**PREDIKSI NILAI TUKAR BITCOIN DAN US DOLLAR DENGAN METODE LONG SHORT-TERM MEMORY (LSTM)**

**SKRIPSI**

diajukan untuk menempuh ujian sarjana

pada Fakultas Matematika dan Ilmu Pengetahuan Alam  
Universitas Padjadjaran

HUSEIN IRFAN

NPM 140810170035



UNIVERSITAS PADJADJARAN  
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM  
PROGRAM STUDI TEKNIK INFORMATIKA  
SUMEDANG  
2023

**SKRIPSI**

**PREDIKSI NILAI TUKAR BITCOIN DAN US DOLLAR DENGAN METODE LONG SHORT-TERM MEMORY (LSTM)**

**INTRODUCTION TO EMOTIONAL CONTEXT IN CONVERSATION TEXT USING NETWORK DEEP LEARNING METHODS LONG SHORT-TERM MEMORY (LSTM) AND ATTENTION**

Telah dipersiapkan dan disusun oleh

HUSEIN IRFAN

NPM 140810170035

Telah dipertahankan di depan Tim Penguji

pada tanggal …………….

Susunan Tim Penguji

1. Dr. Juli Rejito, M.Kom. Ketua Tim Penguji

NIP. 19680717 199303 1 003 ……………………….

1. ……………………………… Pembimbing

NIP. ………………………... ……………………….

1. ……………………………… Co-Pembimbing

NIP. ………………………... ……………………….

1. ……………………………… Penguji

NIP. ………………………... ……………………….

1. ……………………………… Penguji

NIP. ………………………... ……………………….

1. ……………………………… Penguji

NIP. ………………………... ……………………….

**KATA PENGANTAR**

*Bismillahirahmanirrahim*. Dengan menyebut nama Allah yang Maha Pengasih dan Maha Penyayang. Segala puji dan syukur penulis panjatkan ke hadirat Allah SWT yang telah memberikan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan penyusunan skripsi yang berjudul “**………………………………**” sebagai salah satu syarat menempuh sarjana pada Program Studi S-1 Teknik Informatika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Padjadjaran.

Dalam proses penyusunan dan penulisan skripsi ini tidak terlepas dari bantuan, bimbingan, serta dukungan dari berbagai pihak. Oleh karena itu dalam kesempatan ini penulis mengucapkan terima kasih kepada Bapak ……………., sebagai pembimbing utama, Bapak Aditya Pradana, S.T., M.Eng, sebagai pembimbing pendamping yang telah meluangkan waktu dan pikirannya sehingga penulis dapat menyelesaikan skripsi ini. Ucapan terima kasih juga diberikan kepada keluarga penulis yang selalu memberikan motivasi dan doa yang menjadi pendorong dalam penyelesaian skripsi ini. Penulis juga mengucapkan terima kasih sebanyak-banyaknya kepada:

………………., selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Padjadjaran.

………………., selaku Kepala Departemen Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Padjadjaran.

………………., selaku Ketua Program Studi S-1 Teknik Informatika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Padjadjaran

Dosen-dosen Teknik Informatika Unpad yang telah mengajar dan memberikan ilmu kepada penulis selama masa perkuliahan yang membawa penulis pada posisi sekarang ini.

…………………

Akhir kata, penulis berharap semoga skripsi ini dapat bermanfaat bagi pembaca.

Jatinangor, 2 Pebruari 2021

Penulis

ABSTRAK

Di era digital ini, banyak universitas yang menyediakan web repositori dimana mahasiswa dapat mengakses berkas digital dari skripsi atau karya ilmiah milik *civitas academica*. Digitalisasi ini tentunya memudahkan mahasiswa dalam memperoleh referensi dan sumber bacaan tanpa perlu mengunjungi perpustakaan. Pada saat ini, program studi Teknik Informatika Unpad belum memiliki repositori maupun sarana serupa. Hal inilah yang menjadi motivasi dasar pada penelitian ini untuk membangun repositori skripsi Teknik Informatika Unpad. Optimalnya, sebuah web harus responsif terhadap *smartphone* dan memiliki pengalaman pengguna yang baik.

Metoda penelitian yang digunakan dalam penelitian ini memanfaatkan *Extreme Programming* dengan model pengembangan perangkat lunak *Agile* sehingga bersifat gesit dan iteratif. Metoda *Extreme Programming* memiliki 6 fase yaitu fase eksplorasi, fase perencanaan, fase iterasi untuk dirilis, fase produksi, fase pemeliharaan, dan fase akhir. Dengan siklus singkat dan berfokus pada permintaan pengguna, aplikasi terus mendapat *feedback* sehingga meningkatkan kualitas aplikasi.

Dari hasil penelitian didapat nilai *usability testing* sebesar 88.53% dan *heuristic evaluation* sebesar 72%. Dari hasil tersebut dapat disimpulkan bahwa sistem repositori yang dibangun dengan keunggulan PWA dan metode *Extreme Programming* berjalan dengan baik dan memiliki nilai kepuasan yang baik pula.

**Kata Kunci** : ExpressJs*, Extreme Programming*, PWA, ReactJs, Repositori

*ABSTRACT*

*In this digital era, many universities provide a web repository where students can access digital files from theses or scientific works belonging to the academic community. This digitization certainly makes it easier for students to obtain references and reading sources without the need to visit the library. At this time, Unpad Informatics Engineering study program does not yet have a repository or similar facilities. This is the basic motivation for this research to build a thesis repository for Informatics Engineering Unpad. Optimally, a web should be responsive to smartphones and have a good user experience.*

*The research method used in this study utilizes Extreme Programming with the Agile software development model so that it is agile and iterative. The Extreme Programming method has 6 phases, namely the exploration phase, the planning phase, the iteration phase for release, the production phase, the maintenance phase, and the final phase. With a short cycle and focused on user requests, the application continues to receive feedback, thereby improving application quality.*

*The results showed that the usability testing value was 88.53% and the heuristic evaluation was 72%. From these results it can be concluded that the repository system built with the advantages of PWA and the Extreme Programming method runs well and has a good satisfaction score as well.*

***Keywords :*** *ExpressJs, Extreme Programming, PWA, ReactJs, Repository*

DAFTAR ISI

[KATA PENGANTAR iii](#_Toc52929842)

[ABSTRAK v](#_Toc52929843)

[ABSTRACT vi](#_Toc52929844)

[DAFTAR ISI vii](#_Toc52929845)

[DAFTAR TABEL x](#_Toc52929846)

[DAFTAR GAMBAR xi](#_Toc52929847)

[DAFTAR LAMPIRAN xiii](#_Toc52929848)

[BAB I PENDAHULUAN 1](#_Toc52929849)

[1.1 Latar Belakang 1](#_Toc52929850)

[1.2 Identifikasi Masalah 5](#_Toc52929851)

[1.3 Batasan Masalah 5](#_Toc52929852)

[1.4 Maksud dan Tujuan Penelitian 6](#_Toc52929853)

[1.5 Manfaat Penelitian 6](#_Toc52929854)

[1.6 Metodologi Penelitian 7](#_Toc52929855)

[1.7 Sistematika Penulisan 8](#_Toc52929856)

[BAB II TINJAUAN PUSTAKA 10](#_Toc52929857)

[2.1 Repositori Institusi 10](#_Toc52929858)

[2.2 *Extreme Programming* 10](#_Toc52929859)

[2.3 *Progressive Web Apps* 14](#_Toc52929860)

[2.4 ReactJS 15](#_Toc52929861)

[2.5 Express 16](#_Toc52929862)

[2.6 Javascript 17](#_Toc52929863)

[2.7 Bootstrap 18](#_Toc52929864)

[2.8 MySQL 19](#_Toc52929865)

[2.9 *Unified Modeling Language* (UML) 19](#_Toc52929866)

[2.9.1 *Use Case Diagram* 20](#_Toc52929867)

[2.9.2 *Activity Diagram* 21](#_Toc52929868)

[2.9.3 *Deployment Diagram* 22](#_Toc52929869)

[2.10 *Entity Relationship Diagram (ERD)* 23](#_Toc52929870)

[2.11 *Black Box Testing* 24](#_Toc52929871)

[2.12 *Usability Testing* 25](#_Toc52929872)

[2.13 *Heuristic Evaluation* 26](#_Toc52929873)

[2.14 Skala Likert 29](#_Toc52929874)

[2.15 Heroku 30](#_Toc52929875)

[BAB III ANALISIS DAN DESAIN 32](#_Toc52929876)

[3.1 Fase Eksplorasi 32](#_Toc52929877)

[3.1.1 Kebutuhan Pengguna 33](#_Toc52929878)

[3.1.2 Kebutuhan Data 33](#_Toc52929879)

[3.1.3 Kebutuhan Perangkat Lunak 34](#_Toc52929880)

[3.1.4 Kebutuhan Perangkat Keras 34](#_Toc52929881)

[3.1.5 Kebutuhan Sistem 35](#_Toc52929882)

[3.1.6 *User Story* 37](#_Toc52929883)

[3.2 Fase Perencanaan 39](#_Toc52929884)

[3.3 Fase Iterasi 41](#_Toc52929885)

[3.3.1 Analisis Sistem 41](#_Toc52929886)

[3.3.2 Analisis Arsitektur Menggunakan *Deployment Diagram* 44](#_Toc52929887)

[3.3.3 Desain Basis Data 44](#_Toc52929888)

[3.3.4 Desain Antarmuka 49](#_Toc52929889)

[3.3.5 *Testing* 59](#_Toc52929890)

[3.4 Fase Produksi 59](#_Toc52929891)

[3.5 Fase Pemeliharaan dan Fase Akhir 63](#_Toc52929892)

[BAB IV HASIL DAN PEMBAHASAN 65](#_Toc52929893)

[4.1 Impelementasi Program 65](#_Toc52929894)

[4.1.1 Halaman Utama 65](#_Toc52929895)

[4.1.2 Halaman Register 68](#_Toc52929896)

[4.1.3 Menu *Login* 74](#_Toc52929897)

[4.1.4 Halaman Unggah Skripsi 78](#_Toc52929898)

[4.1.5 Halaman Menu Admin 82](#_Toc52929899)

[4.1.6 Halaman Verifikasi Akun 84](#_Toc52929900)

[4.1.7 Halaman Tinjau Skripsi 88](#_Toc52929901)

[4.1.8 Halaman Detail Skripsi 92](#_Toc52929902)

[4.1.9 Fitur Pencarian dan Penyaringan 94](#_Toc52929903)

[4.1.10 Halaman Profil danFitur *Edit Password* 97](#_Toc52929904)

[4.1.11 Halaman Status Skripsi 100](#_Toc52929905)

[4.1.12 Implementasi PWA 104](#_Toc52929906)

[4.2 Fase Produksi 109](#_Toc52929907)

[4.2.1 Rilisan Kecil 109](#_Toc52929908)

[4.2.2 Pengujian 109](#_Toc52929909)

[4.3 Fase Pemeliharaan dan Fase Akhir 114](#_Toc52929910)

[4.3.1 *Feedback* 114](#_Toc52929911)

[4.3.2 Implementasi *Feedback* 115](#_Toc52929912)

[BAB V KESIMPULAN DAN SARAN 125](#_Toc52929913)

[5.1 Kesimpulan 125](#_Toc52929914)

[5.2 Saran 126](#_Toc52929915)

[DAFTAR PUSTAKA 127](#_Toc52929916)

[LAMPIRAN 130](#_Toc52929917)

DAFTAR TABEL

[Tabel 2.1 Simbol-Simbol pada *Use Case Diagram* 20](#_Toc52929958)

[Tabel 2.2 Simbol-Simbol pada *Activity Diagram* 21](#_Toc52929959)

[Tabel 2.3 Simbol-Simbol pada *Deployment Diagram* 22](#_Toc52929960)

[Tabel 2.4 Simbol-simbol pada ERD 24](#_Toc52929961)

[Tabel 2.5 Titik Respons 30](#_Toc52929962)

[Tabel 3.1 Kebutuhan Sistem Pengguna: Mahasiswa 36](#_Toc52929963)

[Tabel 3.2 Kebutuhan Sistem Pengguna: Admin 36](#_Toc52929964)

[Tabel 3.3 *User Story* 37](#_Toc52929965)

[Tabel 3.4 Prioritas Fitur 40](#_Toc52929966)

[Tabel 3.5 Atribut Tabel skripsi 46](#_Toc52929967)

[Tabel 3.6 Atribut Tabel *users* 47](#_Toc52929968)

[Tabel 3.7 Atribut Tabel *temp\_users* 48](#_Toc52929969)

[Tabel 3.8 Atribut Tabel *forums* 48](#_Toc52929970)

[Tabel 3.9 Skenario Uji 60](#_Toc52929971)

[Tabel 3.10 *Form* *Usability Testing* 61](#_Toc52929972)

[Tabel 3.11 *Form Heuristic Evaluation* 62](#_Toc52929973)

[Tabel 3.12 Atribut Kriteria Nilai 63](#_Toc52929974)

[Tabel 4.1 *Black Box Testing* pada Fitur Registrasi 73](#_Toc52929975)

[Tabel 4.2 *Black Box* *Testing* pada Fitur *Login* 77](#_Toc52929976)

[Tabel 4.3 *Black Box Testing* pada Fitur UnggahSkripsi 82](#_Toc52929977)

[Tabel 4.4 *Black Box Testing* pada Fitur Verifikasi Akun 88](#_Toc52929978)

[Tabel 4.5 *Black Box Testing* pada Fitur Tinjau Skripsi 92](#_Toc52929979)

[Tabel 4.6 *Black Box Testing* pada Fitur Pencarian dan Penyaringan 96](#_Toc52929980)

[Tabel 4.7 *Black Box Testing* pada Fitur *Edit Password* 100](#_Toc52929981)

[Tabel 4.8 *Black Box Testing* Fitur PWA 109](#_Toc52929982)

[Tabel 4.9 Hasil *Usability Testing* 110](#_Toc52929983)

[Tabel 4.10 Hasil *Heuristic Evaluation* 111](#_Toc52929984)

[Tabel 4.11 Nilai *Heuristic Evaluation* 113](#_Toc52929985)

DAFTAR GAMBAR

[Gambar 2.1 Siklus HidupXP(Krishna *et al.*, 2011) 12](#_Toc52930057)

[Gambar 2.2Prinsip UtamaPWA (Karpagam *et al.*, 2017) 15](#_Toc52930058)

[Gambar 2.3 Alur *Request* pada Express (Hahn, 2016:7) 16](#_Toc52930059)

[Gambar 2.4 Struktur Aplikasi pada Heroku (Heroku, 2020) 31](#_Toc52930060)

[Gambar 3.1 Kompatibilitas *Browser* Chrome dan Mozila pada Perangkat (Santoni, 2018) 35](#_Toc52930061)

[Gambar 3.2 *Use Case Diagram* 41](#_Toc52930062)

[Gambar 3.3 *Activity Diagram* 43](#_Toc52930063)

[Gambar 3.4 *Deployment Diagram* 44](#_Toc52930064)

[Gambar 3.5 *Entity Relationship Diagram* 45](#_Toc52930065)

[Gambar 3.6 Model Data Fisik 46](#_Toc52930066)

[Gambar 3.7 Desain Halaman Utama 49](#_Toc52930067)

[Gambar 3.8 Desain Halaman Registrasi Bagian 1 50](#_Toc52930068)

[Gambar 3.9 Desain Halaman Registrasi Bagian 2 50](#_Toc52930069)

[Gambar 3.10 Desain *Login Section* 51](#_Toc52930070)

[Gambar 3.11 Desain Halaman Detail Skripsi 52](#_Toc52930071)

[Gambar 3.12 Desain Halaman Unggah Skripsi 53](#_Toc52930072)

[Gambar 3.13 Desain Halaman Profil Mahasiswa 54](#_Toc52930073)

[Gambar 3.14 Desain Halaman *Edit Password* 54](#_Toc52930074)

[Gambar 3.15 Desain Halaman Status Skripsi 55](#_Toc52930075)

[Gambar 3.16 Desain Halaman Menu Admin 55](#_Toc52930076)

[Gambar 3.17 Desain Halaman Verifikasi Akun Mahasiswa 56](#_Toc52930077)

[Gambar 3.18 Desain Halaman Tinjau Skripsi 57](#_Toc52930078)

[Gambar 3.19 Desain Halaman Kontak Admin 58](#_Toc52930079)

[Gambar 3.20 Desain Halaman Lupa *Password* 58](#_Toc52930080)

[Gambar 4.1 Halaman Utama pada *Desktop* (a) dan *Mobile* (b) 66](#_Toc52930081)

[Gambar 4.2 Halaman Registrasi bagian 1 68](#_Toc52930082)

[Gambar 4.3 Halaman Registrasi Bagian 2 69](#_Toc52930083)

[Gambar 4.4 Menu *Login* 74](#_Toc52930084)

[Gambar 4.5 Halaman Unggah Skripsi 78](#_Toc52930085)

[Gambar 4.6 Halaman Menu Admin 83](#_Toc52930086)

[Gambar 4.7 Halaman Verifikasi Akun 84](#_Toc52930087)

[Gambar 4.8 Halaman Tinjau Skripsi 89](#_Toc52930088)

[Gambar 4.9 Halaman Detail Skripsi 92](#_Toc52930089)

[Gambar 4.10 Halaman Profil 97](#_Toc52930090)

[Gambar 4.11 Bagian *Edit Password* 97](#_Toc52930091)

[Gambar 4.12 Halaman Status Skripsi 101](#_Toc52930092)

[Gambar 4.13 Implementasi Fitur *Offline* 106](#_Toc52930093)

[Gambar 4.14 Implementasi Fitur *add to homescreen* 108](#_Toc52930094)

[Gambar 4.15 Web Dalam Bentuk *Native App* 108](#_Toc52930095)

[Gambar 4.16 Tampilan Profil Setelah Diperbaharui 115](#_Toc52930096)

[Gambar 4.17 Halaman Edit Unggahan 116](#_Toc52930097)

[Gambar 4.18 Navigator Menu 118](#_Toc52930098)

[Gambar 4.19 Menu *Login* pada *Smartphone* (a) Lama (b) Baru 119](#_Toc52930099)

[Gambar 4.20 Kontak Admin 119](#_Toc52930100)

[Gambar 4.21 Halaman Lupa *Password* 121](#_Toc52930101)

# **DAFTAR LAMPIRAN**

[Lampiran 1 Ringkasan Hasil *Usability Testing* 130](#_Toc35650951)

[Lampiran 2 Kode Web Aplikasi 136](#_Toc35650952)

# **BAB I**

**PENDAHULUAN**

1. **Latar Belakang**

Bitcoin merupakan *cryptocurrency* pertama yang dibuat oleh Satoshi Nakamoto pada tahun 2009. Bitcoin mulai mendapatkan popularitas pertama pada bulan Februari tahun 2011 saat nilai tukar bitcoin mencapai $1 dan naik ke $31 sekitar bulan Juni 2011 namun kemudian turun ke satu digit beberapa bulan kemudian. Tidak lama kemudian bitcoin kembali naik harganya mencapai $200 pada bulan April tahun 2013 hingga $1000 pada November tahun 2013. Seperti sebelumnya, harganya kembali turun.

Pada tahun 2017, bitcoin mendapat popularitas sekali lagi setelah mencapai nilai $1345 pada bulan Maret yang merupakan harga tertinggi pada masanya. Dan pada 11 Desember tahun 2017, bitcoin menembus angka $20.000 dan seperti pada kasus sebelunya, harganya kembali turun. Pada hari penulis membuat tulisan ini, Bulan Agustus 2023, nilai tukar bitcoin ke USD adalah $29.402,25.

Dari tulisan sebelumnya, bisa disimpulkan bahwa nilai tukar bitcoin tidak stabil dan merupakan infestasi yang memberikan high risk/high reward. Untuk itu, kitab bisa memanfaatkan *machine learning* untuk mencari pola dari pergerakan harga bitcoin dan memprediksi harga bitcoin untuk menyusun strategi *trading* yang menguntungkan.

*Machine learning* adalah salah satu cabang dari sistem kecerdasan buatan atau *Artificial Intelligence* (AI) yang mencapkup perancangan sistem algoritma yang memungkinkan komputer memiliki kemampuan untuk belajar dari data, tanpa secara eksplisit harus mengikuti instruksi yang diprogram oleh pembuat sistem (Budiharto, 2016).

Salah satu metode *machine learning* yang bisa dipakai untuk membuat model prediksi adalah *Long Short-Term Memory* (LSTM). *Long Short-Term Memory* (LSTM) adalah salah satu cabang dari *Recurrent Neural Network* (RNN) yang diperkenalkan oleh Hochreiter & Schmidhuber (1997), dan disempurnakan dan dipopulerkan oleh banyak orang. LSTM sangat cocok untuk diterapkan pada *dataset* berbasis time series dan masalah dengan jangka panjang.

1. **Identifikasi Masalah**

Berdasarkan latar belakang yang telah dijelaskan, berikut adalah rumusan masalah dari penelitian ini:

1. Bagaimana cara membuat model prediksi dengan *Long Short-Term Memory* (LSTM)?
2. Bagaimana cara memprediksi nilai tukar rupiah dengan dollar Amerika Serikat rupiah dengan dollar Amerika Serikat?
3. Bagaimana akurasi model prediksi laju inflasi dengan LSTM?

1. **Batasan Masalah**

Berdasarkan identifikasi masalah yang telah dijelaskan, berikut adalah batasan masalah dari penelitian ini:

1. Data yang digunakan diambil dari halaman website Yahoo Finance.
2. Dataset yang akan digunakan untuk penelitian merupakan data histori nilai tukar bitcoin (BTC) dan dollar amerika serikat (USD).
3. Dataset yang akan digunakan diambil dari tanggal 17 September 2017 sampai 10 Agustus 2023.
4. Perangkat lunak akan dibuat dengan bahasa pemrograman python dan dibantu dengan beberapa library untuk mempermudah pembuatan perangkat lunak.
5. Algoritma yang digunakan untuk pembuatan model prediksi adalah *Long Short-Term Memory* (LSTM).
6. Untuk mengukur besar galat atau error dari hasil prediksi, akan digunakan *Mean Square Error* (MSE) dan *Root Mean Square Error* (RMSE) sebagai evaluasi model.
7. Untuk visualisasi, akan menggunakan Flask untuk membuat GUI (*Graphical User Interface)* berbasis web.
8. **Maksud dan Tujuan Penelitian**

Maksud dari penelitian ini adalah mengembangkan algoritma *Long Short-Term Memory* (LSTM) untuk membuat model prediksi untuk meramal nilai tukar atau kurs mata uang Amerika Serikat, dollar (USD), dengan bitcoin (BTC). Adapun tujuan yang ingin dicapai penulis dari penelitian ini sebagai berikut

1. Dapat menghasilkan model prediksi yang akurat atau nilai galat yang relatif kecil.
2. Dapat mengimplementasikan *Long Short-Term Memory* (LSTM) untuk memrediksi nilai tukar bitcoin dengan dollar Amerika Serikat
3. **Manfaat Penelitian**

Berikut adalah manfaat yang diharapkan dari penulisan penelitian ini:

1. Laporan ini bisa digunakan sebagai referensi untuk pengimplementasian algoritma *Long Short-Term Memory* (LSTM) untuk dataset time series.
2. Dengan memprediksi nilai tukar bitcoin, diharapkan *trader* bisa membuat perencanaan *trading* yang lebih baik.
3. Menjadi dasar penelitian untuk pengembangan lebih lanjut terkait dengan penelitian ini.
4. **Metodologi Penelitian**

Berikut adalah tahapan yang dilakukan pada penelitian ini:

1. Studi literatur dengan mencari referensi dan informasi yang dibutuhkan dalam penulisan skripsi.
2. Mengumpulkan data nilai bitcoin (BTC) dengan dollar amerika serikat (USD) dari situs Yahoo Finance yang bisa diakses di tautan berikut: <https://finance.yahoo.com/quote/BTC-USD>.
3. Melakukan preprocessing data agar data bisa diguakan untuk membuat model prediksi dengan metode *Long Short-Term Memory* (LSTM).
4. Melakukan analisis dan merancang sistem.
5. Pelatihan serta pengujian model.
6. Analisis dan evaluasi model.
7. **Sistematika Penulisan**

Untuk memberi gambaran yang jelas tentang penelitian ini, maka disusunlah sistematika penulisan yang berisi materi yang akan dibahas pada setiap bab. Sistematika dalam penulisan tugas akhir ini adalah sebagai berikut:

**BAB I PENDAHULUAN**

Pada bab ini dijelaskan tentang latar belakang dari topik penulisan skripsi, pokok permasalahan berupa identifikasi dan batasan masalah, tujuan dan manfaat yang diharapkan dari penulisan skripsi, metodologi yang digunakan serta sistematika penulisan.

**BAB II TINJAUAN PUSTAKA**

Pada bab ini dijelaskan seluruh landasan teori yang berhubungan dengan penelitian, yaitu penjelasan singkat tentang bitcoin, *machine learning*, *Recurrent Neural Network* (RNN), *Long Short-Term Memory* (LSTM), dan metode evaluasi model *Mean Square Error* (MSE) dan *Root Mean Square Error* (RMSE).

**BAB III ANALISIS DAN PERANCANGAN**

Pada bab ini dijelaskan tentang metode pengembangan model prediksi yang digunakan meliputi Desain perancangan sistem, analisis kebutuhan, data preprocessing, serta jadwal penelitian.

**BAB IV HASIL DAN PEMBAHASAN**

Pada bab ini dijelaskan tentang implementasi aplikasi yang telah dibangun dan tahapan evaluasi aplikasi.

**BAB V KESIMPULAN DAN SARAN**

Bab ini merupakan penutup yang berisi kesimpulan dan saran dari penelitian yang sudah dilakukan.

# **BAB II**

**TINJAUAN PUSTAKA**

* 1. **Bitcoin**

Bitcoin adalah salah satu *cryptocurrency* atau mata uang digital yang dibuat oleh Satoshi Nakamoto pada tahun 2009. Dari website resmi bitcoin, Bitcoin menggunakan teknologi peer-to-peer untuk beroperasi tanpa otoritas dari pihak ketiga seperti bank. Pengelolaan transaksi dan penerbitan bitcoin dilakukan secara kolektif oleh jaringan. bitcoin desainnya bersifat *public* atau *open source*, tidak ada yang memiliki atau mengontrol Bitcoin.

Seperti mata uang negara, bitcoin juga memiliki nilai tukarnya terhadap mata uang lain seperti dollar amerika serikat (USD). Nilai tukar adalah besarnya atau harga suatu mata uang terhadap mata uang lainnya. Nilai bisa berubah setiap waktunya. Ada beberapa istilah yang biasa dilihat jika anda ingin mencari nilai tukar bitcoin. Istilah ini biasa digunakan oleh trader karena bitcoin juga bisa digunakan untuk investasi. Dibawah ini adalh istilah-istilah yang akan anda lihat saat melihat nilai tukar bitcoin:

1. *Open*, data nilai harga pertama pada saat transaksi dibuka pada hari tersebut.
2. *Close*, data nilai harga terakhir pada saat transaksi ditutup pada hari tersebut.
3. *High*, data nilai transaksi tertinggi pada hari tersebut.
4. *Low*, data nilai transaksi terendah pada hari tersebut
5. *Volume*, Jumlah transaksi pada hari tersebut
   1. ***Machine Learning***

*Machine Learning* merupakan salah satu cabang dari sistem kecerdasan buatan atau *Artificial Intelligence* (AI) yang mencapkup perancangan sistem algoritma yang memungkinkan komputer memiliki kemampuan untuk belajar dari data, tanpa secara eksplisit harus mengikuti instruksi yang diprogram oleh pembuat sistem (Budiharto, 2016). Machine learning berfokus pada pengembangan sistem perangkat lunak yang dapat mengajarkan dirinya sendiri jika diberikan data yang baru.

Ada empat metode machine learning yaitu:

1. *Supervised learning*, data yang diujicobakan sudah memiliki label yang jelas. Contoh algoritma yang menggunakan *supervised learning* adalah *Artificial Neural Network* (ANN), *Support Vector Machine* (SVM), *naïve bayes*, dll.
2. *Unsupervised learning*, data yang digunakan belum memiliki label. Contoh algoritma yang menggunakan *unsupervised learning* adalah *K-Means*, *Association rule*, dll.
3. Semi supervised learning, data yang diujicobakan merupakan gabungan dari *supervised* dan *unsupervised*.
4. *Reinforcement learning*, menciptakan sebuah *agent* yang akan mencari keputusan terbaik. Metode *reinforcement learning* diantaranya *Q-Learning*, *Collaborative reinforcement learning*.
   1. ***Artificial Neural Network* (ANN)**

*Artificial Neural Network* (ANN) adalah metode *machine learning* yang memiliki beberapa lapisan yang biasa disebut *perceptron*. (Setiawan.2020) Pada umumnya, terdapat tiga atau empat lapisan pada arsitektur ANN yaitu *input layer*, *hidden layer*, dan *output layer*. Tahapannya tiap data yang akan di-*input* atau *neuron* akan dikalikan dengan bobot masing-masing *neuron* dan hasil dari semua perkaliannya akan dijumlahkan di *hidden layer*. Selanjutnya, hasil penjumlahan di dalam hidden layer akan masuk ke dalam fungsi aktivasi yaitu *sigmoid* atau *tanh*. *Sigmoid* akan menghasilkan output dengan rentang nilai dari 0 sampai 1, sedangkan fungsi aktivasi *tanh* akan memberikan output dengan rentang nilai dari -1 sampai 1

* 1. ***Long Short-Term Memory* (LSTM)**

Long Short-Term Memory (LSTM) adalah salah satu cabang dari Recurrent Neural Network (RNN) yang diperkenalkan oleh Hochreiter & Schmidhuber (1997), dan disempurnakan dan dipopulerkan oleh banyak orang. Arsitektur LSTM dapat mengatasi masalah jangka panjang (*long term dependencies*) dan oleh karena itu, LSTM biasa digunakan untuk dataset *time series*.

Di dalam arsitektur LSTM terdapat tiga gerbang yang berfungsi untuk menambahkan informasi baru atau menghapus informasi yaitu *forget gate*, *input gate*, dan *output gate*. Ketiga gerbang ini ada di dalam struktur yang biasa disebut dengan *memory cell* yang bisa dilihat ilustrasinya pada gambar di bawah.

A diagram of a process

Description automatically generated

Gambar . Ilustrasi *memory cell*

*Forget gate* berfungsi untuk menentukan informasi apa yang perlu dibuang atau disimpan. Gerbang ini memberikan *output* dengan nilai 0 sampai dengan 1, jika *output* memiliki nilai 0 artinya data dilupakan di *memory cell*.

Selanjutnya, informasi yang sudah diseleksi di *forget gate* akan masuk ke *input gate*. *Input gate* akan memberikan informasi baru di *memory cell* dengan memilih informasi mana yang akan diperbaharui. Setelah itu, informasi akan dilanjutkan ke output gate untuk menentukan informasi apa yang dihasilkan berdasarkan informasi yang sudah diberikan dari *input gate*.

* 1. **Evaluasi Model**

Untuk menguji keakuratan model yang sudah dibuat, Peneliti menggunakan dua metode yaitu *Mean Square Error* (MSE) dan *Root Mean Square Error* (RMSE). Nilai dari MSE bisa didapatkan dengan membagi jumlah semua hasil kuadrat dari selisih nilai asli dan nilai prediksi dengan jumlah data yang digunakan seperti pada rumus berikut.

|  |  |  |
| --- | --- | --- |
|  |  | ( .) |

Keterangan:

MSE = *Mean Square Error*

n = Jumlah seluruh data

y = Nilai sebenarnya

ypred = Nilai prediksi

Sedangakan RMSE merupakan akar kuadrat dari MSE speerti pada rumus berikut.

|  |  |  |
| --- | --- | --- |
|  |  | ( 2.2) |

Keterangan:

RMSE = Root Mean Square Error

n = Jumlah seluruh data

y = Nilai sebenarnya

ypred = Nilai prediksi

# **BAB III**

**ANALISIS DAN PERANCANGAN**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

1. **Fase Eksplorasi**

…………… ……………. …………..…… …….

* + 1. **Kebutuhan Pengguna**

…………… ……………. …………..…… …….

* + 1. **Kebutuhan Data**

…………… ……………. …………..…… …….

* + 1. **Kebutuhan Perangkat Lunak**

…………… ……………. …………..…… …….

* + 1. **Kebutuhan Perangkat Keras**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

…………… ……………. …………..…… …….

…………… ……………. …………..…… …….

…………… ……………. …………..…… …….

* + 1. **Kebutuhan Sistem**

Pada tahapan ini dilakukan survei kepada target pengguna yaitu mahasiswa Teknik Informatika Unpad untuk mendapatkan kebutuhan sistem repositori skripsi yang diinginkan. Dari hasil survei dirancang fitur-fitur seperti yang tertera pada Tabel 3.1 dan Tabel 3.2.

Tabel 3.1 Kebutuhan Sistem Pengguna: Mahasiswa

| **Pengguna : Mahasiswa** | |
| --- | --- |
| **Fitur** | **Deskripsi** |
| Registrasi | Pengguna mendaftarkan diri dengan mengisi nama, NPM, *email*, *password,* dan foto KTM. |
| *Login* | Pengguna *login* dengan mengisi NPM dan *password* |
| Cari Skripsi | Pengguna dapat mencari skripsi berdasarkan judul, nama penulis, kata kunci terkait, dan menyaring skripsi berdasarkan tahun atau kategori skripsi |
| Melihat Skripsi | Pengguna yang telah *login* dapat melihat *file* skripsi |
| Unggah Skripsi | Pengguna dapat mengunggah skripsi |
| Edit *Password* | Pengguna dapat mengubah kata sandi pada halaman profil |
| Status Skripsi | Pengguna dapat mengecek status skripsi apakah telah disetujui admin atau tidak |
| Tampilan *offline* | Web tetap memiliki tampilan pada saat koneksi *offline* |
| *Add to homescreen* | Web dapat ditambahkan ke *homescreen* dan digunakan seperti *native app* |
| Lupa *Password* | Pengguna mendapatkan kata sandi baru yang dikirimkan melalui *email* |
| Kontak admin | Pengguna dapat mengirim pesan ke admin |

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

…………… ……………. …………..…… …….

* + 1. ***User Story***

…………… ……………. …………..…… …….

* 1. **Fase Perencanaan**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

…………… ……………. …………..…… …….

…………… ……………. …………..…… …….

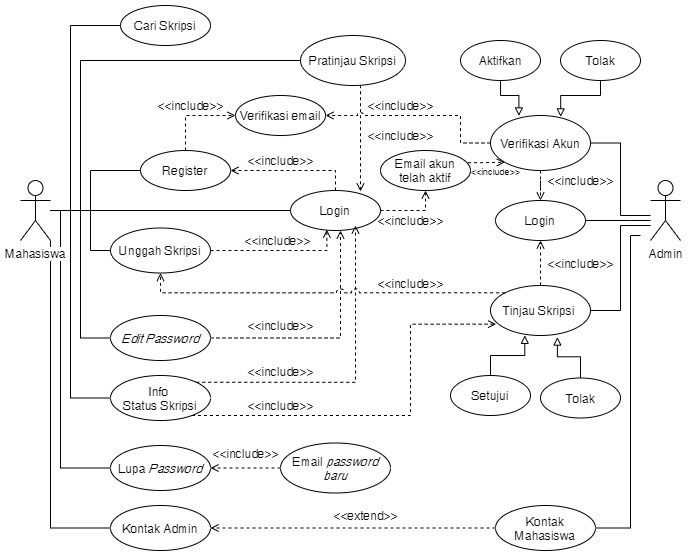
* 1. **Fase Iterasi**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

1. **Analisis Sistem**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

* 1. ***Use Case diagram***

****

Gambar 3.1 *Use Case Diagram*

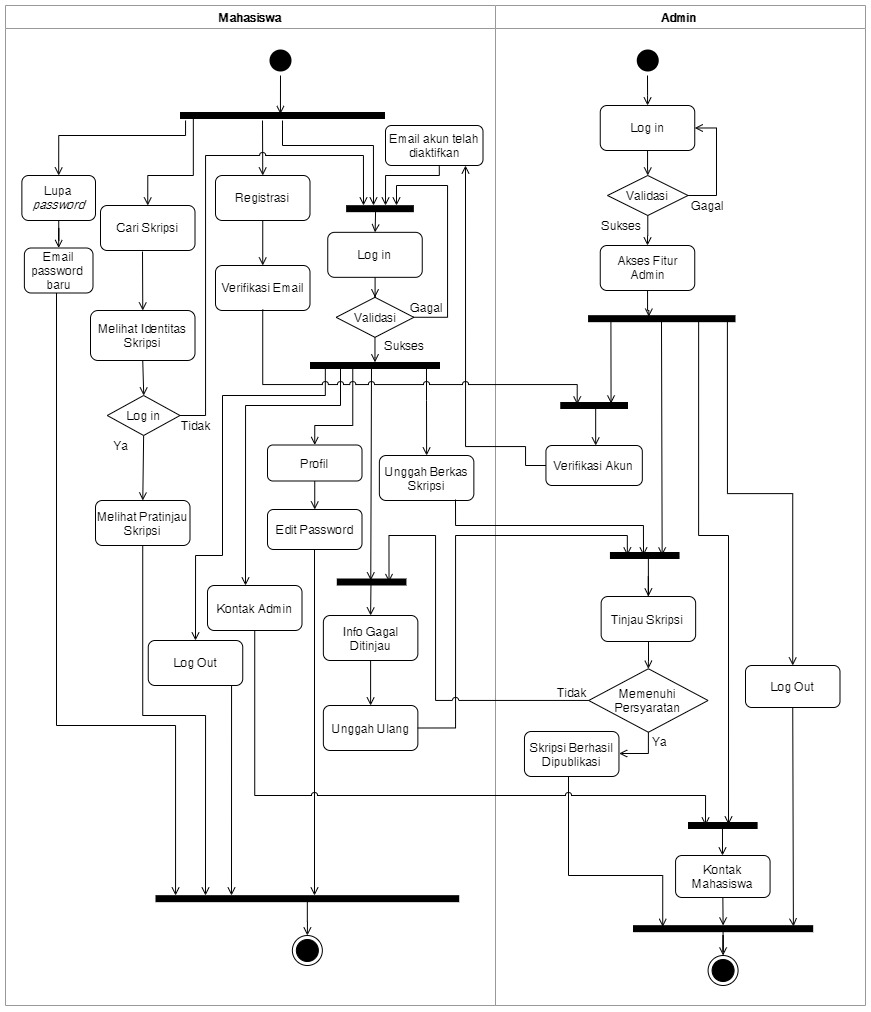
*Use* *case* *diagram* menjelaskan kelakuan dari sistem web repositori skripsi Teknik Informatika Unpad. Pada Gambar 3.2*,* terdapat dua aktor yang merepresentasikan dua jenis pengguna, yaitu mahasiswa dan admin. Mahasiswa dapat melakukan registrasi akun, *login,* pencarian skripsi, melihat skripsi, unggah skripsi, *edit password*, dan melihat info tinjauan skripsi. Pencarian skripsi dapat dilakukan tanpa *login,* sementara fitur-fitur lainnya dapat dilakukan setelah *login*. *Login* dapat dilakukan apabila mahasiswa telah mendaftarkan akun dan akun telah diaktifkan oleh admin. Admin dapat melakukan verifikasi akun dan tinjau skripsi apabila admin telah *login.*

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

* 1. ***Activity diagram***

Diagram aktivitas menggambarkan aliran kerja dari suatu sistem. Pada Gambar 3.3, terdapat dua *swimlane* dengan label mahasiswa dan admin. *Swimlane* memisahkan aktivitas berdasarkan pengguna. Mulai dari status awal, mahasiswa dapat melakukan tiga aktivitas yaitu registrasi, *login*, mencari skripsi. Akun yang didaftarkan akan ditinjau oleh admin terlebih dahulu. Apabila memenuhi syarat maka akun akan diaktifkan. Setelah diaktifkan, mahasiswa dapat *login.*

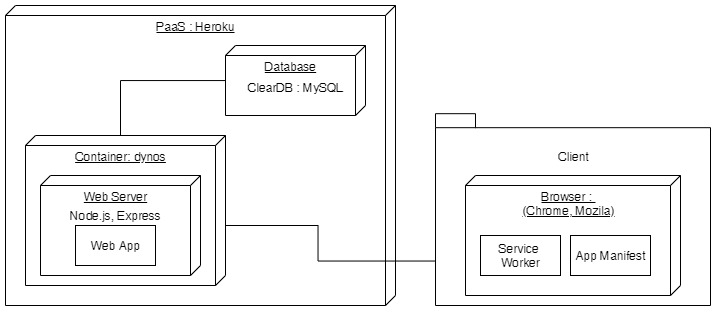
……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….



Gambar 3.2 *Activity Diagram*

1. **Analisis Arsitektur Menggunakan *Deployment Diagram***

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….



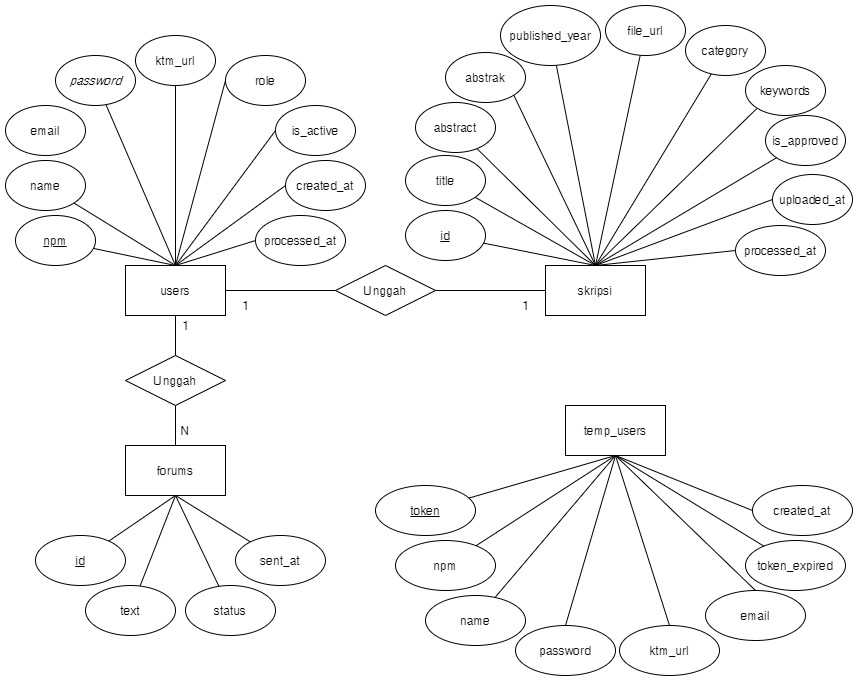
Gambar 3.3 *Deployment Diagram*

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

1. **Desain Basis Data**

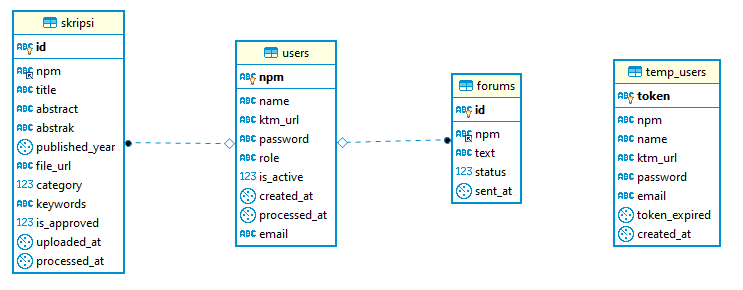
Pada Gambar 3.5 terdapat empat pada entitas dalam ERD yaitu *users, temp\_user,* skripsi, dan *forums*. Tabel *forums* ditambahkan karena hasil *feedback* pada fase pemeliharaan yang menambahkan fitur kontak admin. Tabel *forums* berisi data pesan yang dikirimkan pada fitur kontak admin. Gambar 3.5 adalah ERD yang menjelaskan *database* web aplikasi hasil final rilis.

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….



Gambar 3.4 *Entity Relationship Diagram*

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….



Gambar 3.5 Model Data Fisik

Tabel 3.5 menjelaskan mengenai tipe data, ukuran, dan keterangan dari tiap atribut pada tabel skripsi yang ada pada *database.* Tabel skripsi berisi data dari skripsi yang diunggah oleh mahasiswa*.* Data tersebut terdiri dari id skripsi, judul, abstrak dalam bahasa inggris dan indonesia, *url file* skripsi, kategori, kata kunci, tahun publikasi, waktu diunggah dan ditinjau oleh admin*,* status unggahan skripsi, serta NPM dari penggunaterkait.

Tabel 3.5 Atribut Tabel skripsi

| **No** | **Atribut** | **Tipe** | **Ukuran** | **Keterangan** |
| --- | --- | --- | --- | --- |
| 1 | id | varchar | 5 | *Primary key* sebagai kode identitas skripsi. |
| 2 | npm | varchar | 13 | *Foreign key* dari tabel *users* |
| 3 | title | varchar | 255 | Judul skripsi |
| 4 | abstract | text | - | Abstrak skripsi dalam bahasa inggris |
| 5 | abstrak | text | - | Abstrak skripsi dalam bahasa indonesia |
| 6 | file\_url | varchar | 70 | Url *file* skripsi |
| 7 | category | tinyint | 1 | Kategori skripsi berdasarkan bidang minat. Nilai 1 adalah Sistem Cerdas dan Sistem Grafika (SCSG), 2 adalah Sistem Informasi dan Rekayasa Perangkat Lunak (SIRPL), 3 adalah Minat Jaringan Komputer dan Komunikasi Data (JKKD), dan 4 adalah Ilmu Komputasi dan Metode Numerik (IKMN) |
| 8 | keywords | varchar | 255 | Kata kunci terkait dengan skripsi untuk memudahkan pencarian |
| 9 | published\_year | year | - | Tahun skripsi dipublikasikan |
| 10 | is\_approved | tinyint | 1 | Terdiri dari 0 (ditolak admin), 1 (disetujui oleh admin), dan 2 belum di proses. *Default* awal adalah 2 |
| 11 | uploaded\_at | timestamp | - | Waktu skripsi diunggah |
| 12 | processed\_at | timestamp | - | Waktu skripsi ditinjau oleh admin |

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

1. **Desain Antarmuka**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

……. ……….. …………. ……… …………… . ………… …….. …… …..

……. ……….. …………. ……… …………… . ………… …….. …… …..

……. ……….. …………. ……… …………… . ………… …….. …… …..

1. ***Testing***

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

……. ……….. …………. ……… …………… . ………… …….. ……

……. ……….. …………. ……… …………… . ………… …….. ……

……. ……….. …………. ……… …………… . ………… …….. ……

* 1. **Fase Produksi**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

……. ……….. …………. ……… …………… . ………… …….. ……

……. ……….. …………. ……… …………… . ………… …….. ……

……. ……….. …………. ……… …………… . ………… …….. ……

* 1. **Fase Pemeliharaan dan Fase Akhir**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

……. ……….. …………. ……… …………… . ………… …….. ……

……. ……….. …………. ……… …………… . ………… …….. ……

……. ……….. …………. ……… …………… . ………… …….. ……

# **BAB IV**

**HASIL DAN PEMBAHASAN**

1. **Impelementasi Program**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… ... …… …….

1. **Halaman Utama**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… . …… …….



(a)



(b)

Gambar 4.1 Halaman Utama pada *Desktop* (a) dan *Mobile* (b)

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ………

Script/kode program halaman utama :

|  |
| --- |
| router.get('/list', (req, res) =>{  let sql = `SELECT skripsi.id, skripsi.title, skripsi.published\_year, skripsi.category, skripsi.keywords, users.name FROM skripsi join users on users.npm = skripsi.npm where is\_approved=${1} ORDER BY published\_year desc, skripsi.processed\_at desc`  db.query(sql, (err, result)=>{  if (err) console.log(err)  res.send(result)})  }) |

*Request* dipanggil dalam *componentDidMount* *method*, yang berarti *request* dilakukan setelah inisial *rendering* dilakukan. Berikut adalah kode *request* data skripsi pada *frontend:*

Script/kode program *request* :

|  |
| --- |
| getSkripsi=()=>{ //fungsi request data skripsi menggunakan axios  axios({   method: 'get',    url: '/skripsi/list',   }).then(res=>{  this.setState({     skripsi: res.data,     isLoaded: true,     skripsiFiltered:res.data,     skripsiFilteredTemp:res.data,     skripsiFilteredCat:res.data,     skripsiFilteredYear:res.data,     years: [...new Set(res.data.map((year)=>{        return year.published\_year      }))].sort()  })    localStorage.setItem('list', JSON.stringify(res.data))   }).catch((err) => {    if(err.response) console.log(err.response)   })  }  componentDidMount(){  if (navigator.onLine){ //Saat online, lakukan request data    this.getSkripsi()    this.setState({     offline:false    })  }   else{ //Saat offline ambil data dari local storage    if (localStorage.getItem('list')){     let data = JSON.parse(localStorage.getItem('list'))     this.setState({      skripsi: data,      isLoaded: true,      skripsiFiltered:data,      skripsiFilteredTemp:data,      skripsiFilteredCat:data,      skripsiFilteredYear:data,      years: [...new Set(data.map((year)=>{              return year.published\_year             }))].sort()     })    } }  } |

Data yang didapat dikirimkan sebagai *props* ke *component list* untuk di-*loop* dan ditampilkan ke dalam bentuk *card* pada halaman utama. Pada kode dibawah dapat dilihat bahwa data yang dikirimkan merupakan data hasil *filter* sehingga tiap pencarian maupun penyaringan akan mengubah data yang tampil. Fungsi filter akan dibahas lebih lanjut pada bagian fitur pencarian dan penyaringan.

Script/kode program *filter* :

|  |
| --- |
| const currentPosts = skripsiFiltered.slice(indexOfFirstPost, indexOfLastPost)  <ListCard skripsi={currentPosts} isLoaded={isLoaded}> </ListCard> |

1. **Halaman Registe****r**

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ………

Tabel . *Black Box Testing* pada Fitur Registrasi

| **Skenario** | **Hasil yang diharapkan** | **Hasil** |
| --- | --- | --- |
| Tidak memberi masukan apapun pada *form* data diri | Pengguna tidak dapat melanjutkan ke bagian dua karena tombol lanjuttidak aktif | Berhasil |
| Mengosongkan salah satu *field* | Pengguna tidak dapat melanjutkan ke bagian dua karena tombol lanjuttidak aktif | Berhasil |
| Mengisi NPM atau *email* yang telah terdaftar | Muncul pesan bahwa NPM atau *email* telah terdaftar setelah pengguna mengklik tombol lanjut | Berhasil |
| Mengisi NPM yang salah (bukan angka, !=12 digit) | Muncul pesan bahwa NPM harus 12 digit setelah pengguna selesai menulis NPM dan tombol lanjut tidak aktif | Berhasil |
| Tidak mengisi kolom konfirmasi *password* dengan benar | Muncul pesan bahwa *password* tidak cocok. Pengguna tidak dapat melanjutkan ke bagian dua karena tombol lanjuttidak aktif | Berhasil |
| Mengisi semua *field* data diri dengan benar | Pengguna dapat lanjut ke bagian dua | Berhasil |
| Tidak mengunggah foto KTM | Muncul pesan bahwa *file* tidak boleh kosong | Berhasil |
| Mengunggah *file* yang bukan jpeg atau png | Muncul pesan bahwa *file* harus png, jpg, atau jpeg | Berhasil |
| Menggunggah *file* yang lebih besar dari 5mb | Muncul pesan bahwa *file* terlalu besar | Berhasil |
| Mengunggah *file* foto KTM dengan format jpg/png dengan besar lebih kecil dari 5MB | Muncul pesan bahwa pengguna harus melakukan verifikasi melalui *link* yang dikirimakan ke *email*. *Email* dikirimkan ke pengguna. | Berhasil |
| Mengakses *link* yang dikirimkan ke *email* | Muncul pesan bahwa register berhasil dan pengguna diharap menunggu pengaktifan dari admin. Data pengguna tersimpan di tabel *users* dan dihapus dari *temp\_users* | Berhasil |

……. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ……… …………… . ………… …….. …… ….. ……… …………… . ………… …….. …… ….. ……….. …………. ……… …………… . ………… …….. …… ….. …. …… …. …. …. …. ……. .. …… ………

1. **Menu *Login***

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Halaman Unggah Skripsi**

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Halaman Menu Admin**

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Halaman Verifikasi Akun**

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Halaman Tinjau Skripsi**

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Halaman Detail Skripsi**

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Fitur Pencarian dan Penyaringan**

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Halaman Profil danFitur *Edit Password***

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Halaman Status Skripsi**

……. ……….. …………. ……… …………… . ………… …….. …… ………..

……. ……….. …………. ……… …………… . ………… …….. …… ………..

1. **Implementasi PWA**

……. ……….. …………. ……… …………… . ………… …….. …… ………..

……. ……….. …………. ……… …………… . ………… …….. …… ………..

1. **Fase Produksi**

……. ……….. …………. ……… …………… . ………… …….. …… ……….. ……. ……….. …………. ……… …………… . ………… …….. …… ………..

1. **Rilisan Kecil**

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Pengujian**

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Fase Pemeliharaan dan Fase Akhir**

……. ……….. …………. ……… …………… . ………… …….. ……

1. ***Feedback***

……. ……….. …………. ……… …………… . ………… …….. ……

1. **Implementasi *Feedback***

……. ……….. …………. ……… …………… . ………… …….. ……

# **BAB V**

**KESIMPULAN DAN SARAN**

1. **Kesimpulan**

Berdasarkan berbagai hal yang telah disampaikan pada bab sebelumnya serta perancangan dan implementasi web repositori yang sudah dilakukan penulis, maka dapat diambil simpulan sebagai berikut:

……….. ………….. . ……….. ……….. ……… ………… …………. ……. ……….. …….. . ……….. …….. ……….. .

……….. ………….. . ……….. ……….. ……… ………… …………. ……. ……….. …….. . ……….. …….. ……….. .

……….. ………….. . ……….. ……….. ……… ………… …………. ……. ……….. …….. . ……….. …….. ……….. .

1. **Saran**

Berdasarkan penelitian yang dilakukan, penulis memberikan saran yang dapat

diimplementasikan pada penelitian selanjutnya, yaitu sebagai berikut:

1. 1. ……….. ………….. . ……….. ……….. ……… ………… …………. ……. ……….. …….. . ……….. …….. ……….. .
2. 2. ……….. ………….. . ……….. ……….. ……… ………… …………. ……. ……….. …….. . ……….. …….. ……….. .
3. 3. ……….. ………….. . ……….. ……….. ……… ………… …………. ……. ……….. …….. . ……….. …….. ……….. .

# **DAFTAR PUSTAKA**

Sukamto, R.A. dan Shalahuddin, M. (2018) *Rekayasa Perangkat Lunak*. Bandung: Informatika.

Ater, T. (2017) *Building Progressive Web Apps: Bringing the Power of Native to the Browser*. Sebastopol: O’Reilly Media, Inc.

Beck, K. and Andreas, C. (2004) *Extreme Programming Explained: Embrace Change*. Boston: Addison Wesley.

Beck, K. and Fowler, M. (2000) *Planning Extreme Programming*. Addison Wesley.

Ferguson, R. (2019) *Beginning JavaScript. The Ultimate Guide to Modern JavaScript Development*. New Jersey: Apress.

Freeman, A. (2019) *Pro React 16*. London: Apress.

Gs.statcounter.com (2020) *Desktop vs Mobile vs Tablet Market Share Worldwide StatCounter Global Stats*. Available at: https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet (Accessed: 1 June 2020).

Hahn, E. M. (2016) *Express In Action. Writing, building, and testing Node.js applications*. New York: Manning Publications Co.

Handiwidjojo, W. dan Ernawati, L. (2016) ‘Pengukuran Tingkat Ketergunaan (Usability) Sistem Informasi Keuangan Studi Kasus: Duta Wacana Internal Transaction (Duwit)’, *JUISI*, 2(1).

Heroku.com (2020) *What is Heroku?* Available at: https://www.heroku.com/about (Accessed: 1 June 2020).

Heroku (2020) *Heroku Dynos*. Available at: https://www.heroku.com/dynos (Accessed: 1 June 2020).

Hume, D. A. (2017) *Progressive Web Apps*. New York: Manning Publications.

Jaya, T. S. (2018) ‘Pengujian Aplikasi dengan Metode Blackbox Testing Boundary Value Analysis (Studi Kasus: Kantor Digital Politeknik Negeri Lampung)’, *Jurnal Informatika: Jurnal Pengembangan IT (JPIT)*, 3(2).

Karpagam, D. V. *et al.* (2017) ‘Performance Enhancement of Webpage Using Progressive Web App Features’, *International Journal of Innovative Research in Advanced Engineering (IJIRAE)*, 4(3).

Krishna, T. S. R. *et al.* (2011) ‘Survey on Extreme Programming in Software Engineering’, in *International Journal of Computer Trends and Technology*.

Maryuliana. dkk. (2016) ‘Sistem Informasi Angket Pengukuran Skala Kebutuhan Materi Pembelajaran Tambahan Sebagai Pendukung Pengambilan Keputusan Di Sekolah Menengah Atas Menggunakan Skala Likert’, *Jurnal Transistor Elektro dan Informatika (TRANSISTOR EI)*, 1(2).

Mehta, C. *et al.* (2018) *MySQL 8 Administrator’s Guide*. Birmingham: Packt.

Moran, K. (2019) *Usability Testing 101*. Available at: https://www.nngroup.com/articles/usability-testing-101/ (Accessed: 1 June 2020).

Nadia, R., dkk. (2018) ‘Rancang Bangun Aplikasi CallTenant dengan Penyimpanan Basis Data untuk Form Dinamis Menggunakan Framework Laravel’, *JURNAL TEKIK*, 7(1).

Nielsen, J. (1993) *Usability Engineering*. Morgan Kaufmann.

Nielsen, J. (1994) *10 Heuristics for User Interface Design: Article by Jakob Nielsen*. Available at: https://www.nngroup.com/articles/ten-usability-heuristics/ (Accessed: 2 June 2020).

Repanovici, A. (2009) ‘Marketing Research about Attitudes, Difficulties and Interest of Academic Community about Institutional Repository’, *Proceedings of the 3rd International Conference in Management, Marketing and Finances*, MMF’09, pp. 88–95.

Ropianto, M. (2016) ‘Pemahaman Penggunaan Unified Modelling Language’, *JT-IBSI*, 01(01).

Santoni, M. (2018) *Progressive Web Apps browser support & compatibility*. Available at: https://www.goodbarber.com/blog/progressive-web-apps-browser-support-compatibility-a883/ (Accessed: 20 January 2019).

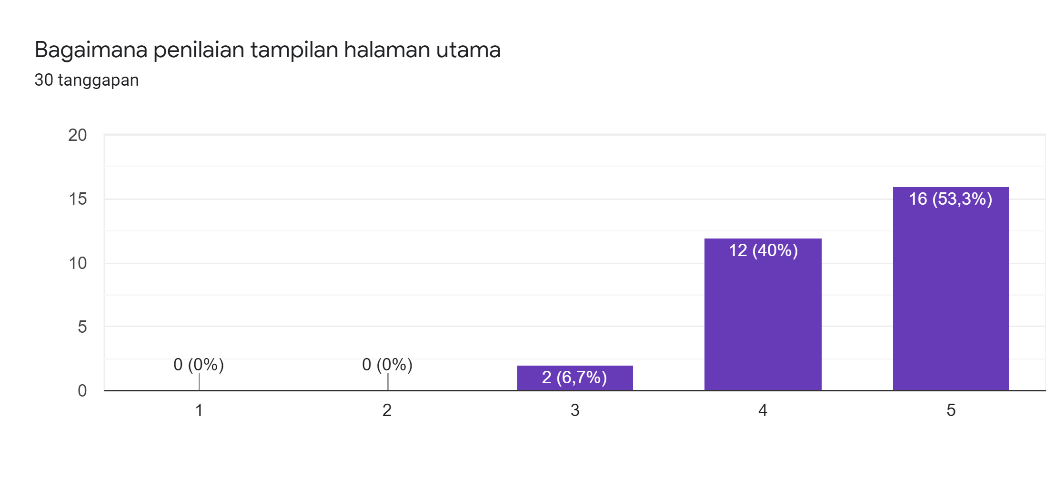
Singh, H. and Bhatt, M. (2016) *Learning Web Development with React and Bootstrap*. Birmingham: Packt Publishing Ltd.

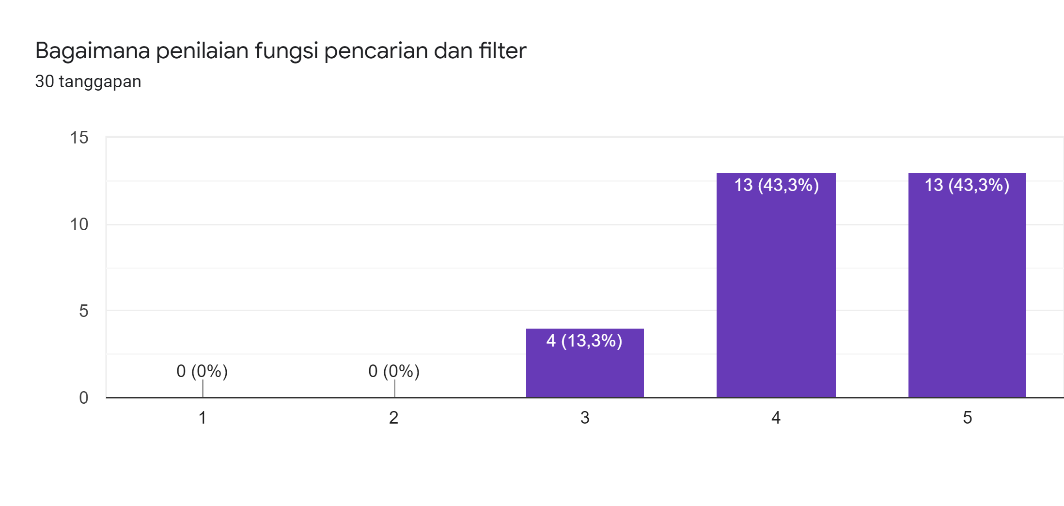
Spillner, A., *et al.* (2014) *Software Testing Foundations*. Santa Barbara: Rocky Nook Inc.

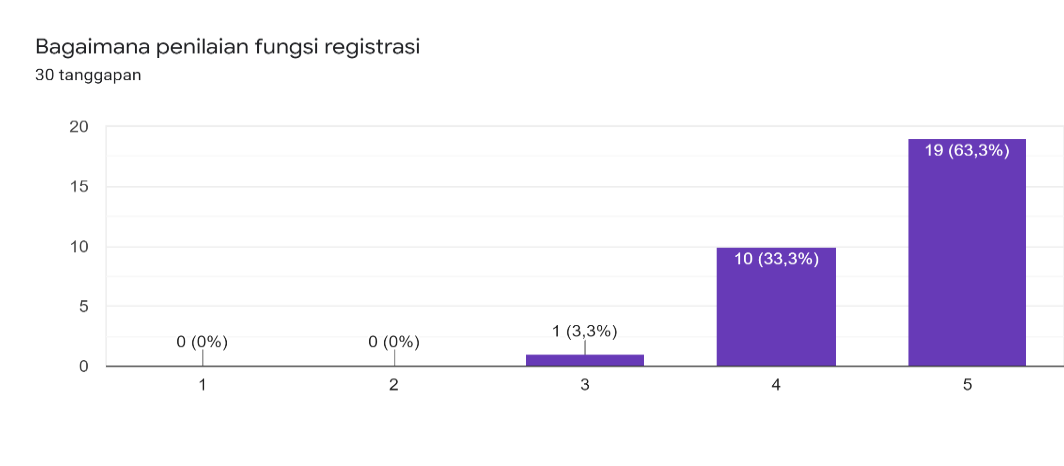
Sutedjo, M. (2014) ‘Pengelolaan Repositori Perguruan Tinggi dan Pengembangan Repositori Karya Seni’, *Makalah Seminar Nasional ”Digital Local Content: Strategi Membangun Repository Karya Seni”, GKU FSR ISI Yogyakarta*.

# **LAMPIRAN**

**Lampiran 1 Ringkasan Hasil *Usability Testing***

****

****

****

**Lampiran 2** **Kode Web Aplikasi**

Register

|  |
| --- |
| import React, { Component } from 'react'  import bg2 from '../icons/bg2.webp'  import axios from 'axios'  import { scrollToTop } from '../helpers/autoScroll'  import { ProgressBar, Modal} from 'react-bootstrap'  import { Redirect } from 'react-router-dom'  import { connect } from 'react-redux'  export class Register extends Component {  initialState = {  showLoading:false,  name:'',  npm: '',  email:'',  pass:'',  rePass: '',  message: '',  status:'',  displayForm1:'block',  displayForm2:'none',  displayForm3:'none',  file:null,  progress:34  }  state=this.initialState  next = (e) =>{  e.preventDefault()  this.setState({  showLoading:true,  })  let {name, npm, pass, email}= this.state  let data={  name:name,  email:email,  npm:npm,  password:pass  }  axios({  method: 'POST',  url: '/check-form',  data: data  }).then(res => {  this.setState({  message: '',  displayForm1: 'none',  displayForm2: 'block',  progress: this.state.progress + 33,  showLoading:false  })  scrollToTop()  }).catch(err => {  this.setState({  showLoading:false  })  if (err.response) {  this.setState({  message: err.response.data.message,  status: err.response.data.status,  })  }  else{  this.setState({  message: 'Network error, Cek Koneksi Anda',  status: 500,  })  }  })  } |

*Login*

|  |
| --- |
| import React, { PureComponent } from 'react'  import UserMenu from '../components/UserMenu'  import AdminMenu from '../components/AdminMenu'  import { Link, Redirect } from 'react-router-dom'  import { setToken, delToken} from '../reducers/authReducer'  import { connect } from 'react-redux'  import { Modal } from 'react-bootstrap'  import { FaRegCheckCircle } from 'react-icons/fa'  import '../styles/nav.css'  import axios from 'axios'  import { scrollToTop } from '../helpers/autoScroll'  import MediaQuery from 'react-responsive'  export class Nav extends PureComponent {  state = {  npm: '',  pass: '',  message: '',  status: null,  showLogin: false,  showLoading:false,  justLoggedIn:false, //ketika pertama kali login, agar refresh tidak redirect  }  handleInput = (e) =>{  this.setState({  status:'',  [e.target.id] : e.target.value  })  }  submitLogin = e => {  e.preventDefault()  this.setState({  showLoading:true  })  axios({  method: 'post',  url: '/login',  data: {  npm: this.state.npm,  password: this.state.pass  }  }).then(res => {  let loginInfo = res.data.data  console.log(loginInfo)  if (loginInfo.isLogged){ //response didapat  this.props.login(loginInfo) //set state global  this.setState({ //show modal  showLoading:false,  status:res.data.status,  justLoggedIn:true,  showLogin:true,  })  scrollToTop()  setTimeout(() =>  this.setState({ //hide modal  showLogin:false  }), 1000)  }  else{  this.setState({  showLoading:false,  status:500  })  }  }).catch((err) => {  if(err.response){  this.setState({  message:err.response.data.message,  status:err.response.data.status,  showLoading:false,  })  } else{  this.setState({  status: 500,  showLoading:false,  })  }  })  }  logout = () =>{  this.props.logout()  }  handleClose = () => {  this.setState({  showLogin:false  })  } |

Unggah

|  |
| --- |
| import React, { Component } from 'react'  import { connect } from 'react-redux'  import { Redirect, Link } from 'react-router-dom'  import axios from 'axios'  import { Spinner, Modal } from 'react-bootstrap'  export class Upload extends Component {  initialState={  skripsi:{},  isLoaded:false,  offline: false,  showLoading:false,  title:'',  titleAlert:'initial',  year:'',  yearAlert:'initial',  abstract:'',  abstractAlert:'initial',  abstrak:'',  abstrakAlert:'initial',  category:'',  keywords:'',  file:null,  message: '',  status:'',  }  state=this.initialState  submit = (e) =>{  e.preventDefault()  this.setState({  showLoading:true  })  let {title, year, abstrak, abstract, category, keywords} = this.state  let {file} = this.state  const formData = new FormData()  formData.append('file', file)  formData.append('title', title)  formData.append('year', year)  formData.append('abstrak', abstrak)  formData.append('abstract', abstract)  formData.append('category', category)  formData.append('keywords', keywords)  axios({  method: 'POST',  url: `/user/upload/`,  data: formData,  headers:{  'Content-Type':'multipart/form-data',  'Authorization': this.props.token  }  }).then((res) =>{  this.refs.uploadForm.reset()  this.setState({  message:res.data.message,  status:res.data.status,  showLoading:false  })  }).catch((err) => {  this.setState({  showLoading:false  })  if( err.response){  this.setState({  message:err.response.data.message,  status:err.response.data.status,  })  }  })  }  handleInput = (e) =>{  if(e.target.id==='title'){  e.target.value=e.target.value.replace(/\n/g, ' ')  }  this.setState({  [e.target.id] : e.target.value,  })  if (e.target.id==='title'){  this.setState({  titleAlert: e.target.value,  })  }  else if (e.target.id==='year'){  this.setState({  yearAlert: e.target.value,  })  }  else if (e.target.id==='abstract'){  this.setState({  abstractAlert: e.target.value,  })  }  else if (e.target.id==='abstrak'){  this.setState({  abstrakAlert: e.target.value,  })  }  } |

Detail Skripsi

|  |
| --- |
| import React, { Component } from 'react'  import { Spinner } from 'react-bootstrap'  import axios from 'axios'  import { connect } from 'react-redux'  import { FaChevronLeft, FaChevronRight } from 'react-icons/fa'  import {Document, pdfjs, Page} from 'react-pdf'  pdfjs.GlobalWorkerOptions.workerSrc = `//cdnjs.cloudflare.com/ajax/libs/pdf.js/${pdfjs.version}/pdf.worker.js`  export class SkripsiDetail extends Component {  state={  skripsi:[],  isLoaded:false,  offline:false,  pageNumber:1,  numPages:null  }  getData =()=>{  let id = this.props.match.params.id  if(this.props.token){  axios({  method: 'get',  url: `/skripsi/detail/`,  params:{  id : id  },  headers: {  Authorization: this.props.token  }  }).then(res=>{  this.setState({  skripsi: res.data[0],  isLoaded: true  })  }).catch(err=>{  if(err.response){  console.log(err.response)  }  })  }  else{  axios({  method: 'get',  url: `/skripsi/info/`,  params:{  id : id  }  }).then(res=>{  this.setState({  skripsi: res.data[0],  isLoaded: true  })  }).catch(err=>{  if(err.response){  console.log(err.response)  }  })  }  }  onDocumentLoadSuccess = ({ numPages }) => {  this.setState({ numPages });  }  next = () => {  let {pageNumber, numPages} = this.state  if( pageNumber<=numPages){  this.setState({pageNumber:this.state.pageNumber+1})  }  }  before = () => {  let {pageNumber} = this.state  if( pageNumber>1){  this.setState({pageNumber:this.state.pageNumber-1})  }  }  componentDidMount(){  if (navigator.onLine){  this.getData()  this.setState({  offline:false  })  }  else{  this.setState({  offline:true,  })  }  } |

Profil dan Edit *Password*

|  |
| --- |
| import React, { Component } from 'react'  import { Spinner, Modal} from 'react-bootstrap'  import { connect } from 'react-redux'  import axios from 'axios'  import moment from 'moment'  export class ProfileInfo extends Component {  state={  user:{},  isLoaded:false,  showLoading:false,  offline:false,  newPass:'',  oldPass:'',  confirmPass:'',  message:'',  status:null  }  getProfile= ()=>{  axios({  method: 'get',  url: `/user/profile/`,  headers: {  Authorization: this.props.token  }  }).then(res=>{  this.setState({  user: res.data,  isLoaded: true  })  }).catch(err=>{  console.log(err.response)  })  }  componentDidMount(){  if (navigator.onLine){  this.getProfile()  this.setState({  offline:false  })  }  else{  this.setState({  offline:true,  })  }  } |

Status Skripsi

|  |
| --- |
| import React, { Component } from 'react'  import { connect } from 'react-redux'  import axios from 'axios'  import {Spinner} from 'react-bootstrap'  import { FaCheck, FaFilePdf, FaTimes } from 'react-icons/fa'  import { Link } from 'react-router-dom'  import moment from 'moment'  export class SkripsiStatus extends Component {  state={  skripsi:{},  isLoaded:false,  Offline:false  }  getSkripsi=()=>{  axios({  method: 'get',  url: `/user/skripsi/`,  headers: {  Authorization: this.props.token  }  }).then(res=>{  this.setState({  skripsi: res.data,  isLoaded: true  })  }).catch(err=>{  if(err.response){  console.log(err.response)  }  })  }  componentDidMount(){  if (navigator.onLine){  this.getSkripsi()  this.setState({  offline:false  })  }  else{  this.setState({  offline:true,  })  }  } |

Kontak Admin

|  |
| --- |
| import React, { Component } from 'react'  import { Redirect } from 'react-router-dom'  import { Spinner } from 'react-bootstrap'  import { connect } from 'react-redux'  import axios from 'axios'  import Forum from '../components/Forum'  class UserForum extends Component {  state={  text:'',  dataLoaded:false,  isLoading:false,  chats:[],  offline:false,  message:''  }  handleText = e =>{  this.setState({  [e.target.id] : e.target.value.trim()  })  }  submit = e => {  let {text} =this.state  e.preventDefault()  if(text){  this.setState({  isLoading:true  })  axios({  method: 'post',  url: '/user/insert-text',  headers: {  Authorization:this.props.token  },  data: {  text: text,  }  }).then(res => {  this.setState({  isLoading:false,  text:''  })  this.getForum()  this.refs.messages.reset()  }).catch(err=>{  console.log(err.response)  if(err.response){  this.setState({  message:err.response.data.message  })  }  this.setState({  isLoading:false,  })  setTimeout(() =>  this.setState({  message:''  }), 5000)  })  }  }  getForum= ()=>{  axios({  method: 'get',  url: '/user/forum',  headers: {  Authorization:this.props.token  }  }).then(res=>{  this.setState({  dataLoaded:true,  chats:res.data  })  }).catch((err) => {  if(err.response){  console.log(err.response.statusText)  }  this.setState({  dataLoaded:true  })  })  }  componentDidMount(){  if (navigator.onLine){  this.getForum()  this.setState({  offline:false  })  }  else{  this.setState({  offline:true,  })  }  } |

Lupa *Password*

|  |
| --- |
| import React, { Component } from 'react'  import { Spinner } from 'react-bootstrap'  import { connect } from 'react-redux'  import { Redirect } from 'react-router-dom'  import Bg3 from '../components/Bg3'  import axios from 'axios'  export class Forgot extends Component {  state={  email:'',  npm:'',  checknpm:'',  displaySection1:'block',  displaySection2:'none',  sending:false,  status:'',  message:''  }  handleInput = e => {  this.setState({  [e.target.id] : e.target.value,  })  }  checkNpm = e =>{  this.setState({  checknpm : e.target.value,  })  }  sendEmail = e =>{  e.preventDefault()  this.setState({  sending:true  })  axios({  method: 'put',  url: '/forgot-pass',  data: {  npm: this.state.npm,  email: this.state.email  }  }).then(res => {  this.refs.forgotForm.reset()  this.setState({  sending:false,  status:res.data.status,  message:res.data.message,  displaySection1:'none',  displaySection2:'block'  })  }).catch((err) => {  console.log(err.response)  if(err.response){  this.setState({  status:err.response.data.status,  message:err.response.data.message,  sending:false,  })  }  else{  this.setState({  message: 'Network error, Cek Koneksi Anda',  status: 500,  sending:false,  })  }  })  } |

PWA

|  |
| --- |
| //Lifecycle = register -> install -> activate https://bit.ly/CRA-PWA  //ip localhost ([::1] - 127.0.0.0/8 )  const isLocalhost = Boolean(  window.location.hostname === 'localhost' ||  window.location.hostname === '[::1]' ||  window.location.hostname.match(  /^127(?:\.(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)){3}$/  )  )  export function register(config) {  //Cek apakah service worker support pada browser  if ('serviceWorker' in navigator) {  const publicUrl = new URL(process.env.PUBLIC\_URL, window.location.href)  //Service worker tidak jalan jika public file diluar url  if (publicUrl.origin !== window.location.origin) {  return  }  //Ketika halaman telah di load  window.addEventListener('load', () => {  const swUrl = `${process.env.PUBLIC\_URL}/sw-custom.js`  if (isLocalhost) {  checkValidServiceWorker(swUrl, config)  navigator.serviceWorker.ready.then(() => {  console.log('Web app served cache-first')  })  } else {  //Register SW  registerValidSW(swUrl, config)  }  })  }  }  function registerValidSW(swUrl, config) {  //------Register SW  navigator.serviceWorker  .register(swUrl)  .then(registration => {  registration.onupdatefound = () => {  const installingWorker = registration.installing  if (installingWorker == null) {  return  }  installingWorker.onstatechange = () => {  //------------install SW  if (installingWorker.state === 'installed') {  //Cek apakah sw aktif  if (navigator.serviceWorker.controller) {  // Update precached content di fetch tapi SW lama masih di jalankan sampai tab di tutup  console.log('Content baru telah di fetch dan siap digunakan saat tab telah ditutup')  if (config && config.onUpdate) {  config.onUpdate(registration)  }  } else {  console.log('Content telah di cached untuk mode offline.')  if (config && config.onSuccess) {  config.onSuccess(registration)  }  }  }  }  }  })  .catch(error => {  console.error('Error saat register service worker:', error)  })  }  function checkValidServiceWorker(swUrl, config) {  // Cek apakah SW sudah ada  fetch(swUrl, {  headers: { 'Service-Worker': 'script' }  })  .then(response => {  const contentType = response.headers.get('content-type')  if (response.status === 404 || (contentType != null && contentType.indexOf('javascript') === -1)) {  // SW tidak di temukan  navigator.serviceWorker.ready.then(registration => {  registration.unregister().then(() => {  window.location.reload()  })  })  } else {  // SW ditemukan. Proses dilanjutkan  registerValidSW(swUrl, config)  }  })  .catch(() => {  console.log('Tidak ada koneksi Internet. Web App dalam offline mode.')  })  }  export function unregister() {  if ('serviceWorker' in navigator) {  navigator.serviceWorker.ready.then(registration => {  registration.unregister()  })  }  } |

# **RIWAYAT HIDUP**