

**RANCANG BANGUN SISTEM *APPLICATION PROGRAMMING
INTERFACE* (API) MENGGUNAKAN GOLANG DENGAN METODE
KANBAN UNTUK MENGIMPLEMENTASI METODE *AUTISM
TREATMENT EVALUATION CHECKLIST* (ATEC)**

(Skripsi)

**Oleh
LUCKY AKBAR
1915061056**



**FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025**

**RANCANG BANGUN SISTEM *APPLICATION PROGRAMMING
INTERFACE (API)* MENGGUNAKAN GOLANG DENGAN METODE
KANBAN UNTUK MENGIMPLEMENTASI METODE *AUTISM
TREATMENT EVALUATION CHECKLIST (ATEC)***

Oleh:

LUCKY AKBAR

Skripsi

**Sebagai Salah Satu Syarat untuk Mencapai Gelar
SARJANA TEKNIK**

Pada

**Program Studi Teknik Informatika
Jurusan Teknik Elektro
Fakultas Teknik Universitas Lampung**



**FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025**

ABSTRAK

RANCANG BANGUN SISTEM *APPLICATION PROGRAMMING INTERFACE* (API) MENGGUNAKAN GOLANG DENGAN METODE KANBAN UNTUK MENGIMPLEMENTASI METODE *AUTISM TREATMENT EVALUATION CHECKLIST* (ATEC)

Oleh

LUCKY AKBAR

Autism Spectrum Disorder (ASD) merupakan kondisi neurologis kompleks yang memerlukan intervensi dini dan terapi berkelanjutan. Keberhasilan terapi ini perlu dipantau secara efektif. *Autism Treatment Evaluation Checklist* (ATEC) adalah alat penilaian yang terbukti secara klinis untuk mengukur perkembangan dan respons individu terhadap terapi. Namun, implementasinya masih terhambat oleh sistem pencatatan manual berbasis kertas, yang tidak efisien, rentan error, dan menyulitkan pelacakan progress jangka panjang.

Penelitian ini bertujuan untuk mengembangkan dan mengimplementasikan sebuah sistem *Application Programming Interface* (API) yang mengadopsi metode ATEC. Pengembangan sistem ini ditujukan untuk memberikan solusi terdigitalisasi, terotomasi, dan komprehensif agar dapat diintegrasikan dengan berbagai platform aplikasi, sehingga memudahkan pengguna dalam memanfaatkan ATEC.

Pengembangan perangkat lunak dilakukan dengan menerapkan metodologi Kanban untuk mengelola alur tugas secara efektif dan efisien. Untuk memastikan keandalan sistem, dilakukan pengujian menyeluruh yang meliputi *unit testing* untuk memvalidasi fungsi individual komponen dan *black-box testing* untuk menguji fungsionalitas sistem secara keseluruhan dari perspektif pengguna akhir.

Penelitian ini berhasil menghasilkan sebuah *RESTful API* yang lengkap dan fungsional. API ini menyediakan endpoint untuk menghitung skor ATEC secara otomatis, menyimpan riwayat *assessment*, dan melacak perkembangan skor pengguna dari waktu ke waktu. Pengujian *black-box* mengonfirmasi bahwa semua fungsionalitas utama sistem berjalan sesuai dengan persyaratan yang ditetapkan, sementara *unit testing* menjamin keakuratan perhitungan skor dan validitas data.

Sistem API yang dikembangkan berhasil mengatasi keterbatasan sistem manual dengan menawarkan aksesibilitas, automasi, dan akurasi yang lebih baik. Penerapan metode Kanban terbukti efektif dalam mengelola proses pengembangan. Solusi ini tidak hanya memudahkan proses monitoring terapi autisme tetapi juga membuka peluang untuk integrasi yang lebih luas dengan berbagai sistem kesehatan digital (*e-health*) di masa depan, sehingga potensi manfaat ATEC dapat dimaksimalkan.

Kata kunci: *Autism Spectrum Disorder (ASD), Autism Treatment Evaluation Checklist (ATEC), Application Programming Interface (API), Kanban, Black-box Testing*

ABSTRACT

DESIGN AND DEVELOPMENT OF AN APPLICATION PROGRAMMING INTERFACE (API) USING GOLANG WITH THE KANBAN METHOD TO IMPLEMENT AUTISM TREATMENT EVALUATION CHECKLIST (ATEC)

By

LUCKY AKBAR

Autism Spectrum Disorder (ASD) is a complex neurological condition that requires early intervention and continuous therapy. The effectiveness of this therapy needs to be monitored efficiently. The Autism Treatment Evaluation Checklist (ATEC) is a clinically proven assessment tool to measure an individual's progress and response to therapy. However, its implementation is still hindered by a manual, paper-based recording system, which is inefficient, prone to error, and complicates long-term progress tracking.

This research aims to develop and implement an Application Programming Interface (API) system that adopts the ATEC methodology. The development of this system is intended to provide a digitized, automated, and comprehensive solution that can be integrated with various application platforms, thereby making the ATEC more accessible and easier to use.

The software development was carried out by applying the Kanban methodology to manage the workflow effectively and efficiently. To ensure system reliability, thorough testing was conducted, including Unit Testing to validate the individual functions of components and Black-Box Testing to examine the overall system functionality from an end-user perspective.

This study successfully produced a fully functional RESTful API. The API provides endpoints for calculating ATEC scores automatically, storing assessment history, and tracking user score progress over time. Black-Box Testing confirmed that all core system functionalities operate as specified in the requirements, while Unit Testing guaranteed the accuracy of score calculations and data validity.

The developed API system successfully overcomes the limitations of manual systems by offering greater accessibility, automation, and accuracy. The application of the Kanban method proved effective in managing the development process. This solution not only simplifies the process of monitoring autism therapy but also opens opportunities for broader integration with various digital health (e-health) systems in the future, thus maximizing the potential benefits of the ATEC.

Keywords: Autism Spectrum Disorder (ASD), Autism Treatment Evaluation Checklist (ATEC), Application Programming Interface (API), Kanban, Black-box Testing

Judul Skripsi

: RANCANG BANGUN SISTEM *APPLICATION PROGRAMMING INTERFACE* (API) MENGGUNAKAN GOLANG DENGAN METODE KANBAN UNTUK MENGIMPLEMENTASI METODE *AUTISM TREATMENT EVALUATION CHECKLIST* (ATEC)

Nama Mahasiswa

: Lucky Akbar

Nomor Pokok Mahasiswa

: 1915061056

Program Studi

: Teknik Informatika

Fakultas

: Teknik



1. Komisi Pembimbing

Pembimbing Utama

Pembimbing Pendamping

Ir. Gigih Forda Nama, S.T., M.T.I., IPM.

NIP. 19830712 200812 1 003

Mahendra Pratama, S.T., M.Eng.

NIP. 19911215 201903 1 013

2. Mengetahui

Ketua Jurusan
Teknik Elektro

Ketua Program Studi
Teknik Informatika

Herlinawati, S.T., M.T.

NIP. 19710314 199903 2 001

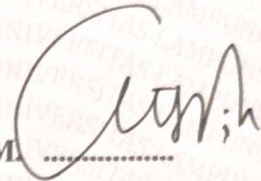
Yessi Mulyani, S.T., M.T.

NIP. 19731226 200012 2 001

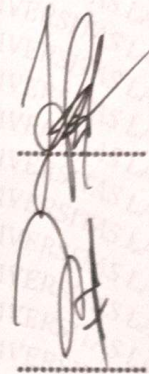
MENGESAHKAN

1. Tim Penguji

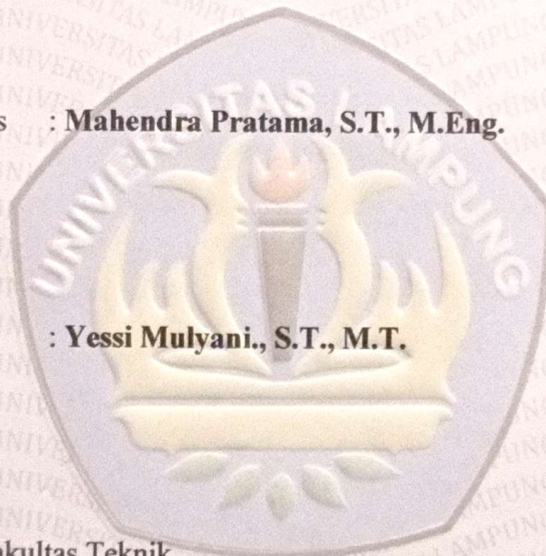
Ketua : **Ir. Gigih Forda Nama, S.T., M.T.I., IPM**



Sekretaris : **Mahendra Pratama, S.T., M.Eng.**



Penguji : **Yessi Mulyani, S.T., M.T.**



2. Dekan Fakultas Teknik



Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.

NIP. 19750928 200112 1 002

Tanggal Lulus Ujian Skripsi : **28 November 2025**

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini, menyatakan bahwa skripsi saya dengan judul “Rancang Bangun Sistem *Application Programming Interface* (API) Menggunakan Golang dengan Metode Kanban untuk Mengimplementasi Metode *Autism Treatment Evaluation Checklist* (ATEC)” dibuat oleh saya sendiri. Semua hasil yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti bahwa skripsi ini merupakan salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan ketentuan hukum atau akademik yang berlaku.

Bandar Lampung, 28 November 2025

Pembuat pernyataan



Lucky Akbar

NPM. 1915061056

RIWAYAT HIDUP



Penulis dilahirkan di Indralaya pada tanggal 29 Oktober 2001. Penulis merupakan anak pertama dari 4 bersaudara dari pasangan Bapak Aiptu Ibenu Hazar dan Ibu Dede.

Jenjang pendidikan yang dialami penulis dimulai dari TK Amarta Tani HKTI, SDN 2 Kampung Baru, SMPN 2 Bandar Lampung, SMAN 5 Bandar Lampung hingga menempuh pendidikan S1 pada program studi Teknik Informatika Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung.

Terhitung sejak SMA penulis sangat tertarik dengan kegiatan ekstrakurikuler berbasis akademik dan kegiatan berorganisasi. Penulis aktif mengikuti OSN bidang kimia, menjadi pengurus OSIS sekaligus menjadi seorang pramuka. Sama halnya di dunia perkuliahan, penulis aktif sebagai pengurus Himatro selama dua periode, aktif pada organisasi Unila Robotika dan Otomasi (URO). Penulis juga mengikuti program Kampus Merdeka selama dua semester, yakni Studi Independen pada subjek keahlian *Front End & Machine Learning* serta magang merdeka sebagai *Back End Developer intern* di PT Dynamo Media Network atau yang lebih dikenal dengan kumparan.

Penulis sangat suka membaca buku, terutama buku-buku saintifik. Penulis sadar sepenuhnya seorang manusia tidak boleh berhenti belajar karena tanpa adanya landasan ilmu pengetahuan yang cukup, maka praktik di dunia nyata akan menjadi sulit atau bahkan tidak mungkin terlaksana dengan baik.

MOTTO

“Sesungguhnya beserta kesulitan ada kemudahan.”

(Q.S. Ash-Sharh (94): 6)

"The mind can atrophy, like the muscles, if it is not used."

(Mortimer J. Adler & Charles Van Doren, *How to Read a Book*)

"Some things are up to us, some things are not up to us"

(Epictetus)

"There are two rules for living in harmony.

1. Don't sweat the small stuff and

2. It's all small stuff"

(Richard Carlson, PhD., *Don't Sweat the Small Stuff*)

PERSEMBAHAN

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Segala puji syukur kepada Allah SWT. atas segala Rahmat dan karunia-Nya sehingga saya dapat menyelesaikan skripsi ini. shalawat serta salam teriring kepada Nabi Muhammad SAW sebagai suri tauladan *akhlakul karimah*.

KUPERSEMBAHKAN KARYA INI KEPADA:

“Kedua orang tuaku, Ayah dan Bunda tercinta, yang senantiasa mengutamakan yang terbaik bagi anak-anaknya. Tidak pernah lelah memberikan nasihat, masukan serta teladan yang baik. ”

“Seluruh Keluarga Besar Jurusan Teknik Elektro yang telah secara langsung dan tidak langsung berkontribusi dalam mendukung penyelesaian skripsi ini.”

“Almamater tercinta, Universitas Lampung”

SANWACANA

Puji syukur penulis panjatkan kepada kehadirat Allah SWT. yang telah memberikan rahmat dan ridho-Nya sehingga penulis dapat menyelesaikan penelitian yang berjudul “Rancang Bangun Sistem *Application Programming Interface* (API) Menggunakan Golang dengan Metode Kanban untuk Mengimplementasi Metode *Autism Treatment Evaluation Checklist* (ATEC)”. Selama masa penelitian penulis mendapatkan banyak bantuan, bimbingan serta arahan. Oleh karena itu penulis mengucapkan terima kasih sebanyak-banyaknya kepada:

1. Kedua orang tua tercinta dan keluarga penulis yang selalu memberikan dukungan, doa, dan kasih sayang tiada akhir;
2. Bapak Dr. Eng. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung;
3. Ibu Herlinawati, S.T., M.T. selaku Ketua Jurusan Teknik Elektro Universitas Lampung;
4. Ibu Yessi Mulyani, S.T., M.T. selaku Ketua Program Studi Teknik Informatika Universitas Lampung, Dosen Pembimbing Akademik serta Dosen Penguji yang telah membantu proses kelancaran pengerjaan penelitian dan perkuliahan;
5. Bapak Ir. Gigih Forda Nama, S.T., M.T.I., IPM selaku Pembimbing Utama yang telah memberikan bimbingan, dukungan serta arahan penulis dalam menyelesaikan penelitian ini;
6. Bapak Mahendra Pratama, S.T., M.Eng. selaku Pembimbing Pendamping yang juga telah memberikan bimbingan, dukungan serta arahan penulis dalam menyelesaikan penelitian ini;
7. Seluruh *civitas* akademik Jurusan Teknik Elektro;

8. Keluarga besar Teknik Elektro Angkatan 2019 yang telah menjadi teman seperjuangan selama perkuliahan;
9. Kak Yori, Kak Jihan, Kak Nadia, Kak Ramadhan, Kak Rio yang sudah sering memberi arahan seputar dunia perkuliahan;
10. Anak-anak BC 19, Abyyu, Alwan, Bang Kholid, Bang Ilham, Bang Caesar, Adi, Nugraha dan Bang Seno yang selalu jadi tempat tukar pikiran, baik ketika mengurus angkatan ataupun seputar perkuliahan;
11. Sahabat-sahabat seperjuangan di perkuliahan, Rizke, Ellangga, Irfan, Juan, Ilham (Eng), Aldi, Arya, Lesy, Julio, Bagus, Anggi, Alfiyah dan Sarah yang senantiasa selalu saling tolong menolong selama masa perkuliahan.

Penulis berharap agar laporan ini dapat menjadi referensi bagi pengembangan keilmuan di bidang Teknik Informatika serta bermanfaat bagi pembacanya.

Bandar Lampung, 28 November 2025

Penulis,

Lucky Akbar

DAFTAR ISI

	Halaman
DAFTAR ISI.....	XV
DAFTAR GAMBAR.....	XVIII
DAFTAR TABEL	XXI
I. PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	4
1.3 Tujuan Penelitian.....	5
1.4 Manfaat Penelitian.....	5
1.5 Batasan Masalah.....	6
1.6 Sistematika Penulisan	6
II. TINJAUAN PUSTAKA	8
2.1 API.....	8
2.2 REST	8
2.3 Go	9
2.4 <i>Autism Treatment Evaluation Checklist (ATEC)</i>	9
2.5 <i>Speech Delay (Keterlambatan Bicara)</i>	11
2.6 Kanban.....	11
2.7 Basis Data.....	14
2.8 <i>JavaScript Object Notation (JSON)</i>	14
2.9 <i>Unit Testing</i>	15
2.10 <i>Black Box Testing</i>	15
2.11 <i>Entity Relationship Diagram (ERD)</i>	16
2.12 Docker	16

2.13 <i>JSON Web Token (JWT)</i>	17
2.14 Visual Studio Code.....	17
2.15 Ubuntu	18
2.16 Insomnia	18
2.17 Swaggo	19
2.18 <i>Clean Architecture</i>	19
2.19 Git dan Github	21
2.20 Penelitian Terkait.....	22
III. METODOLOGI PENELITIAN	28
3.1 Waktu dan Tempat Penelitian	28
3.2 Perangkat Penelitian	28
3.3 Tim Penelitian	29
3.4 Tahapan Penelitian	30
3.3.1 Studi Literatur.....	32
3.3.2 <i>Software Development Life Cycle dengan Kanban</i>	32
3.3.3 Penulisan Laporan	41
IV. HASIL DAN PEMBAHASAN	42
4.1 Studi Literatur.....	42
4.2 <i>Software Development Life Cycle dengan Kanban</i>	43
4.2.1 <i>User's Requests / Stories</i>	45
4.2.2 <i>Requirements Gathering</i>	47
4.2.3 <i>Bug Report</i>	65
4.2.4 <i>Bug Classification & Verification</i>	66
4.2.5 <i>Backlog</i>	67
4.2.6 <i>Work In Progress</i>	70
4.2.7 <i>Code Review</i>	79
4.2.8 <i>Software In Test</i>	81
4.2.9 <i>Deployment Staging</i>	117
4.2.10 <i>User Acceptance Test</i>	117
4.2.11 <i>Deployment Production</i>	118
V. KESIMPULAN DAN SARAN	123
5.1 Kesimpulan.....	123

5.2 Saran	124
DAFTAR PUSTAKA	125

DAFTAR GAMBAR

Gambar	Halaman
Gambar 2.1. Formulir ATEC (sumber https://ariconference.com/ATEC.pdf)	10
Gambar 2.2. Kanban Board.....	13
Gambar 2.3. Clean Architecture (sumber: https://blog.cleancoder.com).....	19
Gambar 2.4. Contoh Repositori Kode Github.....	21
Gambar 3.1. Tim Pengembang.....	30
Gambar 3.2. Tahapan Penelitian	31
Gambar 4.1. Visualisasi Alur Kerja dengan Kanban Board	44
Gambar 4.2. Use Case Diagram	51
Gambar 4.3. ERD.....	53
Gambar 4.4. Arsitektur Sistem.....	55
Gambar 4.5. Kanban Board Pengembangan ATEC API	67
Gambar 4.6. Backlog.....	68
Gambar 4.7. Tampilan Detail Tiket Story.....	69
Gambar 4.8. Tiket Laporan Bug	69
Gambar 4.9. Pemilihan Tiket untuk Tahapan Work In Progress	70
Gambar 4.10. Kode Sumber Modul Use Case Fitur Sign Up	71
Gambar 4.11. Kode Sumber Modul Data Access Layer Tiket ST-2	72
Gambar 4.12. Antarmuka Data Access Method User Repository	73
Gambar 4.13. Kode Sumber Tiket ST-3	74
Gambar 4.14. Kode Sumber Tugas ST-18	75
Gambar 4.15. Kode Sumber Tugas ST-63	76
Gambar 4.16. Kode Sumber Tugas ST-64	77
Gambar 4.17. Lama Waktu Pengerjaan	78

Gambar 4.18. Aturan Linter	79
Gambar 4.19. Workflow Linter Github.....	80
Gambar 4.20. Galat Pada Proses Linter Github	80
Gambar 4.21. Hasil Uji dan Cakupan Unit Testing	81
Gambar 4.22. Keluaran Sistem Pada Pengujian US-1	83
Gambar 4.23. Surel Konfirmasi Pendaftaran	84
Gambar 4.24. Tampilan Berhasil Verifikasi Akun	84
Gambar 4.25. Keluaran Sistem Pada Permintaan Log In.....	85
Gambar 4.26. Surel Penggantian Kata Sandi	87
Gambar 4.27. Antarmuka Penggantian Kata Sandi	88
Gambar 4.28. Percobaan <i>Log In</i> Dengan Kata Sandi Terbaru	88
Gambar 4.29. Permintaan Pembuatan Paket ATEC Aktor Administrator.....	91
Gambar 4.30. Permintaan Pembuatan Paket ATEC Dari Aktor Terapis Ditolak .	91
Gambar 4.31. Hasil Uji <i>Story</i> US-5	94
Gambar 4.32. Hasil Pengujian Perubahan Paket Kuesioner	95
Gambar 4.33. Keluaran Sistem Pada Proses Aktivasi.....	96
Gambar 4.34. Keluaran Sistem Pada Proses Deaktivasi	97
Gambar 4.35. Hasil Pengujian Fitur Penghapusan Paket Kuesioner	98
Gambar 4.36. Sistem Menolak Penghapusan Paket yang Terkunci.....	98
Gambar 4.37. Respons Sistem Terhadap Permintaan Pendaftaran Anak	99
Gambar 4.38. Pesan Kesalahan Sistem Pada Fitur Pendaftaran Anak.....	100
Gambar 4.39. Respons Sistem Pada Fitur Perubahan Data Anak.....	101
Gambar 4.40. Respons Sistem Terhadap Permintaan Perubahan Ilegal	101
Gambar 4.41. Respons Sistem Pada Permintaan Data Anak Terdaftar	102
Gambar 4.42. Hasil Pencarian Data Anak Tanpa Parameter	103
Gambar 4.43. Hasil Pencarian Data Anak Dengan Parameter Khusus.....	104
Gambar 4.44. Respons Sistem Pada Permintaan Paket Kuesioner Aktif.....	105
Gambar 4.45. Respons Sistem Terhadap Permintaan Pengisian Kuesioner	107
Gambar 4.46. Akurasi Perhitungan Skor ATEC Oleh Sistem	107
Gambar 4.47. Respons Permintaan Pengunduhan Hasil Kuesioner ATEC	109
Gambar 4.48. Sistem Menolak Permintaan Unduhan Ilegal.....	109
Gambar 4.49. Respons Sistem Pada Fitur Statistik Skor ATEC.....	111

Gambar 4.50. Respons Sistem Pada Permintaan Riwayat Kuesioner.....	112
Gambar 4.51. Respons Terhadap Permintaan Pencarian Hasil Kuesioner	114
Gambar 4.52. Keluaran Sistem Pada Fitur Hapus Akun.....	115
Gambar 4.53. Alur Pembuatan Aplikasi	118
Gambar 4.54. Implementasi Docker	119
Gambar 4.55. Keluaran Perintah Pembuatan <i>Docker Image</i>	120
Gambar 4.56. Berkas Docker Compose	121
Gambar 4.57. Keluaran Perintah Docker Compose	122

DAFTAR TABEL

Tabel	Halaman
Tabel 2.1. Penelitian Terkait	24
Tabel 3.1. Jadwal Penelitian.....	28
Tabel 3.2. Perangkat Penelitian.....	29
Tabel 4.1. Daftar User Stories.....	45
Tabel 4.2. Kebutuhan Fungsional	48
Tabel 4.3. Kebutuhan non-Fungsional	50
Tabel 4.4. Daftar <i>Sub Tasks</i>	56
Tabel 4.5. <i>API Contracts</i>	63
Tabel 4.6. Laporan <i>Bug</i>	65
Tabel 4.7. Proses Verifikasi Laporan <i>Bug</i>	66
Tabel 4.8. Proses Klasifikasi Laporan <i>Bug</i>	67
Tabel 4.9. Hasil Uji <i>Story</i> US-1	82
Tabel 4.10. Hasil Uji <i>Story</i> US-2	85
Tabel 4.11. Hasil Uji <i>Story</i> US-3	86
Tabel 4.12. Hasil Uji <i>Story</i> US-4	89
Tabel 4.13. Hasil Uji <i>Story</i> US-5	92
Tabel 4.14. Hasil Uji <i>Story</i> US-6	94
Tabel 4.15. Hasil Uji <i>Story</i> US-7	96
Tabel 4.16. Hasil Uji <i>Story</i> US-8	97
Tabel 4.17. Hasil Uji <i>Story</i> US-9	99
Tabel 4.18. Hasil Uji <i>Story</i> US-10	100
Tabel 4.19. Hasil Uji <i>Story</i> US-11	102
Tabel 4.20. Hasil Uji <i>Story</i> US-12	103

Tabel 4.21. Hasil Uji <i>Story</i> US-13	104
Tabel 4.22. Hasil Uji <i>Story</i> US-14	105
Tabel 4.23. Hasil Uji <i>Story</i> US-15	108
Tabel 4.24. Hasil Uji <i>Story</i> US-16	110
Tabel 4.25. Hasil Uji <i>Story</i> US-17	111
Tabel 4.26. Hasil Uji <i>Story</i> US-18	113
Tabel 4.27. Hasil Uji <i>Story</i> US-19	114
Tabel 4.28. Hasil Uji <i>Bug</i> AA-83	116

I. PENDAHULUAN

1.1 Latar Belakang

Autisme adalah gangguan perkembangan sistem syaraf yang umumnya dimulai dari janin hingga tahun-tahun pertama kehidupan yang ditandai dengan kurangnya aktifitas sosial hingga perilaku terbatas yang terus diulang-ulang. [1] Perkembangan dalam studi seputar autisme juga menunjukkan bahwa gangguan autisme sangatlah heterogen, sehingga digunakanlah istilah yang bisa mencakup heterogenitas gangguan autisme yang disebut dengan *Autism Spectrum Disorder* atau disingkat dengan ASD. [2] Pada tahun 2021, perkiraan prevalensi autisme secara global mencapai 120 kasus per 10.000 anak, atau sekitar 1 dari 100 anak terdiagnosa menderita autisme. [3]

Meskipun tidak ada satu sumber pasti penyebab autisme, peran genetik sangatlah erat sebagai penyebab autisme. [4] Masa kehamilan hingga melahirkan juga menunjukkan adanya potensi menjadi sebab autisme pada anak, seperti obesitas serta komplikasi kelahiran yang berhubungan dengan iskemia dan hipoksia. [5] Meskipun penelitian menunjukkan bahwa semakin awal autisme bisa diketahui dan diberikan tindakan pada anak, maka hasilnya akan semakin baik untuk anak tersebut. [6] Sayangnya, hal tersebut tidaklah mudah dilakukan, mengingat jumlah kelahiran bayi yang cukup tinggi dimana UNICEF pernah memberikan estimasi jumlah kelahiran bayi per harinya mencapai 392.000 anak. [7]

Kondisi senada juga terjadi di wilayah Lampung, dimana setiap tahunnya jumlah siswa yang bersekolah di Sekolah Luar Biasa (SLB), baik swasta maupun negeri, terus bertambah setiap tahunnya. Pada tahun ajaran 2016 / 2017 terdapat 92 siswa,

pada tahun ajaran 2019 / 2020 meningkat menjadi 224 siswa, [8] dan pada puncaknya pada tahun 2024 mencapai 2187 siswa. [9] Hal ini menunjukkan bahwasanya tren jumlah peserta didik di Sekolah Luar Biasa khususnya di daerah Lampung terus meningkat setiap tahunnya.

Kondisi ini juga diperburuk dengan sulitnya akses kesehatan balita di Indonesia yang tidak merata, khususnya bagi masyarakat yang tinggal di desa atau golongan masyarakat miskin. [10] Maka dari itu, diperlukan adanya sebuah metode untuk dapat membantu mendeteksi gejala spektrum autisme pada anak di usia yang sedini mungkin serta mudah diakses oleh semua kalangan. Salah satu pendekatannya adalah melalui orang tua, dimana orang tua adalah sosok yang paling dekat dengan anak sehingga intervensi yang dilakukan bisa lebih personal, sensitif dan juga efektif secara biaya dan waktu. [11]

Penerapan dari metode tersebut juga harus dimungkinkan untuk diterapkan melalui media *online*, mengingat akses ke fasilitas kesehatan yang masih tidak merata di semua wilayah Indonesia. [12] Media *online* juga dipilih mengingat secara global, lebih dari 67.1% penduduk global telah memiliki akses ke internet. [13] Data ini juga tidak jauh berbeda dengan Indonesia, dimana sudah lebih dari 215 juta atau sekitar 78% penduduknya yang telah memiliki akses ke Internet. [14] Maka dari itu, penggunaan media yang dapat diakses secara *online* akan sangat menguntungkan baik dari segi jangkauan pengguna dan kemudahan akses.

Setelah penjelasan akan pentingnya diagnosa dan intervensi autisme pada anak sedini mungkin serta berbagai tantangan dalam melakukannya, kita membutuhkan sebuah metode diagnosa autisme yang mudah dilakukan sehingga para orang tua bisa melakukannya, murah, serta efektif dalam mendiagnosa autisme. Salah satu metode yang memenuhi kriteria di atas adalah *Autism Treatment Evaluation Checklist* atau yang biasa disingkat ATEC. Metode ATEC adalah sebuah metode bebas royalti untuk mendiagnosa tingkat keparahan gejala autisme serta mudah digunakan oleh orang tua atau pengasuh anak dan tersedia dalam bentuk formulir pertanyaan. [15]

Metode ATEC sangat berguna untuk melacak efektifitas pengobatan terhadap gejala autisme karena mampu mengukur perubahan gejala autisme pada anak. [16] Meskipun metode ATEC bukanlah standar baku sebagai instrumen dalam mengukur keparahan gejala autisme seperti metode CARS (*Childhood Autism Rating Scale*) yang dirancang untuk dilakukan oleh profesional, metode ATEC dapat meningkatkan kemanjuran pengobatan tanpa peningkatan biaya yang besar. [17] Meskipun demikian, metode ATEC yang mengandalkan orang tua bisa menghasilkan keluaran yang berkorelasi tinggi dengan metode CARS yang dilakukan oleh profesional. [18] Penelitian yang dilakukan pada tahun 2011 juga menunjukkan bahwasanya orang tua bisa menjadi sumber yang andal untuk mendapatkan informasi mengenai kondisi anak yang mengalami autisme sehingga metode ATEC adalah instrumen yang berguna untuk mengumpulkan informasi terkini terhadap anak-anak dengan gejala autisme. [19] Ditambah lagi metode ATEC terbukti mampu mempertahankan validitas hasilnya meskipun diterapkan ke versi budaya / bahasa lain, seperti versi Iran [20], Sri Lanka [21] dan juga versi Thailand. [22]

Meskipun metode ATEC sangat potensial untuk menjadi alat yang mampu diutilisasikan untuk melakukan monitoring perkembangan anak dengan gangguan spektrum autisme, namun solusi terpadu untuk memanfaatkannya masih absen. Belum adanya sebuah solusi berbasis aplikasi yang mampu menerapkan metode ATEC yang dapat diakses melalui internet dan memberikan fitur yang komprehensif sesuai dengan keunggulan metode itu sendiri. Terkhusus di Indonesia, dimana terdapat lebih dari 1000 bahasa daerah [23] serta indeks kecakapan berbahasa Inggris penduduk Indonesia yang tergolong rendah [24] menjadikan dukungan multi bahasa juga diperlukan untuk implementasinya di masyarakat yang heterogen seperti di Indonesia. Maka dari itu, implementasi metode ATEC yang dibuat berbasis *Application Programming Interface* (API) dengan dukungan fitur yang komprehensif serta adanya dukungan multi bahasa menjadikan metode ATEC lebih bisa diimplementasikan serta bisa lebih dimanfaatkan oleh masyarakat luas.

Penelitian ini juga akan bekerja sama dengan salah satu Sekolah Luar Biasa (SLB) di daerah Lampung, khususnya orang tua siswa sebagai partisipan dalam proses pengujian aplikasi. Para orang tua siswa ini diharapkan dapat merepresentasikan target utama pengguna aplikasi sekaligus dapat memberikan umpan balik terhadap pengembangan aplikasi agar sesuai dengan peruntukannya serta dapat memenuhi kebutuhan pengguna.

1.2 Rumusan Masalah

Berdasarkan uraian dari latar belakang, rumusan masalah pada penelitian ini adalah sebagai berikut:

1. Bagaimana merancang dan membangun sistem API berbasis Golang guna mengimplementasikan metode ATEC secara komprehensif, sesuai kebutuhan pengguna, serta dapat diintegrasikan dengan beragam antarmuka pengguna.
2. Bagaimana menerapkan pengujian berbasis *black box testing* dan *unit testing* pada sistem berbasis API sehingga dapat memastikan fungsionalitas sistem berjalan sesuai dengan kebutuhan pengguna.
3. Bagaimana menerapkan metode Kanban pada proses pengembangan perangkat lunak berbasis API agar dapat mengakomodir kebutuhan pengguna yang dinamis, dapat mengintegrasikan umpan balik pengguna di setiap iterasi tahapan pengembangan serta dapat meningkatkan efektivitas dan efisiensi pengembangan perangkat lunak.

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut:

1. Merancang dan membangun sistem API berbasis Golang guna mengimplementasikan metode ATEC secara komprehensif, sesuai kebutuhan pengguna, serta dapat diintegrasikan dengan beragam antarmuka pengguna.
2. Menerapkan pengujian berbasis *black box testing* dan *unit testing* pada sistem berbasis API sehingga dapat memastikan fungsionalitas sistem berjalan sesuai dengan kebutuhan pengguna.
3. Menerapkan metode Kanban pada proses pengembangan perangkat lunak berbasis API agar dapat mengakomodir kebutuhan pengguna yang dinamis, dapat mengintegrasikan umpan balik pengguna di setiap iterasi tahapan pengembangan serta dapat meningkatkan efektivitas dan efisiensi pengembangan perangkat lunak.

1.4 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah sebagai berikut:

1. Membantu orang tua untuk menggunakan alat berbasis ATEC untuk melakukan terapi pengobatan autisme.
2. Memudahkan akses terhadap fasilitas monitoring perkembangan anak dalam bentuk statistik dalam mengatasi autisme.
3. Menjembatani para ahli di bidang autisme untuk memberikan tes autisme berbasiskan ATEC untuk diakses oleh masyarakat luas.
4. Membuka peluang integrasi ke berbagai jenis antarmuka pengguna untuk memaksimalkan potensi menjangkau semua jenis pengguna.
5. Membuka peluang integrasi antar sistem untuk membuat sistem terpadu untuk kesehatan anak secara menyeluruh.
6. Memudahkan dokter dan para ahli di bidang autisme untuk mendapatkan hasil uji ATEC dalam rentang waktu pengobatan dan terapi autisme.

1.5 Batasan Masalah

Pada proses pembangunan sistem ATEC berbasis API menggunakan golang ini memiliki beberapa batasan masalah guna memfokuskan penelitian agar dapat memenuhi tujuan penelitian yaitu sebagai berikut:

1. Penelitian ini difokuskan pada pengembangan perangkat lunak yang mengimplementasi ATEC dan tidak bertujuan memverifikasi akurasi serta validitas hasil uji ATEC.
2. Keluaran dari sistem berupa *response* dari *server* dalam format JSON.
3. Sampel kuesioner interview ATEC akan diambil dari sumber eksternal.
4. Uji coba sistem hanya akan menggunakan pengujian unit (*unit testing*) dan *black box testing*.
5. Akses terhadap sistem membutuhkan koneksi internet yang stabil.
6. Sistem yang dikembangkan masih bersifat lokal *server* dan belum di-*deploy* secara publik.

1.6 Sistematika Penulisan

Sistematika penulisan terdiri dari lima bab yang bertujuan untuk mempermudah penulisan dalam menyusun, mengelola, menyebarkan, dan merangkai data yang telah diperoleh sehingga penulisan ini dapat dituliskan dengan baik dan benar.

BAB I PENDAHULUAN

Dalam bab ini berisi latar belakang, identifikasi masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian serta sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Dalam bab ini membahas terkait tinjauan pustaka dan dasar teori yang digunakan pada skripsi ini.

Bab III METODOLOGI PENELITIAN

Dalam bab ini menjelaskan metodologi penelitian yang digunakan pada penelitian ini serta waktu dan tempat penelitian.

BAB IV HASIL DAN PEMBAHASAN

Dalam bab ini membahas tentang hasil penelitian yang telah dilakukan dan pembahasan pada penelitian ini.

BAB V KESIMPULAN DAN SARAN

Dalam bab ini membahas kesimpulan dari hasil penelitian yang telah dilakukan pada skripsi dan saran dari hasil peneliti.

II. TINJAUAN PUSTAKA

2.1 API

API adalah singkatan dari *Application Programming Interface* merupakan sebuah set berisi definisi dan protokol untuk membangun dan mengintegrasikan sebuah perangkat lunak. Seringkali juga API dianggap sebagai kontrak perjanjian antara penyedia informasi (*server*) dan penerima informasi (*client*) dalam hal pertukaran informasi. [25] Dalam penelitian ini, API akan digunakan sebagai sebuah aturan dan perjanjian antara permintaan (*request*) dan juga respons dari aplikasi yang akan dibangun sehingga baik *client* dan *server* bisa mengembangkan aplikasi tanpa harus terhambat satu sama lain. Tidak hanya itu, dengan menggunakan API, antara *client* dan *server* tidak saling ketergantungan dalam membangun antarmuka pengguna.

2.2 REST

REST adalah singkatan dari *Representational State Transfer* merupakan sebuah gaya arsitektur dalam membangun aplikasi, sehingga memungkinkan developer melakukan implementasi REST dengan beragam metode. [25] Beberapa gaya arsitektur REST adalah sebagai berikut:

1. Menggunakan sistem *client-server* dan protokol *request* menggunakan HTTP.
2. Tidak ada informasi *client* yang tersimpan dalam komunikasi *client-server*, serta setiap *request* yang dilakukan tidak saling terhubung.

3. Mendukung teknik *caching* yang dapat menyederhanakan interaksi *client-server*.
4. Mendukung sistem berlapis (*layered systems*) sehingga memungkinkan sistem memiliki lapisan terpisah dengan tanggung jawabnya sendiri, seperti *load balancer*, *cache*, hingga *proxy*.

2.3 Go

Go atau lebih dikenal dengan Golang adalah sebuah bahasa pemrograman yang dikembangkan Google sejak 2007 bertipe data statis (*staticly typed*) yang sangat cocok digunakan untuk beragam jenis pengembangan software mulai dari CLI (*command line interface*), aplikasi web, hingga membangun sistem operasi. Pemilihan bahasa Go dibanding dengan puluhan bahasa pemrograman lain adalah mudahnya menggunakan teknik *concurrency* yang sangat mudah dan diuntungkan dengan *multi-core processor* sehingga penggunaan Go dapat mendukung pemrosesan yang lebih cepat dengan efisien.

2.4 Autism Treatment Evaluation Checklist (ATEC)

ATEC dikembangkan oleh Bernard Rimland dan Stephen M. Edelson pada tahun 1999 untuk menjadi sebuah metode yang mudah digunakan, sensitif terhadap perubahan, dan valid yang secara khusus dikembangkan untuk anak-anak penderita autisme. ATEC adalah sebuah formulir pendek sepanjang satu halaman bebas hak cipta yang berisi pertanyaan dan jawaban berbentuk ceklis serta didesain untuk digunakan oleh orang tua, guru atau pengasuh anak-anak penderita autisme. [15] Contoh dari formulir ATEC dari pengembang aslinya ada di laman berikut <https://ariconference.com/ATEC.pdf> atau secara gambar, bisa dilihat pada gambar di bawah ini

ARI Form
ATEC-1/11-99

Autism Treatment Evaluation Checklist (ATEC)
Bernard Rimland, Ph.D. and Stephen M. Edelson, Ph.D.
Autism Research Institute
4182 Adams Avenue, San Diego, CA 92116
www.autism.org

Project/Patient: _____

Score	I	II	III	IV	Total
-------	---	----	-----	----	-------

This form is intended to measure the effects of treatment. Free scoring of this form is available on the Internet at: www.autism.org

Name of Child _____ Last _____ First _____ ☐ Male Age _____
☐ Female Date of Birth _____
Form completed by: _____ Relationship: _____ Today's Date _____

Please circle the letters to indicate how true each phrase is:

I. Speech/Language/Communication: [N] Not true [S] Somewhat true [V] Very true

N S V 1. Knows own name	N S V 6. Can use 3 words at a time (Want more milk)	N S V 11. Speech tends to be meaningful/relevant
N S V 2. Responds to 'No' or 'Stop'	N S V 7. Knows 10 or more words	N S V 12. Often uses several successive sentences
N S V 3. Can follow some commands	N S V 8. Can use sentences with 4 or more words	N S V 13. Carries on fairly good conversation
N S V 4. Can use one word at a time (No!, Eat, Water, etc.)	N S V 9. Explains what he/she wants	N S V 14. Has normal ability to communicate for his/her age
N S V 5. Can use 2 words at a time (Don't want, Go home)	N S V 10. Asks meaningful questions	

II. Sociability: [N] Not descriptive [S] Somewhat descriptive [V] Very descriptive

N S V 1. Seems to be in a shell - you cannot reach him/her	N S V 7. Shows no affection	N S V 14. Disagreeable/not compliant
N S V 2. Ignores other people	N S V 8. Fails to greet parents	N S V 15. Temper tantrums
N S V 3. Pays little or no attention when addressed	N S V 9. Avoids contact with others	N S V 16. Lacks friends/companions
N S V 4. Uncooperative and resistant	N S V 10. Does not imitate	N S V 17. Rarely smiles
N S V 5. No eye contact	N S V 11. Dislikes being held/cuddled	N S V 18. Insensitive to other's feelings
N S V 6. Prefers to be left alone	N S V 12. Does not share or show	N S V 19. Indifferent to being liked
	N S V 13. Does not wave 'bye-bye'	N S V 20. Indifferent if parent(s) leave

III. Sensory/Cognitive Awareness: [N] Not descriptive [S] Somewhat descriptive [V] Very descriptive

N S V 1. Responds to own name	N S V 7. Appropriate facial expression	N S V 13. Initiates activities
N S V 2. Responds to praise	N S V 8. Understands stories on T.V.	N S V 14. Dresses self
N S V 3. Looks at people and animals	N S V 9. Understands explanations	N S V 15. Curious, interested
N S V 4. Looks at pictures (and T.V.)	N S V 10. Aware of environment	N S V 16. Venturesome - explores
N S V 5. Does drawing, coloring, art	N S V 11. Aware of danger	N S V 17. "Tuned in" - Not spaced out
N S V 6. Plays with toys appropriately	N S V 12. Shows imagination	N S V 18. Looks where others are looking

IV. Health/Physical/Behavior: Use this code: [N] Not a Problem [MI] Minor Problem [MO] Moderate Problem [S] Serious Problem

N MI MO S 1. Bed-wetting	N MI MO S 10. Lethargic	N MI MO S 19. Rigid routines
N MI MO S 2. Wets pants/diapers	N MI MO S 11. Hits or injures self	N MI MO S 20. Shouts or screams
N MI MO S 3. Soils pants/diapers	N MI MO S 12. Hits or injures others	N MI MO S 21. Demands sameness
N MI MO S 4. Diarrhea	N MI MO S 13. Destructive	N MI MO S 22. Often agitated
N MI MO S 5. Constipation	N MI MO S 14. Sound-sensitive	N MI MO S 23. Not sensitive to pain
N MI MO S 6. Sleep problems	N MI MO S 15. Anxious/fearful	N MI MO S 24. "Hooked" or fixated on certain objects/topics
N MI MO S 7. Eats too much/too little	N MI MO S 16. Unhappy/crying	N MI MO S 25. Repetitive movements (stimming, rocking, etc.)
N MI MO S 8. Extremely limited diet	N MI MO S 17. Seizures	
N MI MO S 9. Hyperactive	N MI MO S 18. Obsessive speech	

Copyright (c) 2016 AUTISM RESEARCH INSTITUTE. ALL RIGHTS RESERVED. THE AUTISM TREATMENT EVALUATION CHECKLIST (ATEC) MAY BE USED ONLY FOR NON-COMMERCIAL PURPOSES.

Gambar 2.1. Formulir ATEC (sumber <https://ariconference.com/ATEC.pdf>)

Metode ATEC terdiri dari 4 bagian, yaitu Ucapan / Bahasa / Komunikasi, Keterampilan Sosial, Kesadaran Sensori / Kognitif, Kesehatan/Fisik/Perilaku. Bagian pertama memiliki 14 pertanyaan yang masing-masing memiliki 3 pilihan dengan rentang nilai dari 0 hingga 2. Bagian kedua memiliki 20 pertanyaan yang masing-masing memiliki 3 pilihan dengan rentang nilai dari 0 hingga 2. Bagian ketiga memiliki 18 pertanyaan yang masing-masing memiliki 3 pilihan dengan rentang nilai dari 0 hingga 2. Bagian keempat memiliki 25 pertanyaan yang masing-masing memiliki 4 pilihan dengan rentang nilai dari 0 hingga 3. Rentang skor dari hasil ATEC adalah 0 hingga 179, dimana semakin tinggi skornya maka menunjukkan gejala yang semakin parah. [21] Sehingga setiap pilihan jawaban

yang menunjukkan gejala yang memburuk akan mendapatkan skor yang lebih tinggi.

2.5 *Speech Delay* (Keterlambatan Bicara)

Keterlambatan bicara atau *speech delay* yaitu pada tingkat perkembangan bicara anak berada di bawah tingkat kualitas bicara anak yang umumnya sama [26]. Beberapa indikasi terjadinya keterlambatan bicara pada anak diantaranya adalah terbata-bata dalam menyebut kata-kata konsonan, tidak pandai membaca teks secara umum, tidak bisa mengeja, tidak pandai dalam memahami makna kalimat yang mereka baca, permasalahan dalam bersikap, bahkan hingga kesulitan bersosialisasi serta perkembangan akademik yang rendah. [27]

Beberapa hal yang melatarbelakangi terjadinya keterlambatan bicara pada anak yaitu meliputi jenis kelamin dimana anak laki-laki cenderung lebih mudah mengalami keterlambatan bicara daripada anak perempuan, trauma fisik, faktor genetika, penggunaan gadget yang berlebihan, hingga pola asuh yang tidak baik. [28] Terdapat beberapa hal sederhana yang tidak begitu akurat namun bisa dilakukan para orang tua di rumah untuk mengetahui apakah anaknya mengalami keterlambatan bicara seperti melakukan perbandingan dengan anak usia sebaya yang berada di lingkungan yang sama, melakukan interaksi sosial pada anaknya seperti bertanya, memberikan pendapat dan memerhatikan artikulasi bicara si anak ketika diajak bicara. [29]

2.6 Kanban

Kata “kanban” sendiri berasal dari bahasa Jepang yang diterjemahkan sebagai “papan visual” ataupun “tanda”. [30] Meskipun cukup terkenal di dunia teknologi informasi, metode kanban sendiri tercipta sekitar tahun 1950 dari industri otomotif

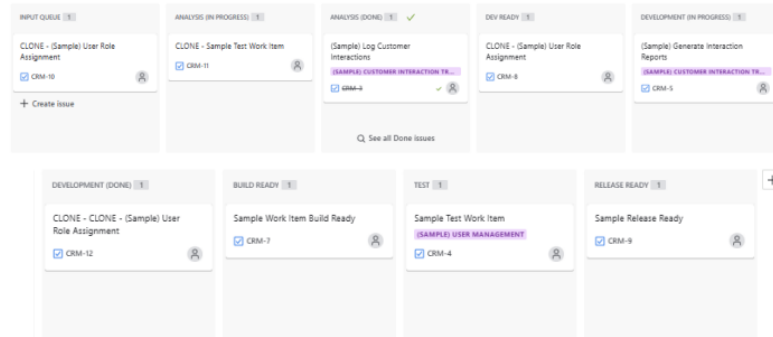
jepang, yaitu Toyota dalam proses sistem penjadwalan untuk teknik produksi *just-in-time*. Kanban sendiri mulai ramai diterapkan di industri teknologi informasi sejak tahun 2000-an karena praktisi industri saat itu merasakan dampak yang positif dari penerapan metode kanban. Salah satu alasannya adalah penerapan metode kanban pada sebuah sistem tidak akan menyebabkan *overload* karena metode ini merupakan metode tarik (*pull system*) yang berarti bahwa tugas baru hanya bisa dikerjakan oleh sistem apabila terdapat sumber daya untuk mengerjakannya. Sebaliknya, pada sistem dorong (*push system*) tugas dimasukkan kedalam sistem sesuai permintaan sehingga terdapat peluang terjadinya *overload*. [31]

Dalam praktiknya, metode kanban membatasi jumlah tugas yang dikerjakan menggunakan alat fisik seperti kartu yang ditempelkan pada sebuah papan yang juga menjadikan metode ini menggunakan pendekatan visual. Meskipun demikian, dalam dunia pengembang perangkat lunak digunakan kartu virtual namun dengan tujuan yang sama. Perlu diketahui juga dalam penerapannya, metode kanban tidak secara eksplisit menyatakan jumlah limitasi tugas, melainkan angka tersebut ditentukan sendiri oleh kapasitas tim.

Implementasi metode kanban pada pengembangan perangkat lunak lebih mengedepankan pembatasan jumlah pekerjaan yang sedang dilakukan dan alur kerja yang dibentuk secara visual. Bentuk visual yang paling banyak dipilih adalah berbentuk papan kartu, baik yang berbentuk fisik ataupun digital. Visualisasi tersebut sebenarnya bukanlah metode kanban, karena setiap kartu yang dibuat tidak lagi melambangkan sebuah sinyal untuk mengerjakan pekerjaan baru, melainkan sebuah detail pekerjaan. Metode kanban yang diadaptasi menjadi sebuah metode visual dalam merepresentasikan kontrol alur kerja menjadikan setiap anggota tim dapat memvisualisasikan pekerjaan apa saja yang sedang dilakukan, mengatur diri sendiri, mengambil pekerjaan, serta memperbarui status pekerjaan secara mandiri. [31]

Dalam bukunya yang berjudul *Kanban: Successful Evolutionary Change for Your Technology Business*, David J. Anderson menerapkan visualisasi alur kerja menggunakan *kanban board* dengan alur sebagai berikut: *Input Queue, Analysis (In*

Progress), *Analysis (Done)*, *Dev Ready*, *Development (In Progress)*, *Development (Done)*, *Build Ready*, *Test*, dan *Release Ready*. Berikut adalah ilustrasi *kanban board* yang ditampilkan dalam buku tersebut.



Gambar 2.2. *Kanban Board*

Hal yang perlu digaris bawahi adalah penerapan visualisasi alur kerja yang terdapat pada *kanban board* harus berasal dari proses asli yang dilakukan dan dikembangkan oleh tim dan tidak boleh meniru alur dari tempat lain karena melakukan hal tersebut justru berlawanan dengan metodologi kanban. [32] Alur yang digambarkan pada *kanban board* juga tidaklah ditentukan oleh metode kanban itu sendiri, melainkan berasal dari proses yang sudah ada dan dilakukan oleh tim. [31] Umumnya, setelah alur kerja divisualisasikan, akan terlihat bagian yang menjadi *bottleneck*, dimana kartu unit kerja akan bertumpuk di salah satu langkah kerja. Setelah itu, dapat diterapkan metode pembatasan jumlah pekerjaan yang sedang dilakukan (*work in progress*). Pembatasan pekerjaan yang sedang dilaksanakan berarti membatasi jumlah tugas / unit kerja yang pada tiap tahap alur kerja. [32] Setelah itu, alur kerja yang digunakan bisa dikembangkan lebih lanjut seiring dengan waktu. Beberapa contoh improvisasi yang bisa dilakukan adalah dengan menambahkan antrian di bagian alur yang dianggap penting atau sebelum bagian yang terjadi *bottleneck*. [31] Sebagai contoh, apabila terlalu banyak unit kerja yang menumpuk di bagian *software in test*, maka bisa ditambahkan satu langkah baru sebelum bagian tersebut, seperti *ready to test*, sehingga tim penguji perangkat lunak bisa mengambil pekerjaan ketika ada kapasitas kerja yang bisa digunakan dan menghindari unit kerja bertumpuk secara berlebihan di suatu tempat.

2.7 Basis Data

Basis data atau yang biasa disebut dengan *database* adalah sekumpulan data yang terorganisir secara sistematis dan dapat diakses, dikelola, dan diperbarui dengan mudah. Pengembangan perangkat lunak pada penelitian ini akan membutuhkan basis data untuk menampung berbagai data yang dihasilkan dan dibutuhkan dari sistem ATEC, diantaranya seperti data berupa pertanyaan kuesioner, data akun pengguna, hingga data hasil uji ATEC. Data yang dibutuhkan dan dihasilkan oleh aplikasi akan memiliki relasi / hubungan satu sama lain, sebagai contoh pertanyaan kuesioner berelasi dengan data pengguna yang membuat, dan data hasil uji ATEC yang menggunakan kuesioner tersebut. Sifat data yang saling berelasi satu sama lain ini membuat sistem membutuhkan basis data yang mendukung data yang bersifat relasional, atau disebut dengan *relational database*. Basis data yang dipilih untuk mendukung aplikasi yang akan dibangun pada penelitian ini adalah PostgreSQL. PostgreSQL adalah sistem perangkat manajemen basis data (RDBMS) yang merupakan pengembangan lanjutan dari paket POSTGRES yang dikembangkan di Universitas Berkeley di California. [33] PostgreSQL memiliki dukungan yang sangat luas terhadap berbagai macam tipe data seperti tipe data standar SQL, *array*, JSON / JSONB, dan beragam tipe data lain. Dukungan komunitas yang luas serta variasi *plugin* yang dapat dipasang pada basis data ini menjadikan PostgreSQL mampu mendukung pengembangan beragam jenis perangkat lunak.

2.8 JavaScript Object Notation (JSON)

JSON (*JavaScript Object Notation*) adalah sebuah format pertukaran data berbasis teks yang ringan, mudah dibaca dan ditulis oleh manusia dan mesin yang berasal dari bahasa pemrograman JavaScript. [34] JSON banyak diadaptasi oleh beragam sistem yang digunakan sebagai format komunikasi antara klien dan *server* untuk saling bertukar data. JSON juga merupakan format yang lazim digunakan oleh

aplikasi berbasis API dalam bertukar data baik sebagai permintaan ataupun respons kepada pengguna. Dalam bertukar data dengan JSON juga dipermudah dengan dukungan tipe data yang bisa dianotasikan dalam format ini, mulai dari tipe data dasar seperti teks, angka, *boolean*, hingga tipe data kompleks seperti *array*, objek, dan sebagainya.

2.9 Unit Testing

Unit testing adalah sebuah tes otomatis yang mencakup verifikasi bagian kecil kode (*unit*) dengan cepat dan dilakukan dalam lingkungan yang terisolasi. [35] *Unit testing* umumnya dilakukan dengan cara memberikan serangkaian input terhadap unit kecil kode, biasanya berupa sebuah fungsi, untuk mengetahui perilaku fungsi terhadap input yang diterima. Dengan menggunakan *unit testing*, kita bisa percaya diri bahwa untuk setiap input yang diterima, fungsi akan bereaksi dengan keluaran yang dapat diprediksi. Sebagai contoh sebuah fungsi yang melakukan pembagian dua bilangan bertipe data integer dapat memberikan fungsi pembagi dengan benar, namun ketika input yang diberikan tidak sesuai (contoh: input 0) maka fungsi bisa memberikan keluaran yang dapat diprediksi semisal mengembalikan 0 atau mengembalikan galat yang terdefinisi dan tidak menyebabkan *crash*. Maka daripada itu, *unit testing* akan sangat berguna pada sistem ini untuk memastikan setiap unit terkecil kode memiliki perilaku yang terdefinisi untuk setiap rangkaian input yang diterima.

2.10 Black Box Testing

Black box testing adalah sebuah metode pengujian perangkat lunak berdasarkan kriteria kebutuhannya tanpa harus melihat kode internalnya. [36] Metode uji ini memberikan keleluasaan bagi penguji untuk mengukur kemampuan perangkat lunak dalam memenuhi tujuan dibuatnya perangkat lunak tersebut. Oleh karena

metode uji ini tidak memerlukan pemeriksaan kode internal perangkat lunak yang diuji, metode ini juga cocok dilakukan oleh pemula yang belum menguasai bahasa pemrograman tertentu. [37]

2.11 *Entity Relationship Diagram (ERD)*

ERD atau singkatan dari *Entity Relationship Diagram* adalah sebuah model visual yang lazim digunakan untuk mendeskripsikan entitas bisnis dan atributnya serta hubungan antar entitas satu dengan yang lainnya. [38] ERD akan sangat berguna dalam memvisualisasikan relasi antar tabel di basis data yang akan digunakan. Melalui ERD, kita juga bisa mengetahui tipe relasi antar tabel seperti relasi satu ke banyak, satu ke satu, ataupun banyak ke banyak.

2.12 Docker

Docker adalah sebuah platform *open source* yang digunakan untuk mengembangkan, distribusi, dan menjalankan aplikasi. [39] Docker memudahkan kita dalam melakukan instalasi aplikasi beserta semua *dependencies*-nya agar bisa dijalankan secara konsisten pada semua platform. Dengan menggunakan docker, kita bisa “membungkus” aplikasi kita ke dalam sebuah wadah sehingga menjadikan aplikasi yang dibangun bisa langsung dijalankan tanpa harus mengulang proses inisiasi secara terus menerus. Docker akan digunakan pada penelitian ini untuk dijadikan patokan terhadap semua *dependencies* aplikasi seperti modul-modul serta cara menjalankannya.

2.13 *JSON Web Token (JWT)*

JSON Web Token adalah sebuah token berbentuk *string* JSON yang sangat padat, berisi informasi terautentikasi dalam pertukaran informasi yang biasa ditempatkan di *header* dalam permintaan API. [40] JWT didesain untuk mendukung metode *signing* secara kriptografi menggunakan algoritma seperti *Hash-based Message Authentication Code* (HMAC) sehingga dapat menjamin integritas data tanpa harus bergantung dengan basis data. [41] Penggunaan JWT pada sistem autentikasi API dapat menyederhanakan proses autentikasi karena token hanya perlu sekali dibuat, dan dapat digunakan secara terus menerus tanpa bergantung dengan basis data untuk memvalidasinya. JWT juga dapat menyimpan data-data penting di dalamnya sebagai rujukan sistem ketika token tersebut digunakan. Perlu diperhatikan bahwa JWT tidak dienkripsi, sehingga data di dalamnya dapat dilihat dengan mudah, sehingga data-data sensitif tidak boleh disimpan di dalam JWT.

2.14 *Visual Studio Code*

Visual Studio Code atau biasa disingkat menjadi *vscode* adalah sebuah aplikasi yang berfokus pada pengembangan kode aplikasi yang didesain untuk mempermudah proses pengembangan aplikasi web, *mobile*, *cloud* yang tersedia untuk berbagai platform serta dukungan untuk diintegrasikan dengan beragam alat pendukung pengembangan perangkat lunak lainnya. [42] *Vscode* tidak hanya dapat melakukan manipulasi kode, namun juga bisa dikembangkan dan diintegrasikan dengan aplikasi lain untuk mendukung proses pengembangan perangkat lunak, salah satunya adalah dengan cara memasang ekstensi tambahan untuk *vscode*. *Vscode* memiliki situs terintegrasi bernama *Visual Studio Marketplace* yang berisi lebih dari 7000 ekstensi yang tersedia untuk digunakan oleh penggunanya. [43] Instalasi *vscode* bisa dilakukan pada situs resminya, <https://code.visualstudio.com>.

2.15 Ubuntu

Ubuntu adalah sebuah sistem operasi turunan dari distribusi sistem operasi berbasis Linux yang dikembangkan oleh Mark Shuttleworth yang bersifat ramah pengguna awam, tanpa biaya, serta bebas digunakan oleh siapapun. [44] Ubuntu sendiri terbagi menjadi beberapa komponen, mulai dari *kernel* yang menjadi penghubung komunikasi antara perangkat lunak dan perangkat keras, utilitas berbasis *command-line*, serta gabungan dari bermacam-macam aplikasi yang bersifat bebas digunakan (*free software*) sehingga penggunaanya bisa langsung menggunakan ubuntu tanpa perlu modifikasi yang rumit. [45] Ubuntu dapat dengan mudah diinstall di beragam perangkat, mulai dari laptop, komputer, *raspberry pi*, serta disediakan secara luas oleh beragam *cloud provider*. Proses instalasi, dokumentasi serta dukungan terkait penggunaan sistem operasi ini dapat ditemukan di situs resminya, <https://ubuntu.com/>.

2.16 Insomnia

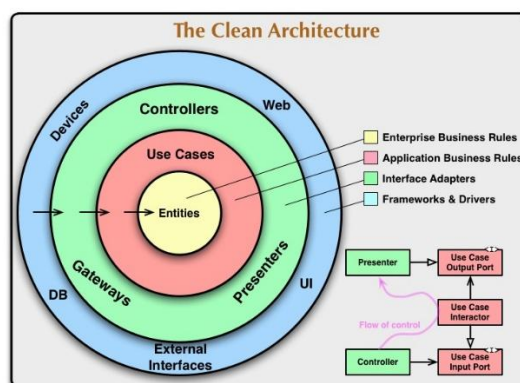
Insomnia adalah sebuah aplikasi berbasis *desktop* yang bersifat *open source* yang berfungsi sebagai alat bantu melakukan proses design, perbaikan dan pengujian API. Insomnia memiliki antarmuka pengguna yang mudah digunakan. Tidak hanya itu, insomnia juga terintegrasi dengan fungsionalitas canggih seperti pembuat kode otomatis, konfigurasi lingkungan, serta alat bantu otentikasi. Insomnia memberikan sebuah cara mudah untuk melakukan interaksi dengan API tanpa harus menggunakan utilitas berbasis teks, sehingga pekerjaan terkait API bisa dilakukan lebih cepat dan mudah. Aplikasi insomnia dapat diunduh di situs resminya, <https://insomnia.rest>.

2.17 Swaggo

Swaggo adalah sebuah modul berbasis bahasa golang yang dapat membuat dokumentasi REST API menggunakan swagger dari anotasi kode golang. Swaggo juga bisa diintegrasikan dengan beragam modul / *framework* berbasis HTTP yang populer, seperti gin, echo, go-fiber, dan lain sebagainya. Penggunaan swaggo juga dapat diatur untuk memberikan sebuah halaman web yang berisi dokumentasi API sekaligus alat untuk berinteraksi dengan API tersebut. Swaggo juga bersifat *open source*, kode sumbernya dapat diakses di halaman github swaggo, <https://github.com/swaggo/swag>. Dokumentasi cara penggunaan hingga instalasinya pun bisa di akses di halaman tersebut.

2.18 Clean Architecture

Clean architecture adalah konsep yang dipopulerkan oleh Robert C. Martin pada bukunya yang berjudul *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. *Clean architecture* adalah sebuah rangkaian petunjuk dalam mendesain dan mengembangkan arsitektur sebuah sistem komputer untuk dapat mencapai kebutuhan bisnis serta meminimalisir kebutuhan sumber daya manusia yang dibutuhkan untuk membangun dan memelihara sistem tersebut. [46] Rangkaian umum dalam menerapkan *clean architecture* ke dalam sebuah sistem komputer digambarkan melalui diagram di bawah ini.



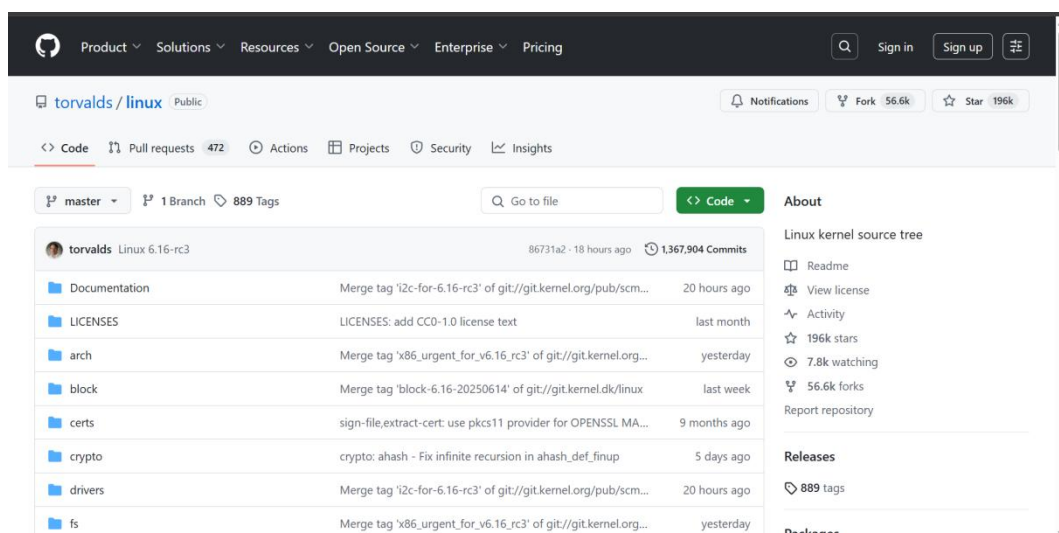
Gambar 2.3. *Clean Architecture* (sumber: <https://blog.cleancoder.com>)

Gambar di atas memvisualisasikan bagaimana setiap lapisan aplikasi bertindak dan berinteraksi satu sama lain. Berdasarkan gambar tersebut, terdapat sebuah aturan yang disebut dengan *dependency rule* yang berisi ketergantungan kode sumber hanya boleh bergantung pada lapisan yang lebih dalam. [46] Sebagai contoh, dalam membangun sebuah sistem, misalnya saja sebuah situs web, harus fokus membangun aturan bisnis dan tidak ketergantungan terhadap teknologi tertentu. Artinya, ketika mendesain situs web tersebut, sebagai seorang arsitek sistem komputer, kita harus bisa leluasa memilih teknologi apapun yang bisa memenuhi kebutuhan bisnis. Apabila ketika melakukan desain sudah ketergantungan dengan sistem basis data ataupun modul kerangka kerja tertentu, bisa dikatakan bahwa arsitektur sistem tersebut tidak memenuhi prinsip *clean architecture*. Arsitektur sistem yang baik juga harus mendukung setidaknya 4 hal, *use case* dan operasional sistem, perawatan, pengembangan serta *deployment* dari sistem tersebut. [46]

Arsitektur sistem yang baik sebenarnya tidak hanya mengatur bagaimana sistem bekerja dari sudut pandang secara luas, namun juga mengatur bagaimana komponen terkecil sistem, seperti fungsi dan kelas pada kode, berinteraksi satu dengan yang lain. Terdapat sebuah prinsip yang juga menjadi acuan dalam hal tersebut, prinsip tersebut adalah prinsip SOLID. Kata SOLID sendiri merupakan sebuah akronim yang terdiri dari *Single Responsibility*, *Open-Closed principle*, *Liskov Substitution principle*, *Interface Segregation*, dan *Dependency Inversion principle*. Prinsip *Single Responsibility* menyatakan bahwa sebuah modul hanya boleh bertanggung jawab kepada satu dan hanya satu aktor pada sistem. Prinsip *Open-Closed* mengatur bahwasanya perangkat lunak harus terbuka untuk ekstensi, namun tertutup untuk modifikasi. Prinsip *Liskov Substitution* mengatur tentang proses pembuatan sub-tipe dari setiap modul / kelas. Prinsip *Interface Segregation* menyatakan bahwa sebuah modul hanya boleh bergantung pada modul lain yang digunakan secara langsung. Terakhir, prinsip *Dependency Inversion* menyatakan bahwa kode sumber yang fleksibel harus bergantung pada abstraksi, dan bukan tipe konkrit.

2.19 Git dan Github

Git adalah sebuah alat pengendali versi yang terdesentralisasi yang dikembangkan oleh Linus Torvald dan bersifat bebas dan terbuka untuk digunakan. [47] Git dapat bekerja secara mandiri di lingkungan lokal dan tidak bergantung pada sistem luar. Sedangkan github adalah sebuah repositori git berbasis web yang tersentralisasi. Git dapat diinstal di banyak sistem operasi, mulai dari sistem operasi Windows, Linux dan juga macOS. Baik git maupun github bisa digunakan secara independen untuk mengelola kode program. Keduanya juga bisa digunakan bersamaan sehingga memberikan keleluasaan bagi beberapa orang sekaligus untuk bekerja pada repositori kode yang sama. Beberapa perintah dasar git yang lazim digunakan seperti perintah menambahkan berkas ke repositori (*git add nama_file.ext*), menyimpan perubahan dan pesan perubahan ke repositori (*git commit -m "pesan perubahan"*) hingga perintah menyimpan perubahan ke repositori pusat (*git push origin nama_branch*). Selain menyediakan repositori terpusat, Github juga menyediakan beberapa fitur yang amat berguna, seperti *Github Actions*, *Ticketing System*, serta *Github Wiki*. Berikut adalah contoh sebuah repositori kode yang tersimpan di github.



Gambar 2.4. Contoh Repositori Kode Github

2.20 Penelitian Terkait

A TEC adalah sebuah metode yang digunakan untuk mengukur kemandirian penanganan dan perkembangan individual terhadap autisme yang dikembangkan pada tahun 1990 oleh Bernard Rimland, Ph.D. dan Stephen M. Edelson, Ph.D.. [21] Metode A TEC dikembangkan untuk dilakukan oleh orang tua, guru, ataupun pengasuh anak penderita autisme yang berbentuk kuesioner satu halaman berisi 77 item yang meliputi kemampuan komunikasi, sosial, sensorik, kesadaran kognitif, kesehatan serta perilaku fisik. [19] Hasil dari penggunaan metode A TEC pada anak penderita autisme berupa skor yang berbanding terbalik atas kemandirian penanganan autisme, dimana skor yang lebih rendah menunjukkan peningkatan kemandirian. [48]

Meskipun metode A TEC ditujukan untuk dilakukan oleh orang tua yang mungkin tidak terlatih secara medis terkait penanganan dan diagnose autisme, metode ini terbukti mampu menghasilkan skor yang berkorelasi tinggi dengan metode diagnosa autisme yang dilakukan oleh tenaga medis profesional, yaitu metode *Childhood Autism Rating Scale* (CARS). Dari keempat domain pada metode A TEC, domain sensorik / kesadaran kognitif adalah domain yang paling tinggi berkorelasi dengan hasil dari metode CARS, diikuti dengan domain kemampuan komunikasi, kemudian domain kemampuan sosial, dan terakhir adalah domain kesehatan dan perilaku fisik. [18] Sebagai catatan, perlu diketahui bahwa korelasi yang paling rendah dari metode A TEC dan CARS yang berada di domain kesehatan dan perilaku fisik disebabkan karena metode CARS dan sebagian besar metode lain yang serupa tidak memiliki bagian khusus yang mencakup masalah kesehatan dan kondisi fisik.

Penelitian lain juga menunjukan bahwasanya metode A TEC bisa menjadi instrumen yang andal dalam memonitoring perkembangan anak yang menderita autisme. Metode A TEC juga mampu menjadi instrumen untuk mengumpulkan informasi terkini tentang berbagai perilaku dan keterampilan pada anak-anak dengan autisme yang memberikan hasil yang andal dan valid tanpa membutuhkan sumber daya

yang mahal. Dengan demikian, ATEC sangat berpotensi untuk menjadi instrumen dalam memberikan gambaran umum anak penderita autisme sebagai pengukuran rutin disamping instrumen formal lainnya. [19]

Metode ATEC yang mengandalkan orang tua / pengasuh anak penderita autisme yang dapat dilakukan secara rutin dapat membantu mengurangi biaya uji klinis untuk mendiagnosa autisme. Selain daripada itu, data yang dihasilkan secara rutin dari metode ini juga membuat prosedur klinis bisa dilakukan secara berkesinambungan serta meningkatkan jumlah partisipan. Meskipun metode ATEC, atau bahkan metode lain yang serupa belum ada yang dapat menghasilkan hal semacam itu, sebuah penelitian pada tahun 2018 memberikan gambaran kemampuan metode ATEC dalam memprediksi skor uji anak penderita autisme berdasarkan usia. Penerapan metode ATEC yang dilakukan secara masal dan dalam kelompok partisipan yang besar dan beragam dapat menggambarkan kemanjuran observasi psikometrik yang diberikan oleh orang tua serta menjadi alternatif dari penggunaan tenaga medis spesialis untuk melakukan pengujian terhadap anak penderita autisme. [16]

Implementasi metode ATEC berbasis API yang bisa diintegrasikan dengan paradigma multi-platform serta tidak mengekang isi kuesioner baik dari segi bahasa yang digunakan pada setiap pertanyaan, ataupun adaptasinya di negara tertentu saja memungkinkan adaptasi dan pengumpulan data dalam skala luas dapat dilakukan. Hal ini senada dengan hasil penelitian yang berfokus pada mengulas literatur terhadap sistem pakar diagnosa autisme dan aplikasi mobile yang menghasilkan kesimpulan berupa kebutuhan terhadap pengembangan alat yang tidak hanya dapat mendiagnosa anak yang mengidap autisme, namun juga tingkat keparahan gejala autisme yang diderita. Penelitian yang sama juga menyimpulkan adanya kebutuhan terhadap pengembangan aplikasi yang dapat melakukan pengumpulan data sekaligus aplikasi yang mampu dikembangkan untuk berbagai bahasa yang ada. [49]

Pengembangan aplikasi dalam penelitian ini yang menggunakan beragam *best practices* serta menggunakan teknologi yang mendukung aplikasi yang andal serta

mendukung skalabilitas menjadi seirama dengan segudang keunggulan metode ATEC itu sendiri. Pengembangan aplikasi dengan mengadopsi metode kanban dalam *Software Development Life Cycle* (SDLC) yang termasuk ke dalam metode *Agile* dapat mendukung dalam mempermudah pengembangan aplikasi bahkan yang kompleks dan dinamis. [50]

Tabel 2.1. Penelitian Terkait

No	Peneliti	Objek	Metode / Alat	Hasil
1.	Amir Hossein Memari dkk. [20]	Adaptasi, keandalan dan validitas ATEC di Iran.	<i>Cronbach's alpha</i> dan <i>Guttman split-half</i> .	Metode ATEC yang diadaptasi ke versi Iran dapat diandalkan dan valid dalam mengevaluasi gejala autisme.
2.	David A. Geier, Janet K. Kern, Mark R. Geier. [18]	Membandingkan metode ATEC dan CARS untuk evaluasi kuantitatif autisme.	Spearman's Correlation.	Hasil metode ATEC dan CARS memiliki korelasi yang signifikan.
3.	Magiati dkk. [19]	Pengujian kegunaan metode ATEC untuk memonitoring perkembangan anak penderita autisme.	Cronbach's alpha.	ATEC merupakan calon alat monitoring perkembangan autisme yang valid dan andal.
4.	Shreyas Mahapatra dkk. [16]	Skor ATEC seiring berjalannya usia anak.	Perubahan skor ATEC terhadap usia anak.	Skor ATEC dapat memberi prediksi terhadap usia dan skor yang akan dihasilkan.

Tabel 2.1. Penelitian Terkait (lanjutan)

No	Peneliti	Objek	Metode / Alat	Hasil
5.	Kanitha Sunakarach & Pattapong Kessomboon. [22]	Validitas dan keandalan ATEC versi Thailand.	<i>Forward-backward translation.</i>	Validitas dan keandalan yang tinggi untuk ATEC versi Thailand.
6.	Parman Suparman & Miftachul Huda. [51]	Pengembangan sistem manajemen skripsi dengan Kanban.	<i>Codeigniter</i> , <i>black box testing</i> , dan Kanban	Sistem manajemen skripsi bisa dibangun menggunakan <i>Codeigniter</i> dan Kanban.
7.	Sigit Priadi & Moh Idris. [52]	Membangun <i>Web Service</i> modul berita aplikasi Satudikti.	Golang, REST API.	<i>Web Service</i> modul berita aplikasi Satudikti dapat dibangun dan dipermudah dengan menggunakan Golang.
8.	Rieke Cahya dkk. [53]	Dampak Hambatan Geografis dan Strategi Akses Pelayanan Kesehatan.	Preferred Reporting Items for Systematic Review and Meta-Analyses (PRISMA).	Hambatan geografis terbukti dapat mempengaruhi status kesehatan masyarakat dan diperparah dengan sistem fasilitas kesehatan yang tidak dirancang untuk dijangkau semua orang.
9.	Muhammad Fachrudin Thohari dkk. [54]	Pengembangan API aplikasi Wvaccine.	Golang, <i>Echo</i> , <i>black box testing</i> .	API berhasil dibangun dengan nilai <i>System Usability Scale</i> (SUS) 82.7.

Berdasarkan tabel di atas, dapat diketahui bahwa metode ATEC memiliki validitas yang cukup tinggi dan dapat diandalkan. Bahkan, metode ini memiliki korelasi yang kuat dengan metode serupa yang dilakukan oleh ahli. Sebagai contoh, penelitian yang dilakukan oleh Amir Hossein Memari dkk dengan objek penelitian untuk mengukur keandalan dan validitas metode ATEC di Iran dengan metode *Cronbach's alpha* dan *Guttman split-half* menghasilkan kesimpulan bahwa metode ATEC dapat diadaptasi ke budaya Iran dan tetap menghasilkan data yang valid dan andal. Hal senada juga didapatkan dari penelitian yang dilakukan oleh David A. Geier, Janet K. Kern, Mark R. Geier yang bertujuan untuk membandingkan metode ATEC dengan metode CARS menggunakan metode Spearman's Correlation dan menghasilkan kesimpulan bahwasanya hasil metode ATEC memiliki korelasi yang tinggi dengan metode CARS.

Penelitian yang dilakukan oleh Magiati dkk yang bertujuan untuk menguji penggunaan metode ATEC untuk melakukan monitoring perkembangan anak penderita autisme dengan metode penelitian Cronbach's alpha menyimpulkan bahwasanya metode ATEC merupakan calon alat monitoring yang valid dan andal. Tidak hanya itu, metode ATEC juga dapat memberikan prediksi terhadap usia dan skor ATEC yang dihasilkan yang bersumber dari hasil penelitian Shreyas Mahapatra dkk yang bertujuan untuk mendapatkan perubahan skor ATEC seiring berjalannya usia. Metode ATEC juga terbukti memiliki adaptabilitas yang tinggi yang dibuktikan dari penelitian yang dilakukan oleh Kanitha Sunakarach & Pattapong Kessomboon untuk mengetahui validitas dan keandalan ATEC versi Thailand menggunakan metode *forward-backward translation* menyimpulkan bahwasanya metode ATEC memiliki keandalan dan validitas yang tinggi dalam budaya Thailand.

Penelitian selanjutnya yang berkaitan dengan topik penelitian ini adalah penelitian yang dilakukan oleh Parman Suparman & Miftachul Huda yang bertujuan untuk melakukan pengembangan sistem manajemen skripsi menggunakan metode kanban menggunakan metode *codeigniter*, *black box testing*, dan kanban yang menyimpulkan metode kanban dan metode uji *black box* dapat membangun sistem manajemen skripsi. Bahasa pemrograman golang serta arsitektur REST API juga

terbukti mampu digunakan untuk membangun *Web Service* modul berita aplikasi Satudikti yang terbukti dari penelitian yang dilakukan oleh Sigit Priadi & Moh Idris.

Penelitian yang dilakukan oleh Rieke Cahya dkk yang bertujuan untuk mengetahui dampak hambatan geografis terhadap akses ke fasilitas kesehatan membuktikan bahwa hambatan geografis terbukti dapat mempengaruhi status kesehatan masyarakat, sehingga fasilitas kesehatan yang dapat diakses dengan mudah akan sangat menguntungkan. Penelitian yang dilakukan oleh Muhammad Fachrudin Thohari dkk yang dilakukan untuk API aplikasi fasilitas kesehatan Wvaccine menggunakan golang serta metode uji *black box* yang menghasilkan sistem API berhasil dibangun dengan skor *usability* yang tinggi juga menunjukkan kapabilitas golang dalam membangun sistem yang terkait dengan fasilitas kesehatan.

III. METODOLOGI PENELITIAN

3.1 Waktu dan Tempat Penelitian

Skripsi ini dilaksanakan dari bulan Februari hingga April 2025 yang bertempat di Universitas Lampung. Penelitian ini dilakukan selama 3 bulan dengan jadwal penelitian sebagai berikut:

Tabel 3.1. Jadwal Penelitian

Kegiatan	Bulan											
	Maret				April				Mei			
	1	2	3	4	1	2	3	4	1	2	3	4
Studi Literatur												
<i>Requirements Gathering</i>												
SDLC Kanban												
Penulisan Laporan												

3.2 Perangkat Penelitian

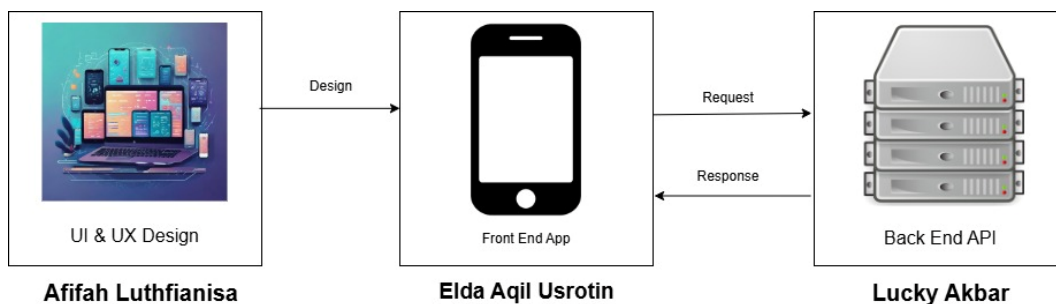
Berikut ini adalah perangkat penelitian berupa perangkat lunak dan perangkat keras yang digunakan pada penelitian ini:

Tabel 3.2. Perangkat Penelitian

No	Perangkat	Spesifikasi	Kegunaan
1.	Laptop	Intel i5 1165g7	Perangkat pengembangan dan pengujian aplikasi.
2.	Visual Studio Code	Versi 1.97.2	Perangkat lunak yang bertindak sebagai <i>code editor</i> .
3.	Sistem Operasi	Ubuntu 24.04.4 LTS	Sistem operasi perangkat pengembangan dan pengujian aplikasi
4.	Go	Versi 1.23.6	Bahasa pemrograman untuk mengembangkan aplikasi.
5.	Insomnia	Versi 10.1.1	Perangkat lunak yang bertindak sebagai pengujian <i>request</i> dan <i>response</i> dari aplikasi
6.	Postgresql	Versi 14	Basis data primer aplikasi
7.	Redis	Versi 7	Basis data sekunder aplikasi
8.	Git & github	<i>Latest version</i>	<i>Version control system</i> (VCS).
9.	Swaggo	<i>Version 1.16.4</i>	Pembuat dokumentasi API otomatis
10.	Jira	-	<i>Website</i> untuk melakukan pelacakan pengerjaan tugas dengan <i>kanban board</i> .

3.3 Tim Penelitian

Penelitian ini akan dilakukan oleh tim yang terdiri dari 3 orang yang secara individu mengerjakan bagian aplikasi terpisah namun saling terhubung. Bagian aplikasi yang dimaksud adalah *user interface* (UI) & *user experience* (UX) *design*, *front end development*, dan *back end development*. Masing-masing anggota tim akan fokus mengerjakan bagiannya. Satu pengembang di bagian UI & UX Afifah Luthfianisa, satu pengembang di bagian *front end* yaitu Elda Aqil Usrotin, dan satu pengembang API *back end* yaitu Lucky Akbar.



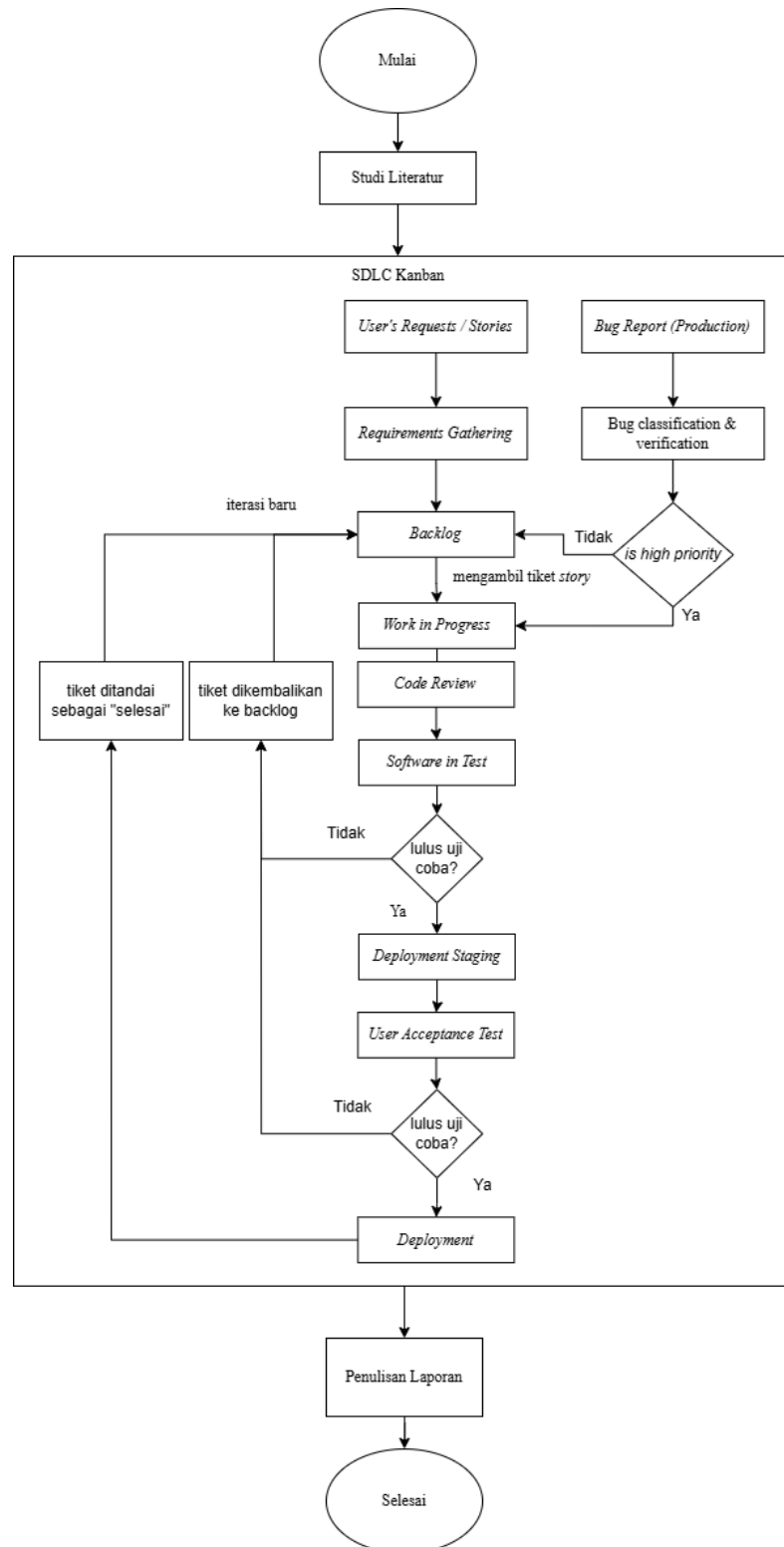
Gambar 3.1. Tim Pengembang

Diagram di atas menunjukkan proses komunikasi antar bagian aplikasi satu sama lain. Bagian *UI & UX design* akan menghasilkan sebuah desain yang juga bertindak sebagai *blue print* yang kemudian akan dijadikan patokan oleh *front end developer* dalam membuat tampilan yang akan dilihat oleh pengguna. Sedangkan bagian *back end development* akan menghasilkan sebuah *API server* yang bertindak sebagai penyedia data dan komputasi pemrosesan data untuk dikonsumsi oleh pengguna melalui aplikasi.

3.4 Tahapan Penelitian

Dalam pengembangan aplikasi yang menjadi inti utama penelitian ini terdapat beberapa tahapan yang akan dilakukan. Tahapan pertama adalah melakukan studi literatur. Tahapan ini amat penting dilakukan untuk mengetahui secara lebih detail mengenai topik utama penelitian, yaitu metode ATEC serta topik lain yang terkait. Tahapan selanjutnya adalah melakukan pengembangan perangkat lunak dengan metode kanban. Sebagaimana telah dijelaskan sebelumnya, tahapan penerapan metode kanban adalah dengan mengikuti proses asli yang sudah berjalan di dalam tim pengembang. Termasuk juga di dalamnya tim / bagian di luar tim pengembang yang memiliki pengaruh terhadap berjalannya pengembangan, salah satunya adalah pengguna. Terakhir, akan masuk ke tahapan penulisan laporan.

Agar lebih mempermudah memvisualisasikan tahapan pengembangan aplikasi secara keseluruhan, berikut adalah diagram alir dari tahapan penelitian yang akan dilakukan.



Gambar 3.2. Tahapan Penelitian

3.3.1 Studi Literatur

Tahapan penelitian studi literatur dilakukan dengan cara mempelajari sumber sumber terkait topik penelitian terkait yang bersumber dari jurnal, buku, ataupun halaman web. Tahapan ini dilakukan dengan tujuan meningkatkan pemahaman terhadap semua topik penelitian agar memperkuat pemahaman teori, kerangka kerja serta menambah pengetahuan yang relevan. Tahapan ini juga akan dilakukan proses identifikasi terkait *stakeholder* atau aktor yang akan terkait dan berinteraksi dengan sistem yang akan dibangun.

3.3.2 *Software Development Life Cycle* dengan *Kanban*

Pada proses pengembangan perangkat lunak, metode kanban dipilih menjadi metode pengembangan proses *Software Development Life Cycle* (SDLC). Sesuai dengan prinsip utama dari metode kanban, visualisasi alur kerja adalah kunci untuk implementasi metode kanban yang baik. Karena alur penelitian sekaligus tahapan kerja yang akan dilakukan sudah tertera di bagian sebelumnya, maka visualisasi alur kerja pada *kanban board* akan merefleksikan alur penelitian tersebut. Tahapan SDLC dengan kanban ini juga terdiri dari beberapa tahapan, yakni *user's requests / stories, requirement gathering, bug report, bug classification & verification, backlog, work in progress, code review, software in test, deployment staging, user acceptance test, dan deployment production*. Masing masing dari tahapan ini akan berperan penting untuk memastikan setiap fitur yang dijalankan akan bisa maksimal serta sesuai dengan kebutuhan penggunaanya.

3.3.2.1 *User's Requests / Stories*

Ketika pengguna memberikan sebuah permintaan terhadap sebuah fitur maka permintaan tersebut harus diterima oleh tim pengembang terlebih dahulu agar bisa kemudian diproses dan ditentukan apakah bisa dikerjakan atau tidak. Umumnya, setiap permintaan dari pengguna bisa ditranslasikan sebagai permintaan terhadap sebuah fitur. Maka dari itu, permintaan pengguna akan diproses di tahapan selanjutnya yaitu *requirements gathering*. Pada bagian pembuatan *user's stories*, penting adanya penggunaan perspektif pengguna dalam membangun cerita yang menggambarkan kebutuhan serta tujuan pengguna dalam menggunakan sistem. Format khusus membangun *stories* adalah sebagai berikut: “sebagai [jenis pengguna], [kebutuhan] supaya [tujuan]”. Format ini penting digunakan karena perspektif pengguna adalah kunci dalam membuat *stories* yang tepat. Setiap *story* yang dibuat akan dijadikan sebagai induk dari tugas selama pengembangan aplikasi serta akan memiliki bagian tugas yang merupakan bagian pendukung dalam mewujudkan *story* tersebut. Setiap *story* yang didapat akan masuk ke tahap *requirement gathering* yang kemudian akan masuk ke *backlog* untuk selanjutnya dikembangkan lebih lanjut di dalam aplikasi.

3.3.2.2 *Requirements Gathering*

Tahapan *requirements gathering* berguna untuk mengumpulkan semua komponen dari aplikasi yang akan dipenuhi selama proses SDLC. Pada tahap ini, kumpulan dari *user's stories* yang sudah didapat akan ditelaah guna mendapatkan kebutuhan apa-apa saja dari sisi teknis yang diperlukan untuk mewujudkan kebutuhan pengguna di dalam *story* tersebut. Pada tahapan ini akan menghasilkan beberapa keluaran kebutuhan, yaitu kebutuhan fungsional, pengumpulan kebutuhan non fungsional, desain *usecase diagram*, desain basis data, desain arsitektur sistem, identifikasi *sub tasks*, serta *API contract*.

a. Kebutuhan Fungsional

Kebutuhan fungsional adalah komponen yang amat penting dalam pengembangan perangkat lunak, dimana kebutuhan fungsional menggambarkan fitur-fitur yang akan dimiliki oleh sistem. Penentuan kebutuhan fungsional akan dilakukan dengan cara mendeskripsikan fungsionalitas apa saja yang harus disediakan oleh sistem. Hasil dari identifikasi kebutuhan fungsional sistem adalah nama kebutuhan itu sendiri serta deskripsi-deskripsi terkait dengan *user's stories* yang telah terkumpul.

b. Kebutuhan non-Fungsional

Kebutuhan non fungsional dari sebuah aplikasi erat kaitannya dengan cara kerja sistem dan karakteristik, serta kualitas kinerja dari sistem yang akan dibangun itu sendiri. Penentuan kebutuhan non fungsional akan didasarkan pada kebutuhan fungsionalnya yang kemudian diidentifikasi karakteristik sistem atau kualitas kinerja apa yang dibutuhkan agar kebutuhan fungsional tersebut bisa berjalan dengan baik. Sebagai contoh adalah verifikasi PIN ATM yang membutuhkan kebutuhan non-fungsional yaitu keamanan supaya PIN yang disimpan oleh sistem ada dalam bentuk yang terenkripsi.

c. *Use Case Diagram*

Dalam membangun *use case diagram*, penting untuk menangkap relasi antar kemampuan atau fungsi yang disediakan sistem dengan aktor terkait. *Use case diagram* sangat berguna untuk memberikan visualisasi penggunaan kapabilitas sistem dari sudut pandang pengguna yang dinotasikan sebagai aktor. Selain itu, diagram ini juga bermanfaat untuk menangkap hubungan antar *use case*. Pada pemodelan ini, sistem akan dianggap sebagai sebuah *black box*, dimana sistem hanya digambarkan sebagai apa yang mampu dikerjakannya, dan bukan bagaimana sistem mengerjakannya. Penentuan pemodelan *use case diagram* didasarkan pada kebutuhan fungsional sistem serta aktor yang terkait dan dapat menggambarkan relasi antar *use case* dengan relasi *include* ataupun *extend*. Sehingga, akan dihasilkan sebuah diagram yang menangkap relasi antar aktor serta interaksinya dengan kebutuhan fungsional alias fitur yang disediakan oleh sistem.

d. Desain Basis Data

Tahapan desain basis data adalah sebuah tahap melakukan translasi kebutuhan pengguna ke dalam sebuah desain basis data untuk memenuhi kebutuhan pengguna tersebut. Tahapan ini akan menghasilkan sebuah fondasi yang akan mendukung fungsionalitas aplikasi yang menggunakannya. Tahapan ini akan dimulai dengan melakukan identifikasi entitas yang berasal dari keseluruhan hasil analisis dari *user's stories* yang kemudian masing-masing entitas itu diidentifikasi kembali atribut pendukungnya serta relasi antar entitas. Hasil dari tahapan ini adalah sebuah diagram ERD yang akan menangkap semua entitas dan relasi antar entitas yang berguna untuk memenuhi semua kebutuhan pengguna.

e. Arsitektur Sistem

Pembuatan desain arsitektur sistem akan menggunakan prinsip *clean architecture*. Arsitektur sistem akan digambarkan menggunakan diagram yang dapat menangkap relasi antara sistem ATEC API dengan menampilkan komponen / *layer* internal sistem serta komponen eksternal sistem, seperti basis data, serta perangkat pengguna. Pada desain arsitektur tersebut juga akan digambarkan hubungan antara komponen internal sistem, serta metode komunikasi sistem dengan perangkat pengguna.

Proses pembuatan desain arsitektur sistem akan terlebih dahulu dilakukan identifikasi komponen internal sistem, seperti *use case*, basis data, serta antarmuka sistem. Kemudian, relasi antar komponen tersebut akan diperiksa untuk disesuaikan dengan prinsip *Dependency Inversion Principle* (DIP). Terakhir, akan digambarkan bagaimana hubungan antar sistem dengan sistem luar seperti basis data dan perangkat pengguna serta metode komunikasinya.

f. Sub tasks

Dari setiap *user's story* yang didapat selanjutnya diidentifikasi komponen-komponen pengerjaan teknis yang diperlukan untuk mewujudkan *story* tersebut ke dalam sistem API. Komponen tersebut merupakan suatu unit pekerjaan yang menyusun keseluruhan detail teknis dalam mewujudkan sebuah *story* dan bersifat

mandiri antar komponen, artinya setiap komponen tersebut bisa dikerjakan secara bersamaan dan pengerjaannya tidak saling ketergantungan. Hal ini dimungkinkan salah satunya karena penggunaan arsitektur sistem yang baik. Maka dari itu, proses penentuan *sub task* untuk setiap *story* dilakukan dengan analisa dari sudut pandang arsitektur sistem, seperti pengerjaan apa yang akan berada di modul *use case*, modul *database access layer*, ataupun modul REST. Hasil yang diharapkan adalah komponen tugas teknis dari masing-masing *story* dapat teridentifikasi dan dapat dikerjakan secara mandiri oleh anggota tim pengembang.

g. *API Contract*

API Contract merupakan sebuah rancangan permintaan dan respons dari API terhadap penggunaanya. *API Contract* berisi detail jenis masukan, jenis keluaran serta metode permintaan yang harus dipenuhi oleh pengguna agar sistem dapat memberikan hasil yang sesuai. Sesuai namanya, *API Contract* adalah perjanjian antara sistem dengan pengguna serta menjadi rujukan sifat sistem terhadap permintaan yang diterima. Pembuatan *API contract* mungkin tidak secara keseluruhan dilakukan untuk tiap-tiap *story*. Inti dari pembuatan *API contract* adalah untuk mengomunikasikan masing-masing fitur yang disediakan sistem yang berisi *endpoint* untuk mengakses fitur tersebut, metode *HTTP request* yang harus digunakan, jenis masukan, ekspektasi keluaran hingga ekspektasi pesan galat yang mungkin diberikan oleh sistem. *API contract* yang akan dibuat juga harus didasarkan pada arsitektur sistem, khususnya arsitektur REST. *API contract* juga akan dibangun dengan tujuan mempermudah bagian *front end* untuk bisa berkomunikasi dengan bagian *back end* bahkan sebelum fitur-fitur tersebut selesai dikembangkan. Dengan demikian, *API contract* juga bisa mempercepat laju integrasi antara *client-server*.

3.3.2.3 Bug Report

Bug report adalah sebuah laporan yang berdasarkan dari adanya anomali, kesalahan, ataupun kekurangan dari API yang telah melewati tahapan *deployment production* yang berarti sudah bisa diakses secara publik. Terjadinya kesalahan yang telah sampai ke tangan pengguna menjadikan kesalahan ini berbeda dengan kesalahan yang terdeteksi pada tahapan pengujian oleh tim penguji perangkat lunak ataupun ketika tahapan *user acceptance test*. Setiap laporan yang diterima harus dicatat dan dimasukkan ke dalam sistem *kanban board*, supaya dapat ditindaklanjuti di tahap berikutnya. Pada tahap ini apabila terdapat laporan bug, selain harus dicatat kode tiket laporan bug tersebut, harus juga ditelaah *story* mana atau tiket *story* mana yang terkait supaya bisa membuat tiket *bug* tersebut terhubung dengan fitur tempat terjadinya kesalahan tersebut.

3.3.2.4 Bug Classification & Verification

Proses klasifikasi dan verifikasi terhadap sebuah laporan *bug* merupakan tahapan penting. Tahapan ini memberikan penentuan apakah *bug* tersebut memanglah sebuah kesalahan dari sistem, atau bukan. Kemudian, tahapan ini juga menentukan tingkat keparahan dari *bug* tersebut. Langkah pertama yang harus dilakukan di tahap ini adalah melakukan reproduksi dari laporan *bug* guna memverifikasi bahwasanya memang *bug* tersebut terjadi pada sistem. Apabila terverifikasi, perlu diadakan analisis untuk mengetahui apa yang menyebabkan masalah tersebut. Setelah diverifikasi, selanjutnya masuk ke tahap klasifikasi, di mana tahapan ini akan menentukan apakah *bug* yang terjadi merupakan *bug* dengan skala prioritas tinggi ataupun rendah serta solusi untuk penyelesaiannya. Ada dua syarat yang apabila salah satunya terpenuhi, maka *bug* tersebut akan memiliki skala prioritas tinggi, yaitu apabila *bug* terjadi di fitur kritikal sistem atau *bug* menghambat laju alur pengembangan sistem.

3.3.2.5 Backlog

Backlog adalah tempat dimana semua pekerjaan yang akan dilakukan akan ditempatkan. Setiap anggota tim yang akan mengerjakan tugas akan mengambil sebuah tugas dari *backlog* dan kemudian mengerjakan tugas tersebut. *Backlog* bersifat dinamis, artinya isi *backlog* dapat ditambahkan seiring perjalanan proyek semisal ada kebutuhan baru, ataupun terdapat umpan balik dari pihak terkait. Isi dari *backlog* adalah *user's stories* yang telah melalui tahapan *requirement gathering* dan berisi cetak biru sebagai acuan implementasinya. *Backlog* juga bisa berasal dari sebuah laporan *bug* dari lingkungan *production* yang bersifat prioritas rendah, alias tidak menjadi penghalang bagi berjalannya pengembangan.

3.3.2.6 Work in Progress

Setelah *backlog* selesai ditentukan, pengembangan perangkat lunak sudah bisa masuk ke tahapan pengerjaan. Karena proses SDLC pada aplikasi ini menggunakan metode kanban, maka jumlah tugas yang bisa dikerjakan dalam satu waktu harus dibatasi. Dalam kasus ini, hal yang membatasi jumlah pekerjaan dalam proses adalah jumlah anggota tim *back end*. Oleh karena hanya terdiri dari satu orang, maka jumlah yang bisa dikerjakan dalam satu waktu adalah satu tugas. Pada tahapan ini, tim pengembang akan menerima sebuah *user's stories* dan beragam detail dari desain yang telah ditentukan untuk diimplementasi menjadi sebuah kode. Fokus pengerjaan tahap ini akan dilakukan berbasis masing-masing tiket di *sub tasks* karena di situ sudah terkandung unit kerja terkecil yang independen. Prosesnya pun dimulai dari pengambilan sebuah tiket *story* dari *backlog* dan digeser / diubah statusnya menjadi *in progress* di papan kanban. Setelah itu, semua tiket *sub tasks* di *story* tersebut sudah bisa dikerjakan.

3.3.2.7 Code Review

Pada tahapan *code review*, kode hasil pekerjaan yang telah dibuat akan dilakukan peninjauan ulang untuk memastikan penulisan kode sudah benar sehingga meminimalisir terjadinya galat ketika kode dijalankan. Pada tahapan ini juga kita bisa memeriksa apakah fitur yang dibuat bisa dijalankan atau tidak. Hasil tangkapan layar, atau semacamnya bisa ditampilkan oleh tim pengembang untuk ditunjukkan sebagai *proof of work*. Tahapan ini akan dilakukan menggunakan alat bernama git & github untuk membuat *pull request*. *Pull request* inilah yang akan ditinjau dan divalidasi agar menjaga kualitas kode yang dibuat. Kemudian, akan dimanfaatkan juga sebuah fitur khusus dari github yang memungkinkan sebuah program dijalankan di setiap *pull request* yang bisa dimanfaatkan untuk melakukan pemeriksaan penulisan kode agar sesuai dengan standar baku yang ditetapkan, yakni dengan memanfaatkan sebuah alat yang bernama *linter*. Ketika terdapat pelanggaran aturan yang telah ditetapkan, *pull request* tersebut akan ditandai kesalahannya dan tidak bisa dilakukan *merge* hingga kesalahan tersebut diperbaiki. Dengan demikian, kualitas penulisan kode yang dihasilkan akan bisa selalu terjaga meskipun di kemudian hari dikembangkan oleh pengembang yang berbeda.

3.3.2.8 Software In Test

Pada tahapan *software in test*, semua tugas yang telah selesai dikerjakan dan telah melalui tahapan *code review* akan diuji menggunakan metode *black box testing* dan juga *unit testing*. Pada pengujian metode *black box testing*, akan dipersiapkan beberapa skenario uji yang akan menilai keluaran sistem berdasarkan masukan yang diterima. Skenario uji tersebut akan berisi kode tiket *story* yang diuji, masukan serta ekspektasi keluaran sistem. Hal ini juga bertujuan untuk memastikan tidak hanya kinerja sistem yang sesuai, namun juga memastikan sistem bisa memenuhi kebutuhan pengguna yang telah didefinisikan pada *user's stories*.

Untuk pengujian *unit testing*, akan diperiksa pada masing-masing komponen utama sistem yang tertera pada arsitektur sistem, yakni modul *use case*, REST dan juga *data access layer* untuk memiliki setidaknya mencakup 70%. Pengujian *unit testing* ini berguna untuk memastikan konsistensi sikap unit terkecil modul, yaitu sebuah fungsi, terhadap setiap masukan dan juga keluarannya. Selain itu, pengujian ini juga bisa memastikan setidaknya 70% dari keseluruhan modul telah teruji kinerjanya sesuai dengan ekspektasi.

3.3.2.9 *Deployment Staging*

Proses yang dilakukan pada *deployment staging* intinya adalah melakukan proses persiapan untuk *user acceptance test*. Dalam tahapan ini akan melakukan *deployment* terhadap fitur yang telah dibuat berdasarkan *user's stories* untuk diuji oleh pengguna itu sendiri sebelum dirilis ke publik. Tahapan yang dilakukan sebenarnya akan sama dengan *deployment production*, hanya berbeda tujuan serta jenis lingkungannya saja

3.3.2.10 *User Acceptance Test*

User Acceptance Test atau biasa disingkat UAT adalah sebuah proses yang bertujuan untuk memperoleh persetujuan dari pengguna dan konfirmasi bahwasanya fitur yang telah dikerjakan sesuai dengan ekspektasi pengguna serta bisa memenuhi peruntukannya. Dengan demikian, apabila terjadi kesalahan dimana fitur tidak sesuai dengan keinginan penggunanya, atau malah tidak mampu menyelesaikan masalah pengguna, hal tersebut bisa diketahui dengan cepat untuk diperbaiki di iterasi selanjutnya

3.3.2.11 *Deployment Production*

Pada tahapan ini yang akan dilakukan adalah melakukan *deployment* terhadap fitur yang telah selesai melalui semua tahapan uji untuk bisa digunakan oleh pengguna publik. Memang dalam penelitian ini dibatasi pada lingkungan lokal, namun konsepnya adalah sama. Semua fitur yang telah melalui tahapan ini juga dianggap telah selesai dikerjakan, dan diharapkan tidak lagi memiliki kesalahan dalam implementasinya yang dapat menimbulkan *bug*. Meskipun demikian, bukan berarti fitur tersebut tidak akan lagi disentuh perkembangan, karena bisa saja ketika dirilis fitur tersebut bermasalah. Apabila hal tersebut terjadi, maka fitur bisa dinonaktifkan dan diperbaiki terlebih dahulu untuk kemudian kembali dirilis versi terbarunya.

Proses *deployment* yang akan dilakukan pada penelitian ini adalah dengan menggunakan alat bernama docker. Apabila proses kompilasi aplikasi berbasis golang menghasilkan berkas *executable*, dengan menggunakan docker, hasil yang didapat adalah berupa *image*. *Image* inilah yang akan dijalankan dalam bentuk *container* yang sudah berisi semua kebutuhan aplikasi untuk dijalankan. Penelitian ini juga akan menggunakan *best practices* pada penggunaan docker, seperti menggunakan teknik *multi-stage builds*, *layer caching*, dan lain-lain. Penggunaan teknik tersebut menjadikan *image* aplikasi berukuran kecil, serta proses *build image* yang efisien.

3.3.3 **Penulisan Laporan**

Tahapan terakhir pada penelitian ini adalah melakukan penulisan laporan yang bertujuan untuk melakukan dokumentasi hasil penelitian agar bisa dimanfaatkan oleh pembaca ataupun menjadi tahapan awal penelitian selanjutnya.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dan pembahasan yang diperoleh dari penelitian ini, terdapat beberapa kesimpulan yang diambil sebagai berikut:

1. Sistem ATEC berbasis API berhasil dikembangkan menggunakan bahasa pemrograman Golang dengan menggunakan metode Kanban dengan total tiket *story* berjumlah 19 yang menghasilkan beragam fitur, mulai dari *sign up*, *log in*, penggantian kata sandi, pengisian kuesioner ATEC, pendaftaran anak, hingga statistik perkembangan skor ATEC yang dapat diakses melalui total 23 API *endpoints*.
2. Berdasarkan hasil pengujian sistem dengan menggunakan metode *black box testing* dan *unit testing*, sistem telah lolos pengujian dengan 64 skenario pengujian fungsional serta cakupan *unit testing* secara rata-rata mencapai 93,3% sehingga memastikan sistem bekerja sesuai peruntukannya untuk memenuhi semua kebutuhan pengguna dan implementasi metode ATEC.
3. Berdasarkan pengembangan perangkat lunak dengan metode Kanban, aplikasi berhasil dikembangkan sesuai dengan prinsip *agile* yang mampu mengakomodir keseluruhan kebutuhan pengguna sekaligus memberikan alur yang mengakomodir permasalahan selama pengembangan sistem serta durasi pengembangan sistem yang dihabiskan sekitar 202 jam.

5.2 Saran

Berdasarkan penelitian yang dilakukan, terdapat beberapa saran untuk pengembangan selanjutnya yakni sebagai berikut:

1. Penambahan proses otomatisasi pada pengujian yang menggunakan metode *black box testing* supaya mempercepat proses pengujian dan memastikan tidak ada kerusakan pada fitur lama ketika ada penambahan fitur baru.
2. Penambahan proses pembuatan *docker image* secara otomatis sebagai bagian dari proses otomatisasi menggunakan alat terintegrasi dengan *workflow* yang telah ada di github.
3. Penambahan proses *deployment* secara otomatis dengan menggunakan alat yang dapat diintegrasikan dengan *version control system* yang digunakan agar memudahkan proses pengembangan dan *deployment*.

DAFTAR PUSTAKA

- [1] C. Lord, E. H. Cook, B. L. Leventhal, and D. G. Amaral, "Autism spectrum disorders," *Neuron*, vol. 28, no. 2, pp. 355–363, Nov. 2000, doi: 10.1016/S0896-6273(00)00115-X.
- [2] C. Lord *et al.*, "Autism spectrum disorder," *Nature Rev. Dis. Primers*, vol. 6, no. 1, p. 5, Jan. 2020, doi: 10.1038/s41572-019-0138-4.
- [3] J. Zeidan *et al.*, "Global prevalence of autism: A systematic review update," *Autism Res.*, vol. 15, no. 5, pp. 778–790, May 2022, doi: 10.1002/aur.2696..
- [4] R. M. Botelho, A. L. M. Silva, and A. U. Borbely, "The autism spectrum disorder and its possible origins in pregnancy," *Int. J. Environ. Res. Public Health*, vol. 21, no. 3, p. 244, Feb. 2024, doi: 10.3390/ijerph21030244.
- [5] A. Modabbernia, E. Velthorst, and A. Reichenberg, "Environmental risk factors for autism: An evidence-based review of systematic reviews and meta-analyses," *Mol. Autism*, vol. 8, no. 1, p. 13, Mar. 2017, doi: 10.1186/s13229-017-0121-4.
- [6] S. L. Hyman *et al.*, "Identification, evaluation, and management of children with autism spectrum disorder," *Pediatrics*, vol. 145, no. 1, p. e20193447, Jan. 2020, doi: 10.1542/peds.2019-3447.
- [7] S. Sidhu, "New Year's Babies: Over 395,000 children will be born worldwide on New Year's Day," UNICEF, Jan. 1, 2019. [Accessed: Oct. 6, 2024]. [Online]. Available: <https://www.unicef.org/press-releases/new-years-babies-over-395000-children-will-be-born-worldwide-new-years-day-unicef>.
- [8] A. Fadila, O. F. Wardany, and H. Herlina, "Jenis pola asuh orang tua anak dengan gangguan spektrum autis di Lampung," *SPECIAL NEED*

- EDUCATION JOURNAL*, vol. 1, no. 2, pp. 16-22, Dec. 2021. [Online]. Available: <https://journal.uml.ac.id/SJ/article/view/670>. doi: 10.36269/sj.v1i2.670.
- [9] Pusat Data dan Teknologi Informasi, "Jumlah Sekolah Luar Biasa (SLB) berdasarkan Layanan Pendidikan," Kementerian Pendidikan Dasar dan Menengah Republik Indonesia 2024. [Accessed: Mar. 16, 2025]. [Online]. Available: <https://data.dikdasmen.go.id/dataset/p/peserta-didik/jumlah-sekolah-luar-biasa-slb-berdasarkan-layanan-pendidikan-2024>.
- [10] Z. K. Nantabah, Z. Auliyati, and A. D. Laksono, "Gambaran akses pelayanan kesehatan pada balita di Indonesia," *Buletin Penelitian Sistem Kesehatan*, vol. 22, no. 1, pp. 54–61, May 2019, doi: 10.22435/hsr.v22i1.439..
- [11] C. E. Conrad *et al.*, "Parent-mediated interventions for children and adolescents with autism spectrum disorders: A systematic review and meta-analysis," *Front. Psychiatry*, vol. 12, p. 773604, Nov. 2021, doi: 10.3389/fpsy.2021.773604.
- [12] A. Hamdhi, "Infrastruktur kesehatan Indonesia dari kacamata pengelola rumah sakit swasta," *KONTAN.CO.ID*, Sep. 16, 2024. [Diakses: 7 Okt. 2024]. [Online]. Tersedia: <https://jelajahekonomi.kontan.co.id/ekonomi-infrastruktur/news/infrastruktur-kesehatan-indonesia-dari-kacamata-pengelola-rumah-sakit-swasta>.
- [13] DataReportal, "Digital 2024: Global Overview Report," DataReportal, Jan. 2024. [Accessed: Oct. 7, 2024]. [Online]. Available: <https://datareportal.com/reports/digital-2024-global-overview-report>.
- [14] "List of countries by number of Internet users," Wikipedia: The Free Encyclopedia. Oct. 07, 2024. [Online]. Available: https://en.wikipedia.org/wiki/List_of_countries_by_number_of_Internet_users. [Accessed: Oct. 07, 2024].
- [15] B. Rimland and S. M. Edelson, "Autism Treatment Evaluation Checklist (ATEC)," Autism Research Institute, San Diego, CA, USA, 1999. [Online]. Available: <https://www.autism.org/autism-treatment-evaluation-checklist/> [Accessed: Oct. 07, 2024].

- [16] S. Mahapatra *et al.*, "Autism Treatment Evaluation Checklist (ATEC) norms: A 'growth chart' for ATEC score changes as a function of age," *Children*, vol. 5, no. 2, p. 25, Feb. 2018, doi: 10.3390/children5020025.
- [17] R. Netson *et al.*, "A comparison of parent reports, the Mental Synthesis Evaluation Checklist (MSEC) and the Autism Treatment Evaluation Checklist (ATEC), with the Childhood Autism Rating Scale (CARS)," *Pediatr. Rep.*, vol. 16, no. 1, pp. 174–189, Mar. 2024, doi: 10.3390/pediatric16010016.
- [18] D. A. Geier, J. K. Kern, and M. R. Geier, "A comparison of the Autism Treatment Evaluation Checklist (ATEC) and the Childhood Autism Rating Scale (CARS) for the quantitative evaluation of autism," *J. Ment. Health Res. Intellect. Disabil.*, vol. 6, no. 4, pp. 255–267, Oct. 2013, doi: 10.1080/19315864.2012.681340.
- [19] I. Magiati, J. Moss, R. Yates, T. Charman, and P. Howlin, "Is the Autism Treatment Evaluation Checklist a useful tool for monitoring progress in children with autism spectrum disorders?," *J. Intellect. Disabil. Res.*, vol. 55, no. 3, pp. 302–312, Mar. 2011, doi: 10.1111/j.1365-2788.2010.01359.x.
- [20] A. H. Memari, M. Shayestehfar, F.-S. Mirfazeli, T. Rashidi, P. Ghanouni, and S. Hafizi, "Cross-cultural adaptation, reliability, and validity of the autism treatment evaluation checklist in Persian," *Iran. J. Pediatr.*, vol. 23, no. 3, pp. 269–275, Jun. 2013. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3684470/>.
- [21] C. L. A. Elvitigala, N. B. Walpita, D. Dahanayake, and G. S. Wijetunge, "Validation of the Sinhala version of the Autism Treatment Evaluation Checklist (ATEC)," *Sri Lanka Journal of Psychiatry*, vol. 15, no. 1, pp. 10–15, Oct. 2024, doi: 10.4038/slpsyc.v15i1.8513.
- [22] K. Sunakarach and P. Kessomboon, "Validity and reliability of the Thai version of the Autism Treatment Evaluation Checklist: A two-phase diagnostic accuracy study," *F1000Research*, vol. 7, Art. no. 538, May 2018, doi: 10.12688/f1000research.14537.1. [Version 1; peer review: 1 approved, 1 approved with reservations].

- [23] Pusat Data dan Teknologi Informasi, "Jumlah bahasa daerah per provinsi," Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, 2025. [Accessed: Mar. 16, 2025]. [Online]. Available: <https://referensi.data.kemdikbud.go.id/kebahasaan/bahasadaerah>
- [24] Education First, "EF English Proficiency Index 2024," Education First, 2024. [Accessed: Mar. 16, 2025]. [Online]. Available: <https://www.ef.com/wwen/epi/>.
- [25] Red Hat, "What is a REST API," Red Hat, 2025. [Accessed: Mar. 16, 2025]. [Online]. Available: <https://www.redhat.com/en/topics/api/what-is-a-rest-api>.
- [26] E. M. Ladapase, "Keterlambatan bicara (speech delay) pada anak usia 4 tahun (studi kasus di lembaga layanan anak berkebutuhan khusus karya ilahi)," *Empowerment*, vol. 1, no. 2, pp. 149–160, 2022. [Online]. Available: <https://www.mendeley.com/catalogue/85765f3d-0940-3f6d-a767-81071b57debf/>
- [27] N. Hasanah and S. Sugito, "Analisis Pola Asuh Orang Tua terhadap Keterlambatan Bicara pada Anak Usia Dini," *Jurnal Obsesi : Jurnal Pendidikan Anak Usia Dini*, vol. 4, no. 2, p. 913, Mar. 2020, doi: 10.31004/obsesi.v4i2.456.
- [28] F. Aulia, H. Ain, and N. Pujiastuti, "Factors affecting speech delay in toddlers," *J. Nurs. Sci. Update*, vol. 11, no. 1, pp. 9–17, Jun. 2023, doi: 10.21776/ub.jik.2023.011.01.2.
- [29] F. R. Makarim, "3 cara untuk mendeteksi speech delay pada anak," halodoc.com, 2023. [Accessed: Mar. 16, 2025]. [Online]. Available: <https://www.halodoc.com/artikel/3-cara-untuk-mendeteksi-speech-delay-pada-anak>.
- [30] P. Naydenov, "What is Kanban? A simple guide to improve efficiency," *businessmap.io*, 2024. [Accessed: Aug. 18, 2024]. [Online]. Available: <https://businessmap.io/kanban-resources/getting-started/what-is-kanban>.
- [31] D. J. Anderson, *Kanban: Successful Evolutionary Change for Your Technology Business*, 1st ed. Sequim, WA: Blue Hole Press, 2010.

- [32] A. Stellman and J. Greene, *Learning Agile*, 1st ed. Sebastopol, CA: O'Reilly Media, 2015.
- [33] S. V. Salunke and A. Ouda, "A performance benchmark for the PostgreSQL and MySQL databases," *Future Internet*, vol. 16, no. 10, p. 382, Oct. 2024, doi: 10.3390/fi16100382.
- [34] M. Zhou, "A review of JavaScript object notation in data analysis," in *Proc. 2022 8th Int. Conf. Humanities Soc. Sci. Res. (ICHSSR)*, 2022, pp. 1771–1774, doi: 10.2991/assehr.k.220504.321.
- [35] V. Khorikov, *Unit Testing: Principles, Practices, and Patterns*, 1st ed. Shelter Island, NY: Manning Publications, 2020.
- [36] A. Maspupah, "Literature review: Advantages and disadvantages of black box and white box testing methods," *J. Techno Nusa Mandiri*, vol. 21, no. 2, pp. 151–162, Sep. 2024, doi: 10.33480/techno.v21i2.5776.
- [37] M. N. Ichsanudin, M. Yusuf, and S. Suraya, "Pengujian fungsional perangkat lunak sistem informasi perpustakaan dengan metode black box testing bagi pemula," *STORAGE: J. Ilm. Tek. Ilmu Komput.*, vol. 1, no. 2, pp. 113–120, Mei 2022, doi: 10.55123/storage.v1i2.270.
- [38] R. Rashkovits and I. Lavy, "Mapping common errors in entity relationship diagram design of novice designers," *Int. J. Database Manag. Syst.*, vol. 13, no. 1, pp. 1–19, Feb. 2021, doi: 10.5121/ijdms.2021.13101.
- [39] Docker, "What is Docker?," *Docker Documentation*, 2023. [Accessed: Nov. 06, 2024]. [Online]. Available: <https://docs.docker.com/get-started/docker-overview/>.
- [40] G. Pujiyanto, F. Riana, and S. H. A. Safaruddin, "Pengembangan Web Service pada Sistem Informasi Jahe Menggunakan Metode REST API," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 5, Sep. 29, 2024, doi: 10.36040/jati.v8i5.10669.
- [41] N. Dimitrijević, N. Zdravković, M. Bogdanović, and A. Mesterović, "Advanced Security Mechanisms in the Spring Framework: JWT, OAuth, LDAP and Keycloak," in *Proc. of the 8th International Conference on Digital Technologies and Applications (ICDTA 2024)*, CEUR Workshop

- Proceedings, vol. 3676, pp. 80-91, 2024. [Online]. Available: https://ceur-ws.org/Vol-3676/short_09.pdf.
- [42] A. Del Sole, *Visual Studio Code Distilled: Evolved Code Editing for Windows, macOS, and Linux*. Berkeley, CA: Apress, 2019. doi: 10.1007/978-1-4842-4224-7.
 - [43] S. Uzayr, *Mastering Visual Studio Code: A Beginner's Guide*, 1st ed. Boca Raton, FL: CRC Press, 2023. doi: 10.1201/9781003311973.
 - [44] S. bin Uzayr, *Mastering Ubuntu*. Boca Raton, FL: CRC Press, 2022. doi: 10.1201/9781003311997.
 - [45] N. Haines, *Beginning Ubuntu for Windows and Mac Users*. Berkeley, CA: Apress, 2023. doi: 10.1007/978-1-4842-8972-3.
 - [46] R. C. Martin, *Clean Architecture: A Craftsman's Guide to Software Structure and Design*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2017. ISBN 978-0-13-449416-6.
 - [47] B. Öggl and M. Kofler, *Git Project Management for Developers and DevOps Teams: Everything about Conventions, Standards, and Best Practices for Productive Project Management with Git*, 1st ed. Boston, MA: Rheinwerk Publishing Inc., 2023. ISBN: 978-1-4932-2290-2.
 - [48] C. R. D'Adamo, J. L. Nelson, S. N. Miller, M. Rickert Hong, E. Lambert, and H. Tallman Ruhm, "Reversal of Autism Symptoms among Dizygotic Twins through a Personalized Lifestyle and Environmental Modification Approach: A Case Report and Review of the Literature," *J. Pers. Med.*, vol. 14, no. 6, p. 641, Jun. 2024, doi: 10.3390/jpm14060641.
 - [49] A. S. Adamu and S. E.-Y. Abdullahi, "A review of autism diagnosis/screening expert system and mobile application," in *2020 IEEE 2nd International Conference on Cyberspace (CYBER NIGERIA)*, 23-25 Feb. 2021, pp. 80-84. doi: 10.1109/CYBERNIGERIA51635.2021.9428833..
 - [50] A. Rahman, E. Indrajit, A. Unggul, and E. Dazki, "Agile Project Management Impacts Software Development Team Productivity," *Sinkron*, vol. 8, no. 3, pp. 1847-1858, Jul. 2023, doi: 10.33395/sinkron.v8i3.13853.
 - [51] P. Suparman and M. Huda, "Penerapan Kanban Agile Development dalam Pengembangan Sistem Manajemen Skripsi dan Tugas Akhir STMIK

- Cikarang Menggunakan Framework Codeigniter," *J. Inform. SIMANTIK*, vol. 6, no. 1, pp. 7-11, Mar. 2021, [Accessed: Nov. 11, 2024]. [Online]. Available: <https://simantik-panca-sakti.ac.id/index.php/simantik/article/view/70>.
- [52] S. Priadi and M. Idris, "Implementasi REST dalam Membangun Web Service Menggunakan Golang (Studi Kasus: Modul Berita Aplikasi Satudikti)," *J. INSTEK (Informatika Sains Tekn.)*, vol. 8, no. 2, pp. 349-358, 2023, doi: 10.24252/instek.v8i2.42264.
- [53] R. Cahya, W. Sulistiadi, N. F. Tu, and P. H. Trenggono, "Dampak Hambatan Geografis dan Strategi Akses Pelayanan Kesehatan: Literature Review," *Media Publikasi Promosi Kesehatan Indonesia (MPPKI)*, vol. 6, no. 5, pp. 868–877, Mei 2023, doi: 10.56338/mppki.v6i5.2935. [Online]. Available: <https://jurnal.unismuhpalu.ac.id/index.php/MPPKI/article/view/2935>.
- [54] M. F. Thohari, Darmastuti, and S. Damerianta, "Perancangan Application Programming Interface (API) untuk Mengakses Layanan Vaksinasi COVID-19 Menggunakan Golang Echo Framework," *Jurnal Ilmiah Informatika Komputer*, vol. 28, no. 3, pp. 218-229, 2023, doi: 10.35760/ik.2023.v28i3.9423.