunan skripsi ini maupun selama perjalanan perkuliahan penulis secara keseluruhan, yaitu:

1. Kedua orang tua, Bunda dan Papa, serta adik penulis, Haqi, yang senantiasa mengajarkan kebaikan, mengusahakan segala kebutuhan, memberikan perhatian, dukungan, cinta dan kasih sayang, do'a terbaik, serta kepercayaan atas setiap keputusan yang diambil oleh penulis hingga detik ini.
2. Keluarga besar penulis, termasuk Nenek Sumiarti, Nenek Mufida, Bunga, Umi, Om Dedy, Om Kai, Aca, dan lainnya yang tidak dapat disebutkan satu per satu, atas segala dukungan, doa, dan kasih sayang yang telah diberikan.
3. Bapak Dr. Eng. Heri Satria, S.Si., M.Si. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.
4. Bapak Dwi Sakethi, S.Si., M.Kom. selaku Ketua Jurusan Ilmu Komputer Universitas Lampung.
5. Ibu Ossy Dwi Endah Wulansari, S.Si., M.T. selaku Dosen Pembimbing Akademik selama penulis menempuh perkuliahan.

6. Bapak Febi Eka Febriansyah, M.T. selaku Dosen Pembimbing Utama yang senantiasa memberikan bimbingan, arahan, kritik, serta saran yang bermanfaat kepada penulis baik dalam penyusunan skripsi maupun hal-hal di luar skripsi.
7. Bapak Didik Kurniawan, S.Si., M.T. selaku Dosen Pembahas sekaligus sosok pembimbing yang senantiasa memberikan bimbingan, arahan, kritik, dan saran yang berharga, baik dalam penyusunan skripsi maupun perjalanan akademik penulis secara keseluruhan.
8. Bapak Wartariyus, S.Kom., M.T.I. selaku Dosen Pembahas yang telah memberikan kritik, saran, serta masukan kepada penulis dalam penyusunan skripsi.
9. Ibu Anie Rose Irawati, S.T., M.Cs. yang senantiasa memberikan bimbingan dan arahan yang berharga selama penulis menempuh perkuliahan.
10. Bapak dan Ibu Dosen Jurusan Ilmu Komputer Universitas Lampung yang telah memberikan ilmu, pengetahuan, serta pengalaman terbaik selama penulis menjadi mahasiswa.
11. Seluruh Staf Jurusan Ilmu Komputer, termasuk Ibu Ade Nora Maela, yang telah membantu penulis dari awal hingga akhir masa perkuliahan.
12. Iqbal dan Cindy sebagai rekan penelitian penulis yang telah memberikan dukungan, kerjasama, serta ide-ide konstruktif yang sangat membantu dalam kelancaran dan keberhasilan proses penelitian ini.
13. Teman-teman seperjuangan, Ayuni, Shafa, Vidya, Abiyyi, Cindy, Billa, Roy, Ikhsan, Annisa, serta seluruh mahasiswa Jurusan Ilmu Komputer angkatan 2021 yang telah menemani dengan dukungan, motivasi, kekuatan, dan kebersamaan yang sangat berharga.
14. Teman-teman jarak jauh, Wafiy, Ara, Aem, dan Riesti yang tidak hanya menjadi tempat penulis berbagi cerita dan semangat, tetapi juga memberikan dukungan, motivasi, dan inspirasi yang berharga sejak masa sekolah hingga kini.
15. Teman-teman magang, Vania, Murti, Dimas A, Dimas H, Bara, serta seluruh mahasiswa magang CLP 8 dan 9, beserta para mentor yang telah membimbing dan membersamai penulis dalam proses belajar, berbagi pengalaman, dan bekerja sama selama masa magang.

16. Teman-teman KKN, Zuan, Abdan, Yazi, Zia, Isti, Vivi, serta warga Desa Rejomulyo Dusun 2 atas kerja sama dan kebersamaan yang terjalin selama masa tinggal dan pelaksanaan kegiatan di desa.
17. Seluruh pihak yang telah membantu penulis secara langsung maupun tidak langsung, atas dukungannya dalam menyelesaikan skripsi dan perkuliahan.

Penulis menyadari bahwa masih banyak kekurangan dalam penulisan skripsi ini. Akan tetapi, penulis berharap skripsi ini dapat membawa manfaat dan keberkahan bagi perkembangan ilmu pengetahuan terutama bagi civitas Ilmu Komputer Universitas Lampung.

Bandar Lampung, 19 Mei 2025

Jihan Haya Mufialdo
NPM. 2117051095

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Gambar Halaman

DAFTAR TABEL

Tabel Halaman

DAFTAR KODE

BAB I
PENDAHULUAN

1.1. Latar Belakang

Masalah penjadwalan telah menjadi fokus penelitian sejak tahun 1950-an, dimana berbagai pendekatan *Operation Research* (OR) dan *Artificial Intelligent* (AI) telah digunakan untuk menemukan solusi yang efektif (Hlaing & Nyo, 2019). Di antara berbagai jenis masalah penjadwalan, penjadwalan dalam dunia pendidikan menjadi salah satu yang paling rumit dan banyak dibahas oleh para peneliti. Secara umum, ada tiga jenis penjadwalan pendidikan, yaitu penjadwalan universitas, penjadwalan ujian, dan penjadwalan sekolah (Iqbal dkk., 2021). Secara kategoris, penjadwalan universitas dapat diklasifikasikan menjadi dua kategori utama, yaitu penjadwalan kelas dan penjadwalan ujian, masing-masing dengan seperangkat batasan dan persyaratan yang berbeda (Hlaing & Nyo, 2019). Penelitian ini berfokus pada masalah penjadwalan kelas, yang dikenal sebagai *University Course Timetabling Problem* (UCTP).

University Course Timetabling Problem (UCTP) melibatkan penugasan mata kuliah yang diambil oleh sekelompok mahasiswa, yang diajarkan oleh seorang atau beberapa dosen tertentu, ke sejumlah slot waktu yang terbatas di ruang kelas yang sesuai. Penugasan dilakukan dengan cara yang memastikan tidak ada konflik antara ruangan, mahasiswa, dan dosen, serta memenuhi berbagai persyaratan lain atau apa yang didefinisikan sebagai batasan atau *constraint* (Zaulir dkk., 2022). Selain itu, *University Course Timetabling Problem* (UCTP) adalah masalah optimasi kombinatorial yang bersifat NP-*hard*, yaitu tidak ada algoritma polinomial yang dikenal untuk menyelesaikan masalah ini (Kralev dkk., 2019). Hal ini menunjukkan bahwa jika semua kombinasi harus diperiksa, waktu untuk menemukan solusi pada masalah yang kompleks akan meningkat secara drastis

(Hlaing & Nyo, 2019). Untuk mendapatkan solusi yang optimal untuk masalah ini, semua opsi yang memungkinkan harus dipertimbangkan guna memilih solusi terbaik yang mampu memenuhi beragam batasan, preferensi, dan kebutuhan pihak-pihak terkait, serta dapat diselesaikan dalam waktu yang efisien.

Penjadwalan merupakan aktivitas administratif yang penting, rutin, dan kompleks di sebagian besar institusi akademik, namun hanya sedikit organisasi yang memiliki penyusun jadwal otomatis yang andal, dan bahkan lebih sedikit lagi yang sepenuhnya bebas dari intervensi manual (Hlaing & Nyo, 2019). Sementara itu, di banyak institusi pendidikan tinggi, sebagian besar tugas administratif seperti penerimaan, pendaftaran, dan pembayaran telah diotomatisasi secara signifikan. Namun, meski aspek lain telah mencapai otomatisasi, penjadwalan kelas tetap menjadi tantangan besar yang belum sepenuhnya diatasi oleh sistem otomatisasi (Vivian dkk., 2021). Hal ini juga terjadi di Jurusan Ilmu Komputer Universitas Lampung, dimana penjadwalan perkuliahan masih disusun secara manual oleh Sekretaris Jurusan dengan menggunakan bantuan Microsoft Excel, sebelum kemudian informasi tersebut dimasukkan ke dalam Sistem Informasi Akademik Universitas Lampung (Siakadu).

Penjadwalan kelas merupakan jenis masalah penugasan yang dapat diselesaikan menggunakan pendekatan *Constraint Satisfaction Problems* (CSP). *Constraint Satisfaction Problem* (CSP) adalah pendekatan untuk menyelesaikan sebuah masalah dengan menemukan keadaan atau objek yang memenuhi beberapa persyaratan atau kriteria. Pendekatan CSP dapat meningkatkan efisiensi dan efektivitas proses penjadwalan di institusi pendidikan. Sebagaimana ditunjukkan oleh penelitian oleh (Rosmasari dkk., 2019) dalam penelitiannya di Fakultas Teknologi Informasi (FKTI) Universitas Mulawarman (UNMUL) yang menerapkan algoritma CSP untuk mengoptimalkan proses penjadwalan mata kuliah, hasil penelitian tersebut menunjukkan bahwa penggunaan CSP dapat mengurangi bentrok jadwal dan meningkatkan efisiensi ruang serta waktu.

Salah satu paradigma pemrograman yang dapat diterapkan untuk menyelesaikan *Constraint Satisfaction Problem* (CSP) adalah *Constraint Programming* (CP), di mana pengguna menentukan batasan-batasan masalah dan sebuah *solver* digunakan untuk menyelesaikannya. Sebagai contoh, penelitian oleh (Canque & Pinto, 2024) di perusahaan penyedia bahan bakar pesawat mengembangkan sistem penjadwalan otomatis berbasis CP menggunakan CP-SAT *solver* dari Google OR-Tools. Penelitian ini menunjukkan efektivitas CP, dengan berhasil mengurangi waktu perencanaan shift karyawan dari dua jam menjadi hanya 0.036800 detik, sambil tetap memastikan distribusi shift yang adil. Hasil tersebut menegaskan potensi CP sebagai metode yang efisien dalam menangani masalah penjadwalan yang kompleks.

Berdasarkan uraian latar belakang yang ada maka penulis menetapkan untuk melakukan penelitian dengan judul "IMPLEMENTASI *CONSTRAINT SATISFACTION PROBLEM* PADA APLIKASI PENJADWALAN PERKULIAHAN BERBASIS WEB DI JURUSAN ILMU KOMPUTER UNIVERSITAS LAMPUNG".

1.2. Rumusan Masalah

Berdasarkan pemaparan pada latar belakang, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana penerapan pendekatan *Constraint Satisfaction Problem* (CSP) menggunakan *solver* CP-SAT dari *library* OR-Tools dengan bahasa pemrograman Python.
2. Bagaimana pengembangan *web service* dengan *framework* FastAPI untuk mengintegrasikan model *Constraint Satisfaction Problem* (CSP).

1.3. Batasan Masalah

Agar penelitian ini lebih terfokus, berikut merupakan batasan masalah dalam penelitian ini, yaitu:

1. Penelitian berfokus pada pemodelan kasus sebagai *Constraint Satisfaction Problem* (CSP) dan penyelesaiannya menggunakan *solver* yang disediakan oleh modul CP-SAT dari *library* OR-Tools pada Python.
2. Penelitian tidak mencakup penjelasan mengenai cara kerja *solver* dalam mendapatkan solusi.
3. *Constraint* atau batasan yang digunakan pada penelitian ini didasarkan pada aturan penjadwalan yang diterapkan oleh Jurusan Ilmu Komputer Universitas Lampung.
4. Studi kasus dan data yang digunakan adalah jadwal perkuliahan di Jurusan Ilmu Komputer Universitas Lampung T.A. 2023/2024.
5. Hasil penelitian berupa *web service* yang dibangun menggunakan *framework* FastAPI.

1.4. Tujuan Penelitian

Tujuan penelitian ini adalah mengembangkan *web service* dengan FastAPI yang mengimplementasikan pendekatan *Constraint Satisfaction Problem* (CSP) menggunakan *solver* CP-SAT dari *library* OR-Tools.

1.5. Manfaat Penelitian

Dengan dilakukannya penelitian ini, diharapkan dapat memberikan manfaat di antaranya:

1. Meningkatkan efisiensi dalam penyusunan jadwal perkuliahan secara otomatis di Jurusan Ilmu Komputer Universitas Lampung, mengurangi intervensi manual, dan meminimalkan kesalahan penjadwalan.
2. Memberikan solusi *web service* yang fleksibel dan mudah diintegrasikan dengan aplikasi penjadwalan berbasis web.
3. Menyediakan referensi yang bermanfaat bagi mahasiswa dan peneliti lain dalam penerapan pendekatan *Constraint Satisfaction Problem* (CSP) dan pengembangan *web service* untuk aplikasi di bidang pendidikan dan manajemen akademik.

BAB II
TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Penelitian terdahulu bertujuan untuk membandingkan penelitian yang sudah ada dengan penelitian yang akan dilakukan. Selain itu, penelitian terdahulu akan dijadikan sebagai acuan dalam upaya tinjauan pustaka terkait dengan penelitian yang dilakukan. Penelitian tersebut antara lain.

2.1.1. Penjadwalan Kuliah Otomatis dengan *Constraint Programming*

Penelitian oleh (Muhyi, 2008) dengan judul Penjadwalan Kuliah Otomatis dengan *Constraint Programming* membahas penerapan *Constraint Programming* (CP) untuk memodelkan dan menyelesaikan masalah penjadwalan kuliah di STMIK Supra. Dalam penelitian ini, pemodelan menggunakan bahasa pemrograman Java dan alat bantu Choco untuk mengimplementasikan CP. Masalah penjadwalan dimodelkan sebagai *Constraint Satisfaction Problem* (CSP), dengan tujuan mengatasi keterbatasan sumber daya, seperti jumlah kelas dan dosen, serta waktu yang tersedia untuk mengajar. Pemodelan ini memungkinkan penyusunan penjadwalan dengan cara mendefinisikan batasan-batasan yang relevan, sehingga tidak perlu memodelkan solusi secara manual dan men-*detail*.

Hasil penelitian menunjukkan bahwa penerapan CP menghasilkan proses penjadwalan yang lebih efisien dan cepat, serta memberikan solusi optimal terhadap keterbatasan yang ada. Penjadwalan dapat dilakukan secara otomatis dengan mempertimbangkan berbagai batasan yang didefinisikan, dan

implementasi dalam sistem informasi akademik memberikan kinerja yang memadai serta dapat digunakan berulang kali. Kontribusi penelitian ini terletak pada pengembangan sistem penjadwalan yang lebih responsif terhadap kebutuhan akademik dan membuka peluang penerapan metode CP pada pemodelan sistem lainnya.

2.1.2. Implementasi Algoritma *Constraint Satisfaction Problems* pada Sistem Penjadwalan Mata Kuliah

Penelitian oleh (Rosmasari dkk., 2019) dengan judul Implementasi Algoritma *Constraint Satisfaction Problems* pada Sistem Penjadwalan Mata Kuliah membahas penerapan algoritma *Constraint Satisfaction Problems* (CSP) dalam penjadwalan mata kuliah di Fakultas Teknologi Informasi (FKTI) Universitas Mulawarman (UNMUL). Tujuan utama dari penelitian ini adalah untuk mengoptimalkan proses penjadwalan yang sebelumnya dilakukan secara manual, yang sering kali menghasilkan jadwal yang tidak efisien dan menyebabkan kelelahan bagi dosen.

Hasil penelitian menunjukkan bahwa dengan menggunakan algoritma CSP, penjadwalan mata kuliah dapat dioptimalkan menjadi 5 slot mata kuliah 3 SKS dengan total waktu 12 jam 30 menit per minggu, ditambah 1 slot ekstra mata kuliah 2 SKS dengan waktu 1 jam 40 menit. Total optimasi waktu yang diperoleh adalah 14 jam 10 menit per minggu. Penelitian ini menyimpulkan bahwa penjadwalan menggunakan algoritma CSP lebih efektif dibandingkan dengan metode manual, karena dapat mengurangi kemungkinan terjadinya bentrok jadwal dan meningkatkan efisiensi penggunaan ruang dan waktu.

2.1.3. *A Constraint Programming-Based System for Shift Scheduling in A Fuel Supply Company*

Penelitian oleh (Canque & Pinto, 2024) dengan judul *A Constraint Programming-Based System for Shift Scheduling in A Fuel Supply Company* berfokus pada pengembangan model *Constraint Programming* untuk diimplementasikan dalam sistem ARGOS, yang merupakan aplikasi web yang sudah ada sebelumnya untuk penjadwalan *shift* di perusahaan penyedia bahan bakar. Dengan memodelkan masalah penjadwalan sebagai *Constraint Satisfaction Problem* (CSP), penelitian ini bertujuan untuk meningkatkan efisiensi dan keadilan dalam proses penjadwalan *shift*. Penggunaan CP-SAT *solver* dari Google OR-Tools memungkinkan sistem ARGOS untuk secara otomatis menghasilkan jadwal *shift* yang memenuhi berbagai kendala, seperti hari libur karyawan dan batasan *shift*, serta mengurangi waktu yang diperlukan untuk merencanakan *shift* dari sekitar dua jam menjadi hanya 0.036800 detik.

Melalui integrasi model *Constraint Programming* ini, ARGOS tidak hanya dapat mempercepat proses penjadwalan, tetapi juga memastikan distribusi *shift* yang lebih adil di antara karyawan. Penelitian ini menunjukkan bahwa penerapan *Constraint Programming* dalam sistem yang sudah ada dapat memberikan solusi yang lebih efektif dan terstruktur dalam manajemen sumber daya manusia, serta meningkatkan kinerja sistem penjadwalan yang ada. Dengan demikian, hasil penelitian ini memberikan kontribusi signifikan terhadap pengembangan sistem penjadwalan yang lebih efisien dan responsif terhadap kebutuhan perusahaan.

2.2. Uraian Landasan Teori

Berikut ini beberapa teori yang berkaitan dengan penelitian yang dilakukan.

2.2.1. Penjadwalan

Penjadwalan merupakan suatu petunjuk atau indikasi apa saja yang harus dilakukan, dengan siapa, dan dengan peralatan apa yang digunakan untuk menyelesaikan suatu pekerjaan pada waktu tertentu (Melayanti dkk., 2019). Penjadwalan adalah aktivitas yang biasanya dilakukan di institusi pendidikan atau institusi lainnya. Penjadwalan dapat didefinisikan sebagai masalah yang memiliki empat parameter, yaitu sekumpulan waktu (T), sekumpulan sumber daya (R), sekumpulan pertemuan (M), dan sekumpulan kendala (C) (Muklason dkk., 2019). Contoh penjadwalan dapat dilihat pada Gambar 1 dan 2.

Mata Kuliah	Kelas
Logika	A
Logika	B
Matematika	A
Matematika	B
Pemrograman	A
Pemrograman	B
Aljabar Linear	A
Aljabar Linear	B
Bahasa Inggris	A
Bahasa Inggris	B

Hari
Senin
Selasa
Rabu
Kamis
Jumat

Ruangan
R1
R2

Sesi
1
2

Gambar 1. Contoh Kumpulan Data yang Akan Diolah Menjadi Jadwal.

Hari	Sesi	R1	R2
Senin	1	Pemrograman A	Logika B
	2		
Selasa	1	Matematika B	Matematika A
	2	Aljabar Linear A	
Rabu	1	Pemrograman B	
	2	Logika A	Aljabar Linear B
Kamis	1		Bahasa Inggris A
	2		Bahasa Inggris B
Jumat	1		
	2		

Gambar 2. Contoh Jadwal Perkuliahan yang Dihasilkan.

2.2.2. *Constraint Satisfaction Problem* (CSP)

Menurut (Tsang, 1993), *Constraint Satisfaction Problem* (CSP) adalah masalah yang terdiri dari himpunan variabel terbatas, di mana setiap variabel terkait dengan himpunan domain terbatas, serta sekumpulan kendala atau *constraint* yang membatasi nilai-nilai yang dapat diambil oleh variabel-variabel tersebut secara bersamaan. Tujuannya adalah menetapkan sebuah nilai untuk setiap variabel dengan mematuhi semua kendala. *Constraint Satisfaction Problem* (CSP) memiliki tiga komponen, yaitu variabel (X), domain (D), dan *constraint* (C) (Hlaing & Nyo, 2019), dimana:

1. $X = \{x_1, \ldots, x_n\}$ adalah himpunan variabel yang disebut variabel domain.
2. $D = \{D_1, \ldots, D_n\}$ adalah himpunan domain (nilai-nilai yang mungkin untuk variabel yang bersangkutan).
3. $C = \{C_1, \ldots, C_n\}$ adalah himpunan *constraint* (hubungan yang didefinisikan pada sebagian dari semua variabel).

2.2.2.1. Variabel

Variabel adalah elemen yang dapat diisi dengan nilai dari himpunan yang memenuhi batasan tertentu. Terdapat berbagai jenis variabel, seperti variabel numerik, *boolean*, dan simbolik, tergantung pada tipe nilai dalam domainnya.

2.2.2.2. Domain

Domain adalah himpunan semua nilai yang mungkin dapat diberikan kepada sebuah variabel. Jika x adalah sebuah variabel, maka D_x digunakan untuk menunjukkan domain dari variabel tersebut.

1. Ketika domain hanya berisi angka, variabel tersebut disebut variabel numerik.
   Contoh: $D_x = \{1,2,3, \dots\}$.
2. Ketika domain hanya berisi nilai *boolean*, variabel tersebut disebut variabel *boolean*.
   Contoh: $D_x = \{0,1\}$
3. Ketika domain berisi tipe objek yang disebutkan secara khusus, variabel tersebut disebut variabel simbolik.
   Contoh: $D_x = \{Senin, Selasa, Rabu, Kamis, Jumat\}$

2.2.2.3. *Constraint*

Constraint adalah suatu aturan atau batasan terhadap nilai-nilai yang dapat diambil oleh variabel secara bersamaan. *Constraint* dapat dibedakan menjadi dua jenis, yaitu *hard constraint* dan *soft constraint* (Fatchurrochman dkk., 2023).

1. *Hard constraint* adalah *constraint* yang harus dipenuhi untuk mendapatkan solusi. Pelanggaran *hard constraint* akan menyebabkan solusi menjadi tidak layak untuk diterapkan di dunia nyata.

2. *Soft constraint* adalah *constraint* yang pemenuhannya akan meningkatkan kualitas dari solusi. Pelanggaran *soft constraint* masih dapat diterima dan akan memengaruhi nilai fungsi objektif.

2.2.3. OR-Tools pada Python

2.2.3.1. Python

Python merupakan salah satu bahasa pemrograman yang banyak digunakan oleh perusahaaan besar maupun para *developer* untuk mengembangkan berbagai macam aplikasi berbasis *desktop*, web dan *mobile* (Romzi & Kurniawan, 2020). Python merupakan bahasa pemrograman yang bersifat gratis dan *open source*, sehingga mudah diakses dan dapat digunakan oleh banyak pengguna. Python juga memiliki pustaka yang sangat luas, mencakup berbagai *library* dan fungsi yang mendukung beragam bidang aplikasi (Shakila dkk., 2023).

2.2.3.2. OR-Tools

OR-Tools adalah perangkat lunak *open source* untuk optimasi kombinatorial, yang bertujuan untuk menemukan solusi terbaik dari sekumpulan besar kemungkinan solusi. Salah satu potensi keuntungan dari Google OR-Tools adalah, meskipun dikembangkan dalam C++, ia memungkinkan pemodelan masalah dalam berbagai bahasa pemrograman, salah satunya Python. OR-Tools mencakup *solver* untuk *Constraint Programming, Linear Programming, Mixed-Integer Programming,* dan lainnya.

2.2.3.3. *Constraint Programming*

Constraint Programming adalah paradigma yang digunakan dalam penyelesaian masalah pencarian kombinatorial yang didasarkan pada berbagai teknik kecerdasan buatan, ilmu komputer, basis data, bahasa pemrograman, dan riset operasi (Canque & Pinto, 2024). *Constraint Programming* unggul dalam hal pemodelan karena tidak bergantung pada "bagaimana" atau "cara" menemukan solusi. Dengan *Constraint Programming*, masalah didekati dan dicari solusinya cukup hanya dengan menspesifikasikan batasan-batasan yang harus diselesaikan (Muhyi, 2008). *Constraint Programming* lebih didasarkan pada kelayakan (menemukan solusi yang layak) daripada optimasi (menemukan solusi optimal) dan fokus pada batasan dan variabel daripada fungsi objektif.

2.2.3.4. CP-SAT *Solver*

CP-SAT *Solver* pada Google OR-Tools merupakan *solver* utama untuk *Constraint Programming* yang dirancang untuk menyelesaikan masalah dengan kendala kompleks secara efisien. CP-SAT *Solver* bekerja dengan memodelkan masalah ke dalam variabel, domain, dan *constraint*. CP-SAT menggunakan pendekatan berbasis SAT (*Boolean Satisfiability*) untuk mengevaluasi kombinasi variabel dan domain dengan cepat, serta mengidentifikasi solusi yang memenuhi kendala. Proses ini melibatkan teknik pencarian seperti *propagation*, *backtracking*, dan *conflict analysis* untuk mempersempit ruang solusi secara efisien. Jika solusi ditemukan, hasilnya diekstraksi dan dipetakan ke dalam format sesuai kebutuhan, seperti jadwal. Jika tidak ada solusi, model dapat diperbaiki atau kendala disesuaikan, lalu *solver* dijalankan kembali.

Berikut langkah penerapan CP-SAT *Solver* untuk membuat penjadwalan dengan kasus seperti pada Gambar 1.

a. Inisialisasi Model

Objek model CP-SAT dibuat dengan menggunakan *library* dari Google OR-Tools. Model ini akan menampung variabel, domain, dan *constraint*.

Contoh Kode:

```
# Import library
from ortools.sat.python import cp_model

# Buat objek model
model = cp_model.CpModel()
```

b. Definisi Variabel dan Domain

Langkah ini melibatkan definisi variabel yang akan digunakan dalam model dan penetapan domainnya. Domain adalah himpunan nilai yang mungkin diambil oleh variabel.

Contoh Kode:

```
# Data
mataKuliah = ['Logika','Matematika','Dasar-Dasar
Pemrograman','Aljabar Linear','Bahasa
Inggris']
kelas = ['A','B']
hari = ['Senin','Selasa','Rabu','Kamis','Jumat']
sesi = [1,2]
ruangan = ['R1','R2']

total_mataKuliah = len(mataKuliah)
total_kelas = len(kelas)
total_hari = len(hari)
total_sesi = len(sesi)
total_ruangan = len(ruangan)

kelas_mataKuliah = [f'{mk} {k}' for mk in mataKuliah
for k in kelas]
total_kelas = len(kelas_mataKuliah)

# Definisi variabel
jadwal = {}
for k in range(total_kelas):
  for h in range(total_hari):
    for s in range(total_sesi):
      for r in range(total_ruangan):
        jadwal[(k, h, s, r)] =
        model.NewBoolVar(f'jadwal_c{k}_d{h}_s{s}_r{r}')
```

c. Penambahan *Constraint*

Constraint yang harus dipenuhi oleh variabel ditambahkan pada langkah ini. *Constraint* ini membatasi nilai yang dapat diambil oleh variabel berdasarkan hubungan tertentu.

Contoh Kode:

```
# Tambahkan constraint
# Constraint: Setiap kelas harus dijadwalkan di satu
# tempat dan waktu
for k in range(total_kelas):
    model.Add(sum(jadwal[(k, h, s, r)]
    for h in range(total_hari) for s in range(total_sesi)
    for r in range(total_ruangan)) == 1)

# Constraint: Tidak ada dua kelas yang dapat
# dijadwalkan pada waktu dan ruang yang sama
for h in range(total_hari):
    for s in range(total_sesi):
        for r in range(total_ruangan):
            model.Add(sum(jadwal[(k, h, s, r)]
            for k in range(total_kelas)) <= 1)

# Constraint: Kelas A dan B tidak boleh bentrok satu
# sama lain
for h in range(total_hari):
    for s in range(total_sesi):
        model.Add(sum(jadwal[(k, h, s, r)]
        for k in range(total_kelas) for r in range(total_ruangan)
        if k % 2 == 0) <= 1)
        model.Add(sum(jadwal[(k, h, s, r)]
        for k in range(total_kelas) for r in range(total_ruangan)
        if k % 2 == 1) <= 1)
```

d. Penambahan Fungsi Objektif

Fungsi Objektif yang mendefinisikan tujuan optimasi yang ingin dicapai ditambahkan pada langkah ini. Fungsi objektif ini akan memberikan nilai numerik untuk setiap solusi yang mungkin, dan solusi yang memiliki nilai fungsi objektif terbaik (nilai minimum atau maksimum) akan dianggap sebagai solusi optimal.

Contoh Kode:

```
# Fungsi objektif: Minimalkan jumlah kelas pada hari Jumat
total_kelas_j = 0
for k in range(total_kelas):
    for s in range(total_sesi):
```

```
        for r in range(total_ruangan):
            total_kelas_j +=
            sum(jadwal[(k, h, s, r)] for h in [4])
model.Minimize(total_kelas_j)
```

e. Pemecahan Masalah

Setelah model selesai dibuat dengan semua variabel, *constraint*, dan fungsi objektif, *solver* dapat dijalankan untuk mencari solusi.

Contoh Kode:

```
# Buat solver
solver = cp_model.CpSolver()

# Jalankan solver
status = solver.Solve(model)
```

f. Evaluasi Solusi

Setelah *solver* selesai, evaluasi solusi yang ditemukan perlu dilakukan. Evaluasi solusi dilakukan dengan memeriksa status dari *solver* yang dijalankan. Status yang dikembalikan oleh *solver* dapat dilihat pada Tabel 1.

Tabel 1. Status pada CP-SAT *solver*.

Status	Deskripsi
OPTIMAL	Solusi yang ditemukan layak dan optimal
FEASIBLE	Solusi yang ditemukan layak, tetapi tidak diketahui apakah solusi tersebut optimal
INFEASIBLE	Solusi yang ditemukan tidak layak
UNKNOWN	Tidak ada solusi yang ditemukan

Tabel 1 memberikan gambaran tentang status yang dihasilkan oleh CP-SAT *solver* setelah proses penyelesaian masalah. Setiap status mencerminkan keberhasilan

atau kegagalan dalam menemukan solusi yang sesuai dengan batasan yang diberikan.

Contoh Kode:

```
# Evaluasi solusi
if status == cp_model.OPTIMAL or status == cp_model.FEASIBLE:
  # Ekstraksi Hasil
else:
  print('Tidak ada solusi yang ditemukan.')
```

g. Ekstraksi Hasil

Jika solusi ditemukan, nilai variabel dapat diekstraksi untuk digunakan sesuai dengan kebutuhan dan konteks masalah yang sedang diselesaikan. Proses ini memungkinkan hasil yang dihasilkan oleh *solver* untuk diterapkan secara langsung atau dimanfaatkan sebagai dasar pengambilan keputusan.

Contoh Kode:

```
# Ekstraksi Hasil
for k in range(total_kelas):
    for h in range(total_hari):
      for s in range(total_sesi):
        for r in range(total_ruangan):
          if solver.Value(jadwal[(k, h, s, r)]) == 1:
            print(f'{kelas_mataKuliah[k]}: {hari[h]},
            {sesi[s]}, {ruangan[r]}')
```

Berdasarkan langkah penerapan di atas, didapatkan hasil sebagai berikut:

```
Logika A: Rabu, 2, R1
Logika B: Senin, 1, R2
Matematika A: Selasa, 1, R2
Matematika B: Selasa, 1, R1
Pemrograman A: Senin, 1, R1
Pemrograman B: Rabu, 1, R1
Aljabar Linear A: Selasa, 2, R1
Aljabar Linear B: Rabu, 2, R2
Bahasa Inggris A: Kamis, 1, R2
Bahasa Inggris B: Kamis, 2, R2
```

Gambar 3. *Output* Terminal.

Gambar 3 merupakan *output* yang ditampilkan pada terminal setelah ekstraksi hasil. *Output* ini dapat divisualisasikan seperti pada Gambar 2.

2.2.4. *Web Service*

Web service adalah teknologi yang memungkinkan aplikasi dan sistem berbeda untuk saling berinteraksi dan berkomunikasi melalui jaringan internet (Gunawan & Soetanto, 2023). *Web service* mampu menukar data tanpa memandang sumber database, bahasa yang digunakan, dan pada platform apa data tersebut dipakai. *Web service* memiliki ciri khusus berupa URL seperti web, namun yang membedakannya adalah interaksi yang diberikan, dimana URL *web service* hanya berisi sekumpulan informasi, perintah, dan konfigurasi yang berfungsi untuk membangun fungsi aplikasi tertentu (Wagito, 2022).

2.2.5. FastAPI

FastAPI merupakan salah satu *library* yang dapat digunakan dalam membangun *web service*. FastAPI adalah *framework* web Python yang dikembangkan oleh Sebastian Ramirez pada tahun 2018. *Framework* ini dirancang untuk menjadi alternatif yang lebih cepat dan lebih efisien dibandingkan dengan Flask dalam pengembangan layanan REST API. Keunggulan FastAPI terletak pada kemampuannya untuk menangani permintaan secara asinkron, yang meningkatkan performa aplikasi, terutama dalam konteks layanan yang memerlukan respons cepat dan efisiensi tinggi (Kornienko dkk., 2021).

BAB III
METODOLOGI PENELITIAN

3.1. Waktu dan Tempat Penelitian

Penelitian dilakukan pada Semester Ganjil Tahun Ajaran 2024/2025 di lingkungan Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.

3.2. Perangkat Penelitian

Untuk menunjang pelaksanaan penelitian ini, dibutuhkan perangkat keras dan perangkat lunak sebagai berikut.

3.2.1. Perangkat Keras

Perangkat keras yang digunakan dalam penelitian ini adalah sebuah laptop dengan spesifikasi sebagai berikut.

1. *System Manufacturer* : HP
2. *System Model* : HP ZBook 15 G5
3. *Processor* : Intel® Core™ i7-8850H
4. *Graphic* : Intel® UHD Graphics 630
5. *Memory* : 16.0 GB RAM
6. *Storage* : SSD 256 GB

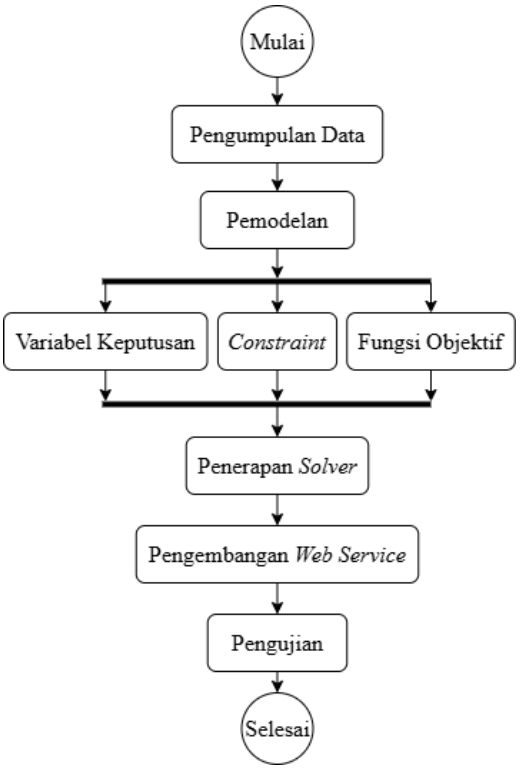
3.2.2. Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini adalah sebagai berikut.

1. Sistem Operasi Windows 11 Pro 64-bit
2. *Web Browser* Google Chrome dan Microsoft Edge
3. Visual Studio Code
4. Python dengan *library* OR-Tools dan FastAPI
5. Postman
6. GitHub

3.3. Tahapan Penelitian

Alur dari tahapan yang akan dilakukan pada penelitian ini diilustrasikan pada Gambar 4.



Gambar 4. Tahapan Penelitian.

Penelitian dilakukan dalam lima tahap, yaitu Pengumpulan Data, Pemodelan, Penerapan *Solver*, Pengembangan *Web Service*, dan Pengujian.

3.3.1. Pengumpulan Data

Pengumpulan data yang diperlukan untuk mendukung penelitian ini didapatkan dari dua sumber, yaitu wawancara dan observasi. Data yang dikumpulkan akan digunakan pada tahap pemodelan.

3.3.1.1. Wawancara

Pada tahap ini, dilakukan wawancara dengan pihak-pihak yang terlibat langsung dalam proses penyusunan jadwal perkuliahan di Jurusan Ilmu Komputer Universitas Lampung. Wawancara dilakukan dengan Sekretaris Jurusan yang bertanggung jawab atas penjadwalan. Tujuan dari wawancara ini adalah untuk mengidentifikasi kendala, aturan, serta kebutuhan spesifik yang menjadi pedoman dalam pembuatan jadwal secara otomatis.

Dari wawancara tersebut, beberapa informasi penting diidentifikasi, antara lain:

1. Kendala Ruangan dan Rasio Kelas: Jumlah ruangan yang tersedia belum memadai untuk menampung seluruh kelas yang ada, terutama ketika ada peningkatan jumlah mahasiswa. Akibatnya, beberapa kelas terpaksa dilaksanakan secara daring (*online*) untuk mengakomodasi keterbatasan kapasitas ruangan. Masalah ini menjadi batasan dalam penelitian, sehingga perlu diperhatikan dalam proses penjadwalan otomatis.
2. Kombinasi Dosen Penanggung Jawab dan Dosen Anggota: Penjadwalan menjadi lebih kompleks ketika melibatkan dosen penanggung jawab (PJ) dan dosen anggota. Kedua jenis dosen ini tidak boleh memiliki jadwal yang bentrok atau berada dalam sesi yang sama untuk mata kuliah berbeda. Hal ini

memerlukan penyesuaian tambahan dalam proses penjadwalan agar dapat memenuhi aturan internal jurusan.

3. Aturan dan *Constraint* Penjadwalan: Hasil wawancara digunakan untuk mengidentifikasi daftar *constraint* yang harus dipenuhi dalam penjadwalan. *Constraint* tersebut meliputi aturan umum serta aturan khusus yang berlaku dalam penjadwalan perkuliahan di Jurusan Ilmu Komputer.

Informasi yang diperoleh dari wawancara ini akan menjadi dasar dalam perancangan sistem penjadwalan otomatis, khususnya dalam menetapkan *constraint* yang harus dipenuhi untuk menghasilkan jadwal yang sesuai dengan kebutuhan dan kondisi aktual di Jurusan Ilmu Komputer.

3.3.1.2. Observasi

Pada tahap observasi, data dikumpulkan secara manual dari Sistem Informasi Akademik Universitas Lampung (Siakadu) dengan menggunakan akun mahasiswa dari program studi S1 Ilmu Komputer dan D3 Manajemen Informatika. Data yang dikumpulkan mencakup data kurikulum, periode akademik, mata kuliah, program studi pengampu, nama kelas, dosen pengajar, jadwal mingguan, serta kapasitas kelas. Data ini diperoleh melalui menu Perkuliahan > Kelas & Jadwal > Kelas Kuliah. Pada menu tersebut, filter Periode Akademik akan diatur pada nilai 2023 Ganjil dan 2023 Genap, sementara filter Kurikulum akan diatur pada nilai 2020. Perlu dicatat bahwa data jadwal yang melibatkan kelas MBKM tidak akan dimasukkan dalam pengumpulan ini. Selanjutnya, informasi mengenai semester setiap mata kuliah akan diperoleh melalui menu Perkuliahan > MK & Kurikulum > Kurikulum Prodi.

Selain mengumpulkan data jadwal perkuliahan, observasi juga dilakukan untuk memperoleh informasi mengenai data ruangan yang tersedia di Jurusan Ilmu Komputer. Data ruangan ini mencakup jenis ruangan (kelas teori atau laboratorium), kapasitas, serta fasilitas pendukung lainnya. Informasi tersebut akan

digunakan untuk memastikan alokasi ruangan yang optimal dalam penjadwalan perkuliahan.

Pada tahap ini, data yang dikumpulkan meliputi:

a. Data Dosen

Tabel 2. Data Dosen Pengampu Mata Kuliah.

Jenis Dosen	Jumlah
Dosen Jurusan Ilmu Komputer	22
Dosen Lain	16
Total	**38**

Data jumlah dosen yang mengampu mata kuliah di Jurusan Ilmu Komputer telah dikumpulkan dan disajikan dalam Tabel 2. Di tabel ini, dapat dilihat rincian jumlah dosen berdasarkan kategori yang ada, dengan total keseluruhan 38 dosen. Untuk detail data, termasuk nama dan informasi lain dari masing-masing dosen, dapat dilihat pada Lampiran 2.

b. Data Mata Kuliah

Tabel 3. Data Mata Kuliah Semester Ganjil.

Program Studi	Semester	Kelompok	Jumlah Mata Kuliah
S1 Ilmu Komputer	1	Wajib	13
		Peminatan	0
	3	Wajib	8
		Peminatan	4
	5	Wajib	6

Program Studi	Semester	Kelompok	Jumlah Mata Kuliah
		Peminatan	5
	7	Wajib	1
		Peminatan	7
D3 Manajemen Informatika	1	Wajib	12
		Peminatan	0
	3	Wajib	9
		Peminatan	0
	5	Wajib	6
		Peminatan	0
Total			**71**

Data mengenai jumlah mata kuliah yang diselenggarakan pada semester ganjil ditampilkan dalam Tabel 3. Tabel ini mengklasifikasikan mata kuliah berdasarkan program studi (S1 Ilmu Komputer dan D3 Manajemen Informatika), semester (1, 3, 5, dan 7), serta kelompok mata kuliah, yaitu mata kuliah wajib dan mata kuliah peminatan. Dari tabel tersebut, terlihat total 71 mata kuliah ditawarkan pada semester ganjil. Rincian lengkap mengenai setiap mata kuliah, termasuk kode mata kuliah, nama mata kuliah, jumlah SKS, dan nama dosen pengampu, disajikan dalam Lampiran 3 dan 5.

Tabel 4. Data Mata Kuliah Semester Genap.

Program Studi	Semester	Kelompok	Jumlah Mata Kuliah
S1 Ilmu Komputer	2	Wajib	9
		Peminatan	0
	4	Wajib	6
		Peminatan	4
	6	Wajib	1
		Peminatan	7
D3 Manajemen Informatika	2	Wajib	9
		Peminatan	0
	4	Wajib	8

Program Studi	Semester	Kelompok	Jumlah Mata Kuliah
		Peminatan	0
Total			**44**

Data mengenai jumlah mata kuliah yang diselenggarakan pada semester genap ditampilkan dalam Tabel 4. Tabel ini mengklasifikasikan mata kuliah berdasarkan program studi (S1 Ilmu Komputer dan D3 Manajemen Informatika), semester (2, 4, dan 6), serta kelompok mata kuliah, yaitu mata kuliah wajib dan mata kuliah peminatan. Dari tabel tersebut, terlihat total 44 mata kuliah ditawarkan pada semester genap. Rincian lengkap mengenai setiap mata kuliah, termasuk kode mata kuliah, nama mata kuliah, jumlah SKS, dan nama dosen pengampu, disajikan dalam Lampiran 4 dan 6.

c. Data Kelas

Tabel 5. Data Kelas Semester Ganjil.

Program Studi	Kelompok	Jumlah Kelas
S1 Ilmu Komputer	Wajib	88
	Peminatan	25
D3 Manajemen Informatika	Wajib	28
	Peminatan	0
Total		**141**

Data jumlah kelas yang diselenggarakan pada semester ganjil ditampilkan dalam Tabel 5. Tabel ini mengklasifikasikan jumlah kelas berdasarkan program studi (S1 Ilmu Komputer dan D3 Manajemen Informatika) dan kelompok mata kuliah, yaitu wajib dan peminatan. Total kelas yang dibuka pada semester ganjil berjumlah 141 kelas. Informasi detail mengenai setiap kelas, termasuk nama kelas, kapasitas kelas, dan informasi lain yang relevan, disajikan dalam Lampiran 7 dan 9.

Tabel 6. Data Kelas Semester Genap.

Program Studi	Kelompok	Jumlah Kelas
S1 Ilmu Komputer	Wajib	59
	Peminatan	23
D3 Manajemen Informatika	Wajib	18
	Peminatan	0
Total		**100**

Data jumlah kelas yang diselenggarakan pada semester genap ditampilkan dalam Tabel 6. Tabel ini mengklasifikasikan jumlah kelas berdasarkan program studi (S1 Ilmu Komputer dan D3 Manajemen Informatika) dan kelompok mata kuliah, yaitu wajib dan peminatan. Total kelas yang dibuka pada semester genap berjumlah 100 kelas. Informasi detail mengenai setiap kelas, termasuk nama kelas, kapasitas kelas, dan informasi lain yang relevan, disajikan dalam Lampiran 8 dan 10.

d. Data Ruangan

Tabel 7. Data Ruangan.

Ruangan	Kapasitas
GIK L1 A	50
GIK L1 B	50
GIK L1 C	100
GIK L2	100
LAB R1	25
LAB R2	25
LAB R3	25
LAB R4	20
LAB RPL	20
MIPA T L1 A	80
MIPA T L1 B	80

Data lengkap mengenai kapasitas ruangan yang digunakan untuk kegiatan perkuliahan disajikan dalam Tabel 7. Tabel ini mencantumkan nama ruangan dan kapasitas masing-masing ruangan yang tersedia di Jurusan Ilmu Komputer.

e. Data Sesi Perkuliahan

Tabel 8. Data Sesi Perkuliahan.

Sesi	Mulai	Selesai
1	7:30	9:10
2	9:20	11:00
3	11:10	12:50
4	13:30	15:10
5	15:30	17:00

Data sesi perkuliahan yang disajikan dalam Tabel 8 memberikan informasi mengenai alokasi waktu untuk setiap sesi perkuliahan yang berlaku di Jurusan Ilmu Komputer T.A. 2023/2024.

3.3.2. Pemodelan

Tahap pemodelan bertujuan untuk menerjemahkan data yang telah dikumpulkan menjadi model matematika yang dapat diselesaikan secara komputasi. Proses iteratif ini meliputi tiga langkah utama, yaitu pendefinisian variabel keputusan, *constraint*, dan fungsi objektif. Model matematika yang telah diformulasikan kemudian diimplementasikan dalam kode Python menggunakan *library* OR-Tools untuk dieksekusi oleh *solver*.

3.3.2.1. Variabel Keputusan

Variabel keputusan merepresentasikan penugasan sebuah kelas (yang diajar oleh seorang atau dua orang dosen) ke sebuah hari, sesi, dan ruangan tertentu.

Variabel keputusan direpresentasikan sebagai variabel *boolean*:

$$S_{cldsr}$$

Variabel ini memiliki arti sebagai berikut:

- $S_{cldsr} = 1$, jika kelas c yang diajar oleh dosen l dijadwalkan pada hari d, sesi s, dan ruangan r.
- $S_{cldsr} = 0$, jika kelas c yang diajar oleh dosen l tidak dijadwalkan pada hari d, sesi s, dan ruangan r.

Dimana:

- $cl \in \{"Matematika - Mr.X", "Logika - Mr.Y", .., CL\}$ adalah himpunan semua pasangan kelas-dosen.
- $d \in \{"Senin", "Selasa", .., D\}$ adalah himpunan semua hari yang tersedia untuk penjadwalan.
- $s \in \{1,2, .., S\}$ adalah himpunan semua sesi yang tersedia dalam sehari.
- $r \in \{"GIK\ L1A", "GIK\ L1B", .., R\}$ adalah himpunan semua ruangan yang tersedia.

3.3.2.2. *Constraint*

Constraint yang digunakan dalam pemodelan *Constraint Satisfaction Problem* (CSP) pada penelitian ini diperoleh dari hasil wawancara dengan Sekretaris Jurusan Ilmu Komputer. Berikut daftar *constraint* yang harus dipenuhi:

a. Setiap kelas teori atau responsi hanya dijadwalkan satu kali pada satu ruangan yang memenuhi kriteria kapasitas atau ruang *online*.

Setiap kelas yang bertipe "Teori" atau "Responsi" hanya boleh dijadwalkan tepat satu kali dalam seminggu. Penjadwalan ini harus dilakukan pada satu kombinasi hari, sesi, dan ruangan yang memenuhi kriteria. Kriteria ruangan yang dimaksud adalah tipe ruangan (harus "Kelas" atau "Online") dan kapasitas ruangan harus mencukupi kapasitas kelas.

$\forall cl \in CL | jenis(cl) \in \{Teori, Responsi\}$:

$$\sum_{d \in D} \sum_{s \in S} \sum_{r \in R | jenis(r) \in \{Kelas, Online\} \wedge kap(r) \geq kap(cl)} S_{cldsr} = 1$$

b. Setiap kelas praktikum hanya boleh dijadwalkan satu kali pada satu ruangan laboratorium.

Setiap kelas yang bertipe "Praktikum" hanya boleh dijadwalkan tepat satu kali dalam seminggu. Penjadwalan ini harus dilakukan pada satu kombinasi hari, sesi, dan ruangan yang tipenya adalah "Lab" (Laboratorium).

$\forall cl \in CL | jenis(cl) = Praktikum$:

$$\sum_{d \in D} \sum_{s \in S} \sum_{r \in R | jenis(r) = Lab} S_{cldsr} = 1$$

c. Dua kelas praktikum yang sama harus dijadwalkan pada satu waktu yang sama.

Jika ada dua *kelas praktikum* yang mengampu kelas yang sama, maka kedua kelas tersebut harus dijadwalkan pada hari dan sesi yang sama. Ini berarti satu kelas praktikum akan dilaksanakan secara paralel di dua laboratorium.

$\forall cl_1, cl_2 \in CL \mid kelas(cl_1) = kelas(cl_2) \wedge cl_1 \neq cl_2 :$
$\forall d \in D, \forall s \in S :$
$$\sum_{r \in R \mid jenis(r)=Lab} S_{cl_1 dsr} = \sum_{r \in R \mid jenis(r)=Lab} S_{cl_2 dsr} \leq 1$$

d. Setiap ruangan baik kelas maupun lab hanya boleh dijadwalkan untuk satu kelas pada satu sesi.

Pada setiap hari dan sesi, setiap ruangan (baik itu kelas maupun laboratorium) hanya boleh digunakan oleh paling banyak satu kelas. Tidak boleh ada dua kelas atau lebih yang dijadwalkan pada ruangan yang sama di waktu yang bersamaan.

$$\forall d \in D, \forall s \in S, \forall r \in R : \sum_{cl \in CL} S_{cldsr} \leq 1$$

e. Tidak boleh ada perkuliahan di hari Jumat sesi 3.

Tidak ada kelas yang boleh dijadwalkan pada hari Jumat di sesi ke-3. Ini berlaku untuk semua jenis kelas (teori, responsi, praktikum) dan semua ruangan.

$$\forall cl \in CL : \sum_{r \in R} S_{cl,Jumat,3,r} = 0$$

f. Setiap dosen hanya boleh mengajar satu kelas teori atau responsi di satu sesi yang sama.

Seorang dosen tidak boleh mengajar lebih dari satu kelas teori atau responsi pada sesi yang sama. Ini berlaku untuk setiap hari dan setiap sesi. Jika seorang dosen terdaftar sebagai dosen utama atau dosen pendamping untuk beberapa kelas teori atau responsi, maka hanya satu dari kelas-kelas tersebut yang dapat dijadwalkan pada sesi yang sama.

$\forall l \in L, \forall d \in D, \forall s \in S$:

$$\sum_{cl \in CL | l = dosen(cl) \wedge jenis(cl) \in \{Teori, Responsi\}} \sum_{r \in R} S_{cldsr} \leq 1$$

Dimana, $l \in \{"Mr.X", "Mr.Y", .., L\}$ adalah himpunan semua dosen yang mengampu mata kuliah.

g. Setiap dosen hanya boleh mengajar dua kelas praktikum di satu sesi yang sama.

Seorang dosen tidak boleh mengajar lebih dari dua kelas praktikum yang dilaksanakan secara paralel pada sesi yang sama. Ini berlaku untuk setiap hari dan setiap sesi. Jika seorang dosen terdaftar sebagai dosen utama atau dosen pendamping untuk beberapa kelas praktikum, maka paling banyak dua (kelas praktikum yang sama) dari kelas-kelas tersebut yang dapat dijadwalkan pada sesi yang sama.

$\forall l \in L, \forall d \in D, \forall s \in S$:

$$\sum_{cl \in CL | l = dosen(cl) \wedge jenis(cl) = Praktikum} \sum_{r \in R} S_{cldsr} \leq 2$$

Dimana: $l \in \{"Mr.X", "Mr.Y", .., L\}$ adalah himpunan semua dosen yang mengampu mata kuliah.

h. Setiap dosen tidak boleh mengajar kelas responsi atau teori dan kelas praktikum secara bersamaan di satu sesi yang sama.

Seorang dosen tidak boleh mengajar kelas teori/responsi dan kelas praktikum pada sesi yang sama. Dengan asumsi bahwa setiap kelas praktikum memiliki dua kelas paralel yang harus dijadwalkan bersamaan, maka jika seorang dosen mengajar

kelas praktikum, dosen itu tidak bisa lagi mengajar kelas teori/responsi pada sesi yang sama.

$\forall l \in L, \forall d \in D, \forall s \in S$:

$$\sum_{cl \in CL | l = dosen(cl) \wedge jenis(cl) \in \{Teori, Responsi\}} \sum_{r \in R} S_{cldsr} + \sum_{cl \in CL | l = dosen(cl) \wedge jenis(cl) = Praktikum} \sum_{r \in R} S_{cldsr} \leq 2$$

Dimana: $l \in \{"Mr.X", "Mr.Y", .., L\}$ adalah himpunan semua dosen yang mengampu mata kuliah.

i. Setiap kelas (mahasiswa) hanya boleh memiliki satu jadwal kelas teori atau responsi di satu sesi yang sama.

Setiap kelas (mahasiswa) untuk masing-masing semester, hanya boleh memiliki paling banyak satu jadwal kelas teori atau responsi pada sesi yang sama. Ini mencegah bentrokan kelas teori atau responsi untuk tiap mahasiswa dengan kelas tertentu.

$\forall sem \in SEM, \forall kel \in KEL, \forall d \in D, \forall s \in S$:

$$\sum_{cl \in CL | kelasMhs(cl) = kel \wedge semester(cl) = sem \wedge jenis(cl) \in \{Teori, Responsi\}} \sum_{r \in R} S_{cldsr} \leq 1$$

Dimana: $sem \in \{1,2,..,SEM\}$ adalah himpunan semester perkuliahan yang ada dan $kel \in \{"A", "B", .., KEL\}$ adalah himpunan kelompok kelas perkuliahan yang ada.

j. Setiap kelas (mahasiswa) hanya boleh memiliki dua jadwal kelas praktikum di satu sesi yang sama.

Setiap kelas (mahasiswa) untuk masing-masing semester, hanya boleh memiliki paling banyak dua jadwal kelas praktikum yang sama (yang dilaksanakan secara paralel di dua ruangan) pada sesi yang sama. Ini mencegah bentrokan kelas praktikum untuk tiap mahasiswa dengan kelas tertentu.

$$\forall sem \in SEM, \forall kel \in KEL, \forall d \in D, \forall s \in S :$$

$$\sum_{cl \in CL \mid l = kelasMhs(cl) = kel \wedge semester(cl) = sem \wedge jenis(cl) = Praktikum} \sum_{r \in R} S_{cldsr} \leq 2$$

k. Setiap kelas (mahasiswa) tidak boleh memiliki jadwal kelas responsi atau teori dan kelas praktikum secara bersamaan di satu sesi yang sama.

Setiap kelas (mahasiswa) untuk masing-masing semester, tidak boleh memiliki jadwal kelas teori/responsi dan kelas praktikum pada sesi yang sama. Dengan asumsi bahwa setiap kelas praktikum memiliki dua kelas paralel yang harus dijadwalkan bersamaan, maka jika kelas (mahasiswa) memiliki jadwal kelas praktikum, kelas (mahasiswa) itu tidak bisa lagi memiliki jadwal kelas teori/responsi pada sesi yang sama.

$$\forall sem \in SEM, \forall kel \in KEL, \forall d \in D, \forall s \in S :$$

$$\sum_{cl \in CL \mid l = kelasMhs(cl) = kel \wedge semester(cl) = sem \wedge jenis(cl) \in \{Teori, Responsi\}} \sum_{r \in R} S_{cldsr} +$$

$$\sum_{cl \in CL \mid l = kelasMhs(cl) = kel \wedge semester(cl) = sem \wedge jenis(cl) = Praktikum} \sum_{r \in R} S_{cldsr} \leq 2$$

l. Setiap dosen hanya boleh memiliki jadwal mengajar maksimal tiga sesi kelas teori dalam satu hari.

Seorang dosen tidak boleh dijadwalkan untuk mengajar lebih dari tiga sesi kelas teori dalam satu hari.

$\forall l \in L, \forall d \in D$:

$$\sum_{cl \in CL \mid l = dosen(cl) \wedge jenis(cl) = Teori} \sum_{s \in S} \sum_{r \in R} S_{cldsr} \leq 3$$

3.3.2.3. Fungsi Objektif

Tujuan paling mendasar dari *Constraint Satisfaction Problem* (CSP) adalah menemukan solusi yang *feasible* (memenuhi semua *hard constraint*). Akan tetapi, fungsi objektif dapat diterapkan untuk memilih solusi yang terbaik jika terdapat beberapa solusi *feasible*. Berikut beberapa fungsi objektif yang dapat diterapkan:

a. Minimalkan Selisih Kapasitas Ruangan dengan Kapasitas Kelas (Kesesuaian Kapasitas).

Fungsi objektif ini bertujuan untuk meminimalkan perbedaan antara kapasitas ruangan dan kapasitas kelas untu kelas teori atau responsi. Tujuannya adalah agar ruangan yang digunakan tidak terlalu besar atau terlalu kecil dibandingkan dengan jumlah mahasiswa dalam kelas.

$$Minimize \sum_{cl \in CL \mid jenis(cl) \in \{Teori, Responsi\}} \sum_{d \in D} \sum_{s \in S} \sum_{r \in R} |kapasitas(r) - kapasitas(cl)| \cdot S_{cldsr}$$

b. Maksimalkan Penggunaan LAB R1 dan R2 (Prioritas Lab).

Fungsi objektif ini bertujuan untuk memaksimalkan penggunaan ruangan LAB R1 dan R2 untuk kelas praktikum. Tujuannya adalah agar laboratorium yang lebih baik dapat digunakan secara maksimal.

$$Maximize \sum_{cl \in CL | jenis(cl) = Praktikum} \sum_{d \in D} \sum_{s \in S} \sum_{r \in R | jenis(r) = Lab \wedge nama(r) \in \{LAB\ R1, LABR2\}} S_{cldsr}$$

3.3.3. Penerapan *Solver*

Setelah model matematika diformulasikan, tahap selanjutnya adalah implementasi dan pencarian solusi menggunakan *solver* CP-SAT dari *library* OR-Tools. Model *Constraint Satisfaction Problem* (CSP) direpresentasikan dalam kode Python menggunakan kelas *CpModel()*. Objek *CpModel()* ini menyimpan informasi tentang variabel keputusan, domain, dan *constraint* yang telah didefinisikan dalam model matematika. Kemudian, objek *CpSolver()* digunakan untuk mengeksekusi algoritma pencarian solusi pada model yang telah dibangun. *CpSolver()* akan mencari nilai untuk variabel-variabel keputusan yang memenuhi semua *constraint*. Hasil pencarian ini kemudian dianalisis untuk mendapatkan solusi penjadwalan yang valid.

3.3.4. Pengembangan *Web Service*

Pengembangan *web service* dalam penelitian ini akan menggunakan *framework* FastAPI, yang dipilih karena kemudahan integrasinya dengan berbagai modul eksternal serta kemampuannya untuk menangani *request* secara asinkron. *Web service* ini dirancang untuk berfungsi sebagai antarmuka antara Aplikasi

Penjadwalan Perkuliahan Berbasis Web dan model *Constraint Satisfaction Problem* (CSP). *Web service* ini bertugas untuk:

1. Menerima *request* pembuatan jadwal dari aplikasi web melalui *endpoint* /api/generate-schedule pada *port* 8000.
2. Mengambil data yang dibutuhkan dari *database* melalui *endpoint* /api/public pada *port* 3000.
3. Memproses data tersebut menggunakan model CSP dan *solver* CP-SAT.
4. Mengirimkan hasil penjadwalan dalam format JSON (*JavaScript Object Notation*) ke *database* melalui *endpoint* /api/public/schedule pada *port* 3000.
5. Mengirimkan konfirmasi atau *response* ke aplikasi web.

3.3.5. Pengujian

Pengujian dilakukan untuk memastikan bahwa model, solusi, dan *web service* yang telah dibuat memenuhi kriteria yang diinginkan. Pengujian yang dilakukan terdiri dari dua aspek, yaitu pengujian model dan pengujian *web service*.

3.3.5.1. Pengujian Model

Pengujian model dilakukan untuk memvalidasi model penjadwalan yang telah dibangun. Pengujian ini melibatkan lima aspek uji. Pertama, *constraint validation* atau validasi kendala dilakukan untuk memastikan bahwa setiap *constraint* dalam model telah diimplementasikan dengan benar dan bekerja sesuai yang diharapkan. Proses ini dilakukan dengan menguji *constraint* satu per satu untuk memastikan tiap *constraint* valid. Kedua, uji *feasibility* atau kelayakan bertujuan untuk memastikan bahwa solusi yang dihasilkan oleh model memenuhi seluruh *constraint* yang telah didefinisikan. Evaluasi keberhasilan model didasarkan pada status yang dikembalikan oleh *solver.* Ketiga, uji *optimality* atau optimalitas bertujuan untuk mengevaluasi bagaimana masing-masing bobot fungsi objektif mempengaruhi kualitas solusi yang dihasilkan. Model akan dijalankan dengan

variasi bobot yang berbeda untuk setiap komponen fungsi objektif. Keempat, uji waktu komputasi bertujuan untuk mengukur efisiensi model dalam menghasilkan solusi. Model dijalankan untuk beberapa ukuran data *input*, dan waktu komputasi yang dibutuhkan untuk menemukan solusi dicatat. Terakhir, *output validation* atau validasi solusi dilakukan untuk memastikan bahwa solusi yang dihasilkan oleh model benar-benar memenuhi setiap *constraint* yang telah ditetapkan. Proses ini dilakukan dengan memeriksa *output* jadwal secara sistematis dan mendeteksi adanya pelanggaran terhadap aturan penjadwalan yang telah didefinisikan.

3.3.5.2. Pengujian *Web Service*

Pengujian *web service* dilakukan untuk memastikan bahwa semua *endpoint* yang telah dikembangkan berfungsi dengan baik dan memenuhi kebutuhan sistem penjadwalan kuliah. Pengujian ini dilakukan dengan menggunakan aplikasi Postman untuk mengirimkan berbagai permintaan HTTP ke setiap *endpoint* yang disediakan oleh *web service*. Setiap permintaan yang dikirimkan akan diuji berdasarkan skenario yang telah ditentukan, termasuk pengujian untuk berbagai parameter yang relevan. Hasil respon yang diterima dari *web service* akan dicatat dan dibandingkan dengan respon yang diharapkan, untuk memastikan bahwa semua fungsionalitas berjalan sesuai dengan yang diinginkan. Skenario pengujian dapat dilihat pada Tabel 9.

Tabel 9. Skenario Pengujian *Web Service*.

Endpoint	**Kasus Uji**	**Respon yang Diharapkan**
/api/generate-schedule	Mengirimkan data lengkap dan valid pada *endpoint* dengan parameter yang nilainya valid	Data jadwal perkuliahan
/api/generate-schedule	Mengirimkan data dengan *ClassLecturer* kosong	Tidak ada hasil jadwal perkuliahan
/api/generate-schedule	Mengirimkan data dengan *ScheduleDay* kosong	Tidak ada hasil jadwal perkuliahan

Endpoint	**Kasus Uji**	**Respon yang Diharapkan**
/api/generate-schedule	Mengirimkan data dengan *ScheduleSession* kosong	Tidak ada hasil jadwal perkuliahan
/api/generate-schedule	Mengirimkan data dengan *Room* kosong	Tidak ada hasil jadwal perkuliahan
/api/generate-schedule	Mengirimkan data ke *endpoint* tanpa parameter	*Error* data tidak dapat diproses (422)
/api/generate-schedule	Mengirimkan data ke *endpoint* dengan parameter tanpa nilai	*Error* data tidak dapat diproses (422)
/api/generate-schedule	Mengirimkan data ke *endpoint* dengan parameter yang nilainya tidak ditemukan	Tidak ada hasil jadwal perkuliahan
/api/generate-schedule	Mengirimkan data ke *endpoint* dengan parameter yang tipe datanya tidak sesuai	*Error* data tidak dapat diproses (422)
/api/generate-schedule	Mengirimkan *request* dengan metode yang salah	*Error* metode tidak diizinkan (405)

BAB V
SIMPULAN DAN SARAN

5.1. Simpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, dapat ditarik beberapa simpulan sebagai berikut:

1. Penelitian ini telah berhasil menerapkan pendekatan *Constraint Satisfaction Problem* (CSP) menggunakan *solver* CP-SAT dari *library* OR-Tools dengan bahasa pemrograman Python untuk menyelesaikan masalah penjadwalan perkuliahan, dengan rata-rata waktu yang dibutuhkan untuk menghasilkan jadwal kuliah untuk satu jurusan di satu periode akademik dengan data sebanyak 264 kelas selama 51,58 detik dan data sebanyak 206 kelas selama 31,18 detik.
2. Waktu komputasi yang dibutuhkan untuk memproses jadwal kuliah dengan *Constraint Satisfaction Problem* (CSP) menggunakan *solver* CP-SAT dipengaruhi oleh jumlah dan jenis *constraint* yang diterapkan, fungsi objektif yang digunakan, serta ukuran data *input* yang diproses. Semakin banyak *constraint* yang diterapkan, semakin rumit fungsi objektif, dan semakin besar data yang diproses, maka semakin lama waktu yang diperlukan untuk menghasilkan jadwal.
3. Penelitian ini juga telah berhasil mengembangkan *web service* dengan *framework* FastAPI yang dapat mengintegrasikan model CSP yang telah dibangun dengan aplikasi *client*. *Web service* ini menyediakan *endpoint* yang dapat diakses untuk melakukan permintaan penjadwalan dengan parameter yang dibutuhkan.

5.2. Saran

Meskipun penelitian ini telah mencapai tujuan yang diharapkan, terdapat beberapa potensi pengembangan lebih lanjut yang dapat dipertimbangkan:

1. Model CSP yang telah dibangun dapat ditingkatkan dengan menambahkan *constraint* yang lebih kompleks, seperti preferensi dosen terhadap jadwal atau ruangan tertentu.
2. Mengembangkan model penjadwalan asisten untuk kelas praktikum dan responsi yang terintegrasi dengan model CSP yang sudah ada. Integrasi ini bertujuan untuk menghasilkan jadwal akhir setelah data perkuliahan asisten dosen diperoleh.
3. Mengembangkan mekanisme konfigurasi yang memungkinkan aturan dan parameter pada *constraint* dapat diubah secara fleksibel melalui file konfigurasi, tanpa perlu melakukan perubahan langsung pada kode sumber. Hal ini akan meningkatkan fleksibilitas sistem dalam menyesuaikan kebijakan penjadwalan sesuai dengan kebutuhan yang berubah.
4. Mengembangkan sistem penjadwalan dengan antarmuka pengguna yang intuitif untuk diintegrasikan dengan model CSP melalui *web service* yang telah dikembangkan.

DAFTAR PUSTAKA

Canque, G. E., & Pinto, R. V. (2024). A constraint programming-based system for shift scheduling in a fuel supply company. *Revista Chilena de Ingeniería*, *32*(6). https://doi.org/10.4067/S0718-33052024000100206

Fatchurrochman, Afandi, A. N., Arifin, M. Z., & Mahmudy, W. Fi. (2023). Algoritma Penanganan Constraint pada Persoalan Penjadwalan Perkuliahan Universitas di Lingkungan Pendidikan Tinggi Keagamaan Islam (PTKI). *JEPIN (Jurnal Edukasi Dan Penelitian Informatika)*, *9*(2).

Gunawan, H., & Soetanto, H. (2023). ANALISA DAN IMPLEMENTASI WEB SERVICE MENGGUNAKAN METODE RESTFUL API PADA APLIKASI PEMINJAMAN ASET. *Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)*, *2*(2).

Hlaing, N. N., & Nyo, M. T. (2019). Constructing University Timetable using CSP Model. *International Journal of Research Available*, *6*(9), 824–828. https://journals.pen2print.org/index.php/ijr/

Iqbal, Z., Ilyas, R., Chan, H. Y., & Ahmed, N. (2021). Effective Solution of University Course Timetabling using Particle Swarm Optimizer based Hyper Heuristic approach. *Baghdad Science Journal*, *18*(4), 1465–1475. https://doi.org/10.21123/bsj.2021.18.4(Suppl.).1465

Kornienko, D. V., Mishina, S. V., Shcherbatykh, S. V., & Melnikov, M. O. (2021). Principles of securing RESTful API web services developed with python frameworks. *Journal of Physics: Conference Series*, *2094*(3). https://doi.org/10.1088/1742-6596/2094/3/032016

Kralev, V., Kraleva, R., & Kumar, S. (2019). *A Modified Event Grouping Based Algorithm for the University Course Timetabling Problem*. *9*(1).

Melayanti, S., Mukhtar, H., & Fuad, E. (2019). APLIKASI PENJADWALAN OTOMATIS UJIAN PROPOSAL DAN SIDANG SKRIPSI PADA FAKULTAS ILMU KOMPUTER UNIVERSITAS MUHAMMADIYAH RIAU. *JURNAL FAKULTAS ILMU KOMPUTER*, *8*(1), 315–333.

Muhyi, Y. (2008). *PENJADWALAN KULIAH OTOMATIS DENGAN CONSTRAINT PROGRAMMING*. https://www.researchgate.net/publication/319122545

Muklason, A., Irianti, R. G., & Marom, A. (2019). Automated Course Timetabling Optimization Using Tabu-Variable Neighborhood Search Based Hyper-Heuristic Algorithm. *Procedia Computer Science*, *161*, 656–664.

Novria, R., Kurniawan, B., & Suryanto. (2022). Aplikasi Pemesanan Makanan Di Bebek dan Ayam Tekaeng Menggunakan Php dan Mysql. *Jurnal Informatika Dan Komputer (JIK)*, *13*(1).

Ovan, & Saputra, A. (2020). *CAMI: Aplikasi Uji Validitas dan Reliabilitas Instrumen Penelitian Berbasis Web* (S. A. Ansari, Ed.). Yayasan Ahmar Cendekia Indonesia. https://books.google.co.id/books?id=mZgMEAAAQBAJ&lpg=PA1&ots=Yl GsfzxBoP&dq=cami%20aplikasi%20uji%20validitas%20dan%20reliabilitas %20instrumen%20penelitian%20berbasis%20web&lr&pg=PR4#v=onepage &q=cami%20aplikasi%20uji%20validitas%20dan%20reliabilitas%20instru men%20penelitian%20berbasis%20web&f=false

Romzi, M., & Kurniawan, B. (2020). PEMBELAJARAN PEMROGRAMAN PYTHON DENGAN PENDEKATAN LOGIKA ALGORITMA. *JTIM: Jurnal Teknik Informatika Mahakarya*, *3*(2), 37–44.

Rosmasari, Dengen, N., & Chandra, F. (2019). IMPLEMENTASI ALGORITMA CONSTRAINT SATISFACTION PROBLEMS PADA SISTEM PENJADWALAN MATA KULIAH. *Jurnal Ilmu Pengetahuan Dan Teknologi Komputer (JITK)*, *4*(2), 169–176. http://www.unmul.ac.id/

SEVIMA. (2020, November 8). *Apa Itu SIAKAD Online dan Apa Tujuan SIAKAD?* Https://Sevima.Com/Apa-Itu-Siakad-Online-Dan-Apa-Tujuan-Siakad/.

Shakila, M., Akmal Aoulia, P., Harson, P. S., Kurniawan, F. M., & Rosyani, P. (2023). Analasis Penerapan Python Dengan Perhitungan Pertidaksamaan. *NEWTON : Jurnal Matematika, Fisika, Algoritma Dan Sains*, *1*(1), 29–33. https://ojs.jurnalmahasiswa.com/ojs/index.php/newtoon

Tsang, E. (1993). *FOUNDATIONS OF CONSTRAINT SATISFACTION*. Academic Press Limited.

Vivian, N. C., Achimugu, P., Nwufoh, C., Achimugu, P., Achimugu, O., & Chollom, T. (2021). *A HARD CONSTRAINT SATISFACTION PROBLEM (HCSP) ALGORITHM FOR UNIVERSITY COURSE TIME TABLING*. https://www.researchgate.net/publication/361283868

Wagito. (2022). IMPLEMENTASI WEB SERVICE WAKTU SHALAT BERBASIS TEKNOLOGI PAAS CLOUD COMPUTING. *Technologia*, *13*(4). http://praytimes.org,

Wei, M., Xu, Z., & Hu, J. (2021). Entity Relationship Extraction Based on Bi-LSTM and Attention Mechanism. *2021 2nd International Conference on*

Artificial Intelligence and Information Systems, 1–5. https://doi.org/10.1145/3469213.3470701

Zaulir, Z. M., Liyana, N., Aziz, A., Aidya, N., & Aizam, H. (2022). A General Mathematical Model for University Courses Timetabling: Implementation to a Public University in Malaysia. *Malaysian Journal of Fundamental and Applied Sciences*, *18*, 82–94.