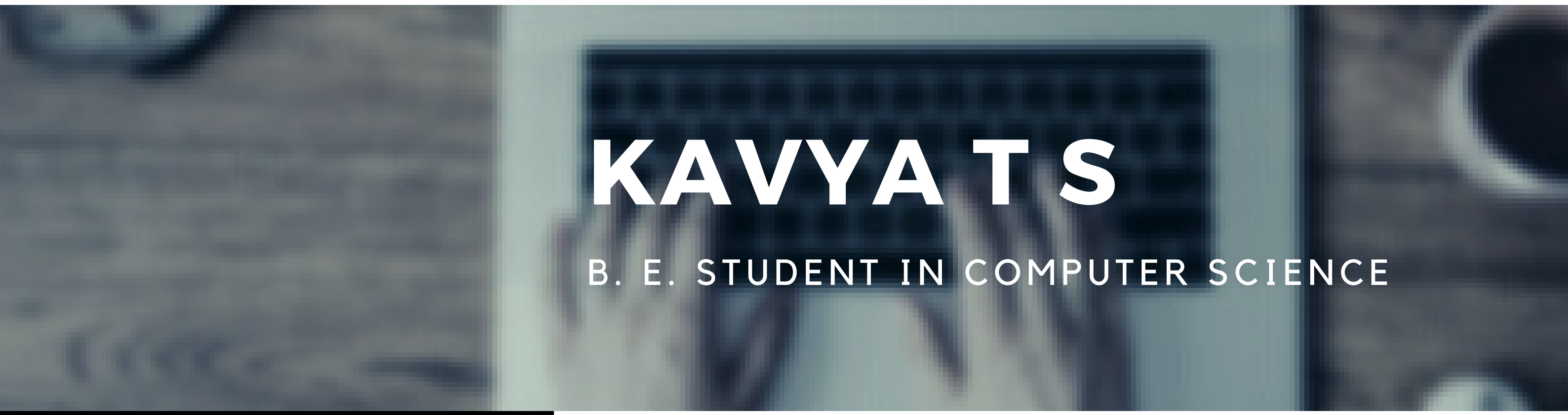

KAVYA T S

B. E. STUDENT IN COMPUTER SCIENCE

TECH STACK

- Java
- Python
- HTML
- OOP
- MySQL
- Machine Learning

SOFT SKILLS

- Communication
- Creativity
- Leadership
- Adaptability
- Analytical thinking
- Team work
- Determination
- Smart work
- Time management

CONTACT

Tumkur

kavyatsofficial@gmail.com

+91 8762029456

linkedin.com/in/kavya-ts

github.com/kavyapangar

instagram.com/kavya__ts

LANGUAGES

English
Full Professional Proficiency

Kannada
Native or Bilingual Proficiency

Hindi
Full Professional Proficiency

Telugu
Limited Working Proficiency

INTERESTS

- Artificial Intelligence (AI)
- Machine Learning
- Learning new life skills
- Cooking
- Dancing
- Cycling
- Trekking
- Crafts & Art work

EDUCATION

B. E. in Computer Science & Engineering (2018 - 2022) **7.80 CGPA**
Channabasaveshwara Institute of Technology, Gubbi (Affiliated to VTU - Belagavi)

2nd PUC (2018) **84.16%**
Sarvodaya Pre-university College, Tumkur (Karnataka State Board)

SSLC (2016) **92.80%**
Maruthi Vidya Kendra , Tumkur (Karnataka State Board)

CERTIFICATES

- **Google IT support professional certificate**
- **IT Fundamentals for Cybersecurity Specialization (IBM)**
- **Cybersecurity Workshop by Pravega & IISc, Bangalore (2020)**
- **24 Hours Hackathon - 2019**
- **Cloud Computing Basics**
- **Winner- NDLI Quiz - 2019**
- **Ideathon - 2020**

ACADEMIC PROJECTS

Text Classification using BERT
2022
- Languages used:Python

Question and Answer using BERT
2021 (Eunoia labs, Tumkur)
- Software used: Google Colab
- Languages used: Python

Online Cake Ordering System
2020
- Languages used: HTML, CSS, PHP, SQL

Android Run Tracker Application
2021
- Software used: Android Studio
- Languages used: Java, XML

2D Helicopter Game
2021
- Software used: Visual Studio
- Languages used: C++

Technologies implemented:
- Sci-kit learn, BERT, Matplotlib, Transformers, pandas

PROFESSIONAL EXPERIENCE

- **Digital head at Sublime Camps Pvt. Ltd.**
  2021-present
- **The International Model United Nations (TIMUN) - 2019**
  International Conference, Colombo, Sri Lanka
- **Habitat for Humanity (Volunteer work)**
  Colombo, Sri Lanka (2019)

EXTRACURRICULAR ACCOMPLISHMENTS

- **Student Event Chair, SPARK-IT Technical Club, CIT**
  2021-2022
- **58 KM Cycling event - 2020 by RHH & FIT INDIA**
- **Student Member of IEEE CIT-SB**
  Since 2019 ID: STD00421
- **Organizing Committee Member of SPANDANA-2019**
  an intercollege technical fest
- **VTU Fest Participation -2019**
  SKIT team from CIT

BITWISE OPERATORS:

In Java, bitwise operators are used to perform operations on individual bits of integer values. Java provides several bitwise operators that can be used to manipulate and analyze bits at a low level. The types of bitwise operators are as follows:

1. **Bitwise AND (&):**

The bitwise AND operator compares the corresponding bits of two operands and produces a result where each bit is set to 1 if both corresponding bits are 1; otherwise, it sets the bit to 0.

Example:

```
int a = 5;      // binary: 0101
int b = 3;      // binary: 0011
int result = a & b;   // binary: 0001, decimal: 1
System.out.println(result);

Output: 1(0001)
```

2. **Bitwise OR (|):**

The bitwise OR operator compares the corresponding bits of two operands and produces a result where each bit is set to 1 if either of the corresponding bits is 1; otherwise, it sets the bit to 0.

Example:

```
int a = 5;      // binary: 0101
int b = 3;      // binary: 0011
int result = a | b;   // binary: 0111, decimal: 7
System.out.println(result);

Output: 7(0111)
```

3. **Bitwise XOR (^):**

The bitwise XOR (exclusive OR) operator compares the corresponding bits of two operands and produces a result where each bit is set to 1 if only one of the corresponding bits is 1; otherwise, it sets the bit to 0.

Example:

```
int a = 5;      // binary: 0101
int b = 3;      // binary: 0011
int result = a ^ b;   // binary: 0110, decimal: 6
System.out.println(result);

Output: 6(0110)
```

4. **Bitwise NOT (~):**

The bitwise NOT operator is a unary operator that flips the bits of its operand. It sets each bit to the opposite of its current value, resulting in a one's complement of the operand.

Example:
```
int a = 5;      // binary: 0101
int result = ~a;     // binary: 1010, decimal: -6 (due to two's complement representation)
System.out.println(result);

Output: -6 (1010)
```

5. **Left Shift (<<):**

The left shift operator shifts the bits of its left operand to the left by a specified number of positions. It discards the shifted bits and fills the vacated positions with zeros.

Example:
```
int a = 5;      // binary: 0101
int result = a << 2; // binary: 010100, decimal: 20
System.out.println(result);

Output: 20 (10100)
```

6. **Right Shift (>>):**

The right shift operator shifts the bits of its left operand to the right by a specified number of positions. It discards the shifted bits and fills the vacated positions with the sign bit (the leftmost bit for signed data types).

Example:
```
int a = -10;    // binary: 1111111111111111111111111110100
int result = a >> 2; // binary: 1111111111111111111111111111101, decimal: -3
System.out.println(result);

Output: -3 (111111111111111111111111110)
```

7. **Unsigned Right Shift (>>>):**

The unsigned right shift operator shifts the bits of its left operand to the right by a specified number of positions. It discards the shifted bits and fills the vacated positions with zeros.

Example:
```
int a = -10;    // binary: 1111111111111111111111111110100
int result = a >>> 2; // binary: 0011111111111111111111111111101, decimal: 1073741821
System.out.println(result);

Output: 1073741822(001111111111111111111)
```