̨﷽

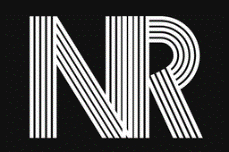
An-Najah National University

Faculty of Engineering and Information Technology

Computer Engineering Department



**Software Graduation Project**

****

**An Najah Rank**

Prepared by:

**-Momen Odeh -Noor Aldeen Abu Shehadeh**

Supervised by:

**Dr. Samer Arandi**

Submitted on:  
**24th, January 2024**

Presented in partial fulfillment of the requirements for Bachelor degree in Computer Engineering.

# Dedication

Dedication to loving memory of our grandmother, our loving parents, family, all the martyrs of Palestine, friends and for everyone who believed and loved us.

# Acknowledgment

*We extend our deepest gratitude and appreciation to the individuals who have played a significant role in our graduation project. Their guidance, support, and unwavering belief in our abilities have been invaluable throughout this journey.*

We would like to **thank our supervisor** **Dr. *Samer Arandi*** a lot for his helpful, kind, patience and taking care of us, and for making everything simple. He was always inspiring and encouraging us to move.

*We would also like to* ***thank all the teachers and teacher’s assistant in the Department of Computer Engineering****, and we feel proud to be students in it, as this helps us to improve our educational level as well as improve our skills.*

# Disclaimer

This report was written by students at the Computer Engineering Department, Faculty of Engineering, An-Najah National University. It has not been altered or corrected, other than editorial corrections, as a result of assessment and it may contain language as well as content errors. The views expressed in it together with any outcomes and recommendations are solely those of the students. An-Najah National University accepts no responsibility or liability for the consequences of this report being used for a purpose other than the purpose for which it was commissioned.

# Table of content

[Dedication 2](#_Toc144160861)

[Acknowledgment 3](#_Toc144160862)

[Disclaimer 4](#_Toc144160863)

[Table of content 5](#_Toc144160864)

[List of Figures 6](#_Toc144160865)

[List of tables 7](#_Toc144160866)

[Abstract 8](#_Toc144160867)

[Chapter 1: Introduction 9](#_Toc144160868)

[ Statement of the problem 9](#_Toc144160869)

[ Objective 10](#_Toc144160870)

[ Scope of the work 11](#_Toc144160871)

[ Importance of the work 12](#_Toc144160872)

[ Organization of the report 13](#_Toc144160873)

[Chapter 2: Theoretical Background and Previous Work 14](#_Toc144160874)

[Chapter 3: Methodology 15](#_Toc144160875)

[ How It Works? **Error! Bookmark not defined.**](#_Toc144160876)

[ The Mechanism of conversion text to braille **Error! Bookmark not defined.**](#_Toc144160877)

[ Components **Error! Bookmark not defined.**](#_Toc144160878)

[ Design **Error! Bookmark not defined.**](#_Toc144160879)

[ Component circuits **Error! Bookmark not defined.**](#_Toc144160880)

[ The Final result of the model **Error! Bookmark not defined.**](#_Toc144160881)

[ Build Web application using React JS **Error! Bookmark not defined.**](#_Toc144160882)

[ Challenges Through design **Error! Bookmark not defined.**](#_Toc144160883)

[ Standards and specifications **Error! Bookmark not defined.**](#_Toc144160884)

[ Constraints and design limitations **Error! Bookmark not defined.**](#_Toc144160885)

[Chapter 4: Result and Analysis 23](#_Toc144160886)

[Chapter 5: Discussion 24](#_Toc144160887)

[Chapter 6: Conclusions and Recommendation 25](#_Toc144160888)

[References 26](#_Toc144160889)

[Appendices 27](#_Toc144160890)

# List of Figures

[Figure 1: Dismantled an old printer **Error! Bookmark not defined.**](#_Toc144157077)

[Figure 2: Paper feeder in paper container **Error! Bookmark not defined.**](#_Toc144157078)

[Figure 3: Build XY coordinate **Error! Bookmark not defined.**](#_Toc144157079)

[Figure 4: Attached the paper roller **Error! Bookmark not defined.**](#_Toc144157080)

[Figure 5: Attached feeder, paper rollers and XY coordinate **Error! Bookmark not defined.**](#_Toc144157081)

[Figure 6: Start working to the final design **Error! Bookmark not defined.**](#_Toc144157082)

[Figure 7: Assemble of the final design **Error! Bookmark not defined.**](#_Toc144157083)

[Figure 8: Stepper motors connection **Error! Bookmark not defined.**](#_Toc144157084)

[Figure 9: DC motors connection **Error! Bookmark not defined.**](#_Toc144157085)

[Figure 10: Solenoid connection **Error! Bookmark not defined.**](#_Toc144157086)

[Figure 11: Buzzer connection **Error! Bookmark not defined.**](#_Toc144157087)

[Figure 12: Servo motor connection **Error! Bookmark not defined.**](#_Toc144157088)

[Figure 13: LCD connection **Error! Bookmark not defined.**](#_Toc144157089)

[Figure 14: Limit switches connection **Error! Bookmark not defined.**](#_Toc144157090)

[Figure 15: Optical sensor connection **Error! Bookmark not defined.**](#_Toc144157091)

[Figure 16: Buck converter connection **Error! Bookmark not defined.**](#_Toc144157092)

[Figure 17: ESP8266 D1 Mini connection **Error! Bookmark not defined.**](#_Toc144157093)

[Figure 18: Top and side view of the model **Error! Bookmark not defined.**](#_Toc144157094)

[Figure 19: Complete model **Error! Bookmark not defined.**](#_Toc144157095)

[Figure 20: Log In page **Error! Bookmark not defined.**](#_Toc144157096)

[Figure 21: Sign Up page **Error! Bookmark not defined.**](#_Toc144157097)

[Figure 22: Printing page **Error! Bookmark not defined.**](#_Toc144157098)

[Figure 23: Start printing **Error! Bookmark not defined.**](#_Toc144157099)

[Figure 24: Finish printing **Error! Bookmark not defined.**](#_Toc144157100)

[Figure 25: Connect driver with inverter **Error! Bookmark not defined.**](#_Toc144157101)

[Figure 26: Connect capacitor with solenoid **Error! Bookmark not defined.**](#_Toc144157102)

# List of tables

[Table 1: Hardware components **Error! Bookmark not defined.**](#_Toc144160847)

[Table 2: Mechanical components **Error! Bookmark not defined.**](#_Toc144160848)

# Abstract

One of the most important skills for any programmer is problem-solving skills, and there are many websites that can be used to train these skills, such as HackerRank, Codeforces, LeetCode, etc.

At An-Najah National University, professor always strive to improve students' problem- solving skills in many subjects such as computer programming, data structures, algorithms, and object-oriented programming by assigning problem-solving assignments and quizzes using problem-solving websites. However, they face several challenges in using these websites, such as difficulty tracking student submissions, an inability to directly identify code similarities among students' submissions, and the inability to manually mark incorrect answers.

We built this project by creating a web application with React JS as the frontend and Flask Python as the backend. We used Docker to containerize the application, allowing easy deployment on the cloud or any local server. Additionally, we leveraged several services from Amazon Web Services (AWS), including S3 for storage, RDS for the MySQL database engine, and EC2 for deploying the web application.

# Chapter 1: Introduction

## Statement of the problem

The problem-solving skills are one of the most important skills in the workplace, so An-Najah National University strives to improve these skills in our students by incorporating problem-solving tasks into many courses using external problem-solving websites. However, these websites lack essential features that would simplify the problem-solving process and make solution grading more efficient. This emphasizes the increasing importance of a web application to address all these challenges.

One of the primary challenges lies in the difficulty of efficiently tracking and managing student submissions. This hinders the seamless monitoring of individual progress and the timely assessment of assignments. Additionally, there is a limitation in directly identifying code similarities among the submissions, making it challenging to address potential collaboration or plagiarism issues effectively.

Another notable challenge is the absence of a streamlined mechanism for manually marking incorrect answers. This deficiency impedes the ability of professors to provide targeted feedback, hindering the learning process for students.

## Objective

The purpose of our work is to create a web application for problem-solving that is easy to use for both students and professors. We aim to achieve this by incorporating new features not available in other problem-solving applications. The objectives of our work are as follow:

* Registration and login for both students, professors and admin on the web application.
* Professors can create new courses and enroll students in them by simply uploading the excel file exported from any zajel course.
* Professors can add programming assignments to their courses. For each such assignment the professor provides a set of input test cases and the expected correct output. The system will automatically correct the assignments based on the output test cases. Each assignment has a starting and ending date, during which it will be available to the students.
* Professors can view a list of students who have submitted assignments, their grades, and the similarity of their submissions. They can also review the submissions and optionally manually mark last submission that was found incorrect by the system.
* The professor can also track the progress of the student submission, i.e. they can see the changes from the first version the students submit to the last (hopefully) correct answer.
* Students can access their homepage on the system which shows information about the assignments and quizzes in current or previous courses. The student can start solving the assignments/quizzes assigned to them by writing code in their preferred programming language, such as C, C++, Java, Python, JavaScript, or REGX directly in the browser. They can then run the code to check if it passes or fails test cases.
* The student can also view the status for each assignment/quiz, their score and general performance.
* User receive notification when a related event occurs.
* Any user can make chatting with other user.

## Scope of the work

* **Frontend using React JS:** We developed the frontend using React JS, building the user interface with the React JS library, utilizing React Bootstrap as the UI kit, and incorporating React-JSS for styled components.
* **Backend using Flask python:** We developed the backend using Flask python microservices framework.
* **Database using MySQL:** We chose the MySQL database because our data is relational. Subsequently, we generated the tables using MySQL Workbench.
* **Amazon Web Services (AWS):** We utilize various services from AWS, using the S3 service for storing files and images, the RDS service for the MySQL database engine, and the EC2 service for deploying the web application.
* **DevOps:** We generated a portable copy of our project that can be easily deployed on any device using Docker and Docker Compose technologies.
* **Testing:** After building our project, we conducted manual tests to ensure that everything worked correctly.

## Importance of the work

The An-Najah Rank web application has many features that enhance usability and includes new functionalities. Here are the reasons that explain why this web application is important:

* **Easy to use:** The web application is user-friendly for all users, including admin, professors, and students. And that appear in simplicity of user interface.
* **Check plagiarism:** We have added a 'calculate similarity' feature that can check the similarity between student code submissions.
* **Show all submissions:** We can easily to traversing student submissions by viewing all last submissions of students in one place and can traverse all submissions on any student easily.
* **Manual Marking:** We have added a manual marking feature that allows professor to remarking the last submission of any student.
* **Flexibility of test cases:** The professor can adjust the final grade of challenge by adding new test case that will automatically run the new test case on the last submission code and adjust the final grade based on all results.
* **This web application is implemented specifically for educational use:** We have customized many features for this purpose, such as limiting the programming languages that can be used to solve the problem. Professors can easily add students by uploading an Excel sheet.

## Organization of the report

The report is structured in a logical and systematic manner to effectively present the information related to the project. The organization of the report is as follows:

* **Introduction:** This section provides an overview of the project, highlighting the problem statement, objectives, and the importance of the work. It sets the context for the rest of the report.
* **Theoretical Background and Previous Work:** In Chapter 2, It presents a comprehensive review of existing research, studies, and relevant literature related to Braille Printers, automation techniques, and similar projects. This section helps establish the project's context and highlights any gaps in the existing knowledge.
* **Methodology:** Chapter 3 explains the materials and methods used throughout the project. It provides a detailed description of the experimental setup, the Web application development process. The chapter outlines the steps taken to achieve the project objectives.
* **Results and Analysis:** Chapter 4 presents the results obtained from the project. It includes the outcomes of the Printing process using the An Najah Rank web application, as well as any relevant data or measurements. The results are analyzed and interpreted to draw meaningful conclusions.
* **Discussion:** Chapter 5 focuses on the discussion of the results. It provides a comprehensive analysis of the findings, highlighting the features, benefits, and limitations of the An Najah Rank web application. The chapter also addresses any challenges faced during the project and offers recommendations for future improvements.
* **Conclusion and Recommendation:** chapter6 concludes report by summarizing the key findings, reiterating the significance of the work, and highlighting its potential impact. It may also include a reflection on the overall project experience and suggestions for further research.
* **References:** A list of all the references cited throughout the report is provided in the References section, following the conclusion.

# Chapter 2: Theoretical Background and Previous Work

These days, there are many problem-solving web applications, such as LeetCode, CodeForces, and HackerRank. However, these web applications are not completely suitable for educational purposes. Therefore, we built a problem-solving web application that combines solving problems for students and adds the educational features needed for professors, making the process more straightforward.

# Chapter 3: Methodology

In our project, we diligently adhere to the Software Development Life Cycle (SDLC), a systematic approach that enables cost-effective and time-efficient software development. SDLC guides our development teams through essential stages such as planning, design, development, testing, deployment, and maintenance. This structured process not only aims to design and build high-quality software but also minimizes project risks through forward planning. By following SDLC, we ensure that the software meets customer expectations during production and beyond, contributing to the overall success and reliability of our projects.

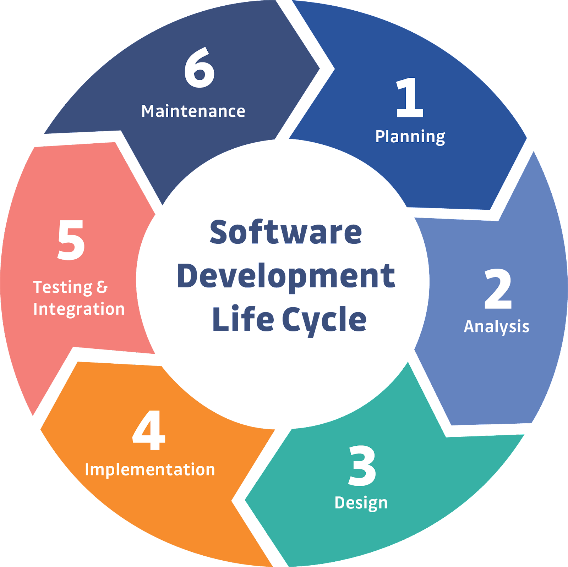


Figure 1: Software Development Life Cycle

In our project, we have embraced the Agile methodology as the guiding framework for our software development process. Agile is a dynamic and iterative approach that prioritizes flexibility, collaboration, and customer satisfaction. Unlike traditional linear models, Agile promotes adaptability to changing requirements and a continuous feedback loop, allowing us to respond promptly to evolving project needs.



Figure 2: Agile Methodology

# 3.1 Planning:

We met with our client, Dr. Samer Arandi, to discuss the project features and decide which ones will be implemented. During our meeting, we explored various problem-solving websites to gain insights and ideas for the project.

Our collaboration extended beyond the existing features as we explored new functionalities to enhance the project. This discussion not only provided a clearer vision for the project but also facilitated the identification of potential innovative features to meet both current and future user needs.

# 3.2 Analysis:

In the initial phase of our software project, thorough analysis was conducted to gather and document project requirements through stakeholder engagement and user feedback sessions. This process involved crafting user stories to delineate specific functionalities and envisioning the system's architecture through Unified Modeling Language (UML) diagrams.

## 3.2.1 UML Diagram:

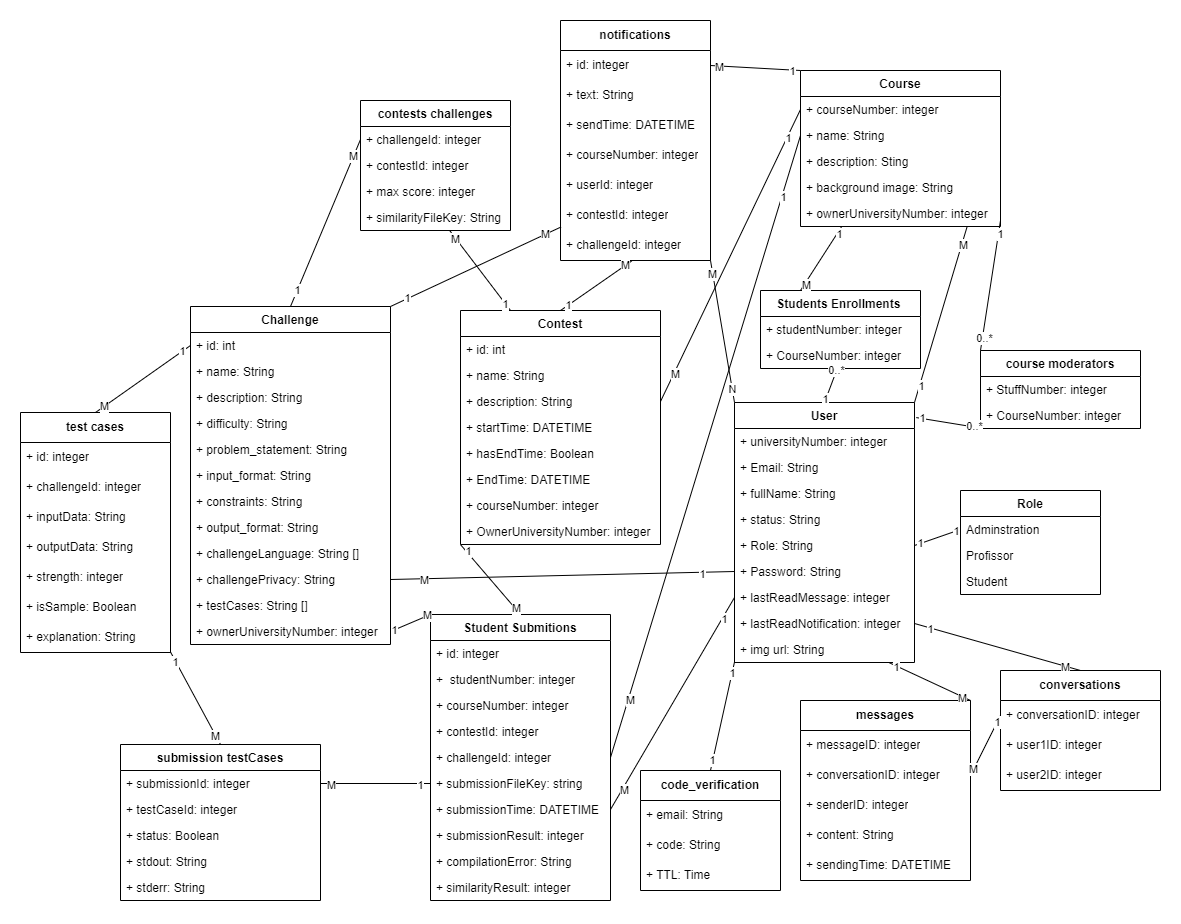


Figure 3: UML Diagram

## 3.2.2 User Stories:

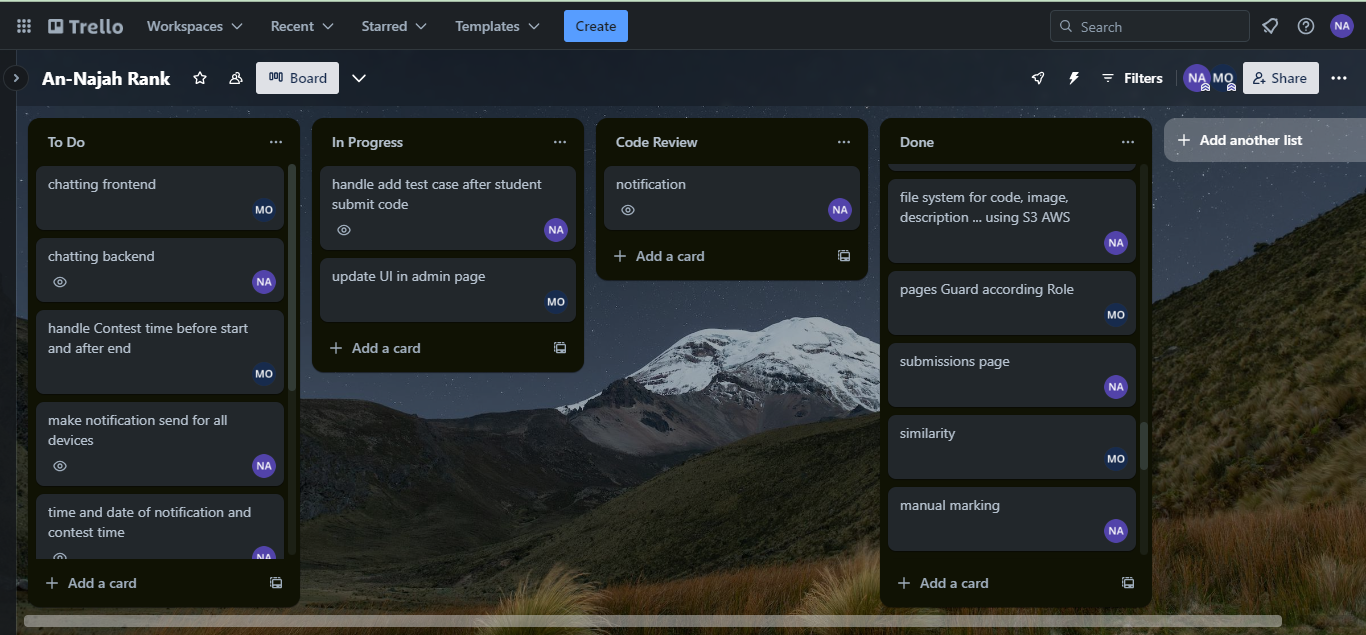


Figure 4: User Stories

# 3.3 Design:

## 3.3.1 Tools:

#### **3.3.1.1 Frontend tools:**

##### **3.3.1.1.1 React JS:**

React is a declarative, efficient, and flexible JavaScript library for building user interfaces. It makes it easy to compose complex UIs from small and isolated pieces of code called components.

In our project we used ReactJS as the front-end technology due to the easiness of learning, rich set of user-interface, community support, and the fast development of software. In addition, it offers the capability to reuse already built components.



Figure 5: React

##### **3.3.1.1.2 React Bootstrap:**

This UI kit contains many ready components that can be used directly with some customization for style. Additionally, this UI kit provides components that can make the design responsive easily.



Figure 6: React Bootstrap

##### **3.3.1.1.3 React JSS:**

Is a library that enables styling React components using JavaScript. Providing powerful features such as:

- Dynamic Theming - allows context-based theme propagation and runtime updates.

- Function values and rules are updated automatically with any data that passed as probs.



Figure 7: React JSS

#### **3.3.1.2 Backend tools:**

##### **3.3.1.2.1 Flask python:**

Flask is a lightweight and user-friendly Python web framework that streamlines backend development. While originally designed for simplicity, Flask proves versatile for building microservices. It provides a simple way to create and deploy dynamic web applications; it enables developers to focus on the application logic rather than worrying about the underlying infrastructure. Moreover, it offers a great deal of freedom and control over application development. Its integration with Python libraries and technologies makes it easy to integrate with a wide variety of software development tools and solutions.



Figure 8: Flask python

##### **3.3.1.2.2 Pandas:**

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.



Figure 9: Pandas

##### **3.3.1.2.3 SocketIO:**

Flask-SocketIO is an extension for Flask that facilitates low-latency, bidirectional communication between the server and clients using WebSockets. It allows real-time, interactive features to be implemented in Flask applications by enabling seamless communication between the server and connected clients.



Figure 10: SocketIO

#### **3.3.1.3 DevOps tools:**

##### **3.3.1.3.1 GitHub:**

Git is open-source version control software, used for managing and tracking file revisions. You can use Git with any file type, but it’s most often used for tracking code files.

GitHub is an online software development platform. It's used for storing, tracking, and collaborating on software projects.



Figure 11: GitHub

##### **3.3.1.3.2 Trello:**

Trello is the visual tool that empowers your team to manage any type of project, workflow, or task tracking. Add files, checklists, or even automation: Customize it all for how your team works best.



Figure 12: Trello

##### **3.3.1.3.3 Docker:**

Docker is a software platform that uses OS-level virtualization to deliver software in packages called containers. It allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that contain everything the software needs to run, including libraries, system tools, code, and runtime. By using Docker, you can quickly deploy and scale applications into any environment and be confident that your code will run.



Figure 13: Docker

##### **3.3.1.3.4 Docker Compose:**

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure the application's services. Then, with a single command, you can create and start all the services from your configuration.



Figure 14: Docker Compose

#### **3.3.1.4 AWS Cloud Formation:**

AWS CloudFormation is Amazon Web Services’ (AWS) native IaC tool. It enables you to define infrastructure resources using YAML or JSON templates, ensuring automation and consistent deployments in the AWS environment.



Figure 15: AWS CloudFormation

##### **3.3.1.4.1 AWS EC2:**

Amazon Elastic Compute Cloud (Amazon EC2) provides on-demand, scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 reduces hardware costs so you can develop and deploy applications faster.

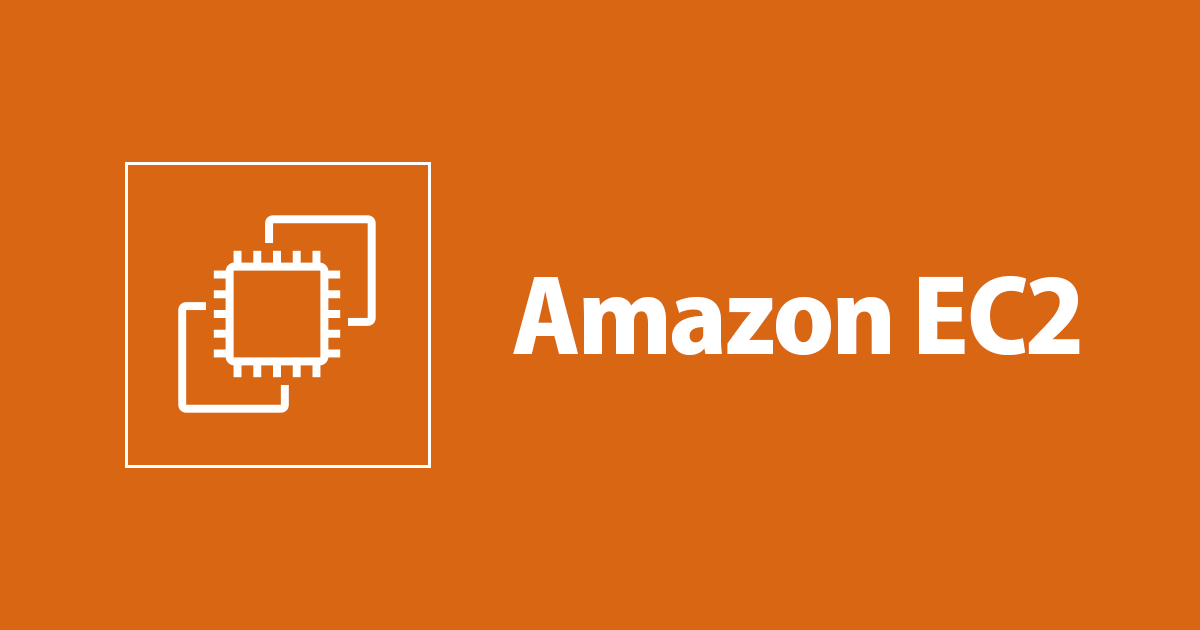


Figure 16: AWS EC2

##### **3.3.1.4.2 AWS S3:**

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance.



Figure 17: AWS S3

##### **3.3.1.4.3 AWS RDS:**

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the AWS Cloud.



Figure 18: AWS RDS

#### **3.3.1.5 Development tools:**

##### **3.3.1.5.1 VS Code:**

Used for React development.

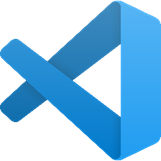


Figure 19: VS Code

##### **3.3.1.5.2 pycharm:**

Used for Flask development.



Figure 20: pycharm

##### **3.3.1.5.3 MySQL Workbench:**

Used for building and monitoring database.



Figure 21: MySQL Workbench

##### **3.3.1.5.4 Postman:**

Used for test backend APIs.



Figure 22: Postman

##### **3.3.1.5.5 Docker Desktop:**

Used for managing images and containers.



Figure 23: Docker Desktop

##### **3.3.1.5.6 Draw io:**

Used for design UML diagram.



Figure 24: Draw io

## 3.3.2 Architecture:

#### **3.3.2.1 Architectural Style:**

We used **RESTful** architectural style, which is a design approach for networked applications prioritizing simplicity, scalability, and loose coupling. It utilizes a stateless client-server model with principles such as statelessness, a uniform interface, and resource-based interactions. Key advantages encompass simplicity, scalability, and a consistent interface.

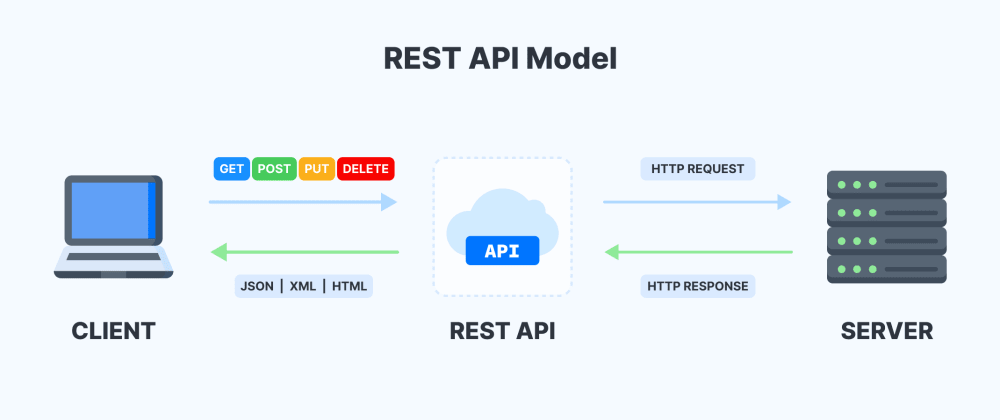


Figure 28: Restful Architectural Style

To send requests from the frontend to the backend, **Axios**, a popular JavaScript library, is commonly used in React applications. Axios simplifies the process of making asynchronous HTTP requests to external resources, particularly APIs. It is favored for its simplicity, flexibility, and notable features, including automatic JSON data transformation in responses.

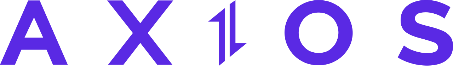


Figure 29: AXIOS library

#### **3.3.2.2 Architectural Pattern:**

We used Microservices architectural pattern, which is particularly beneficial for large and complex applications where different functionalities can be developed and maintained independently.

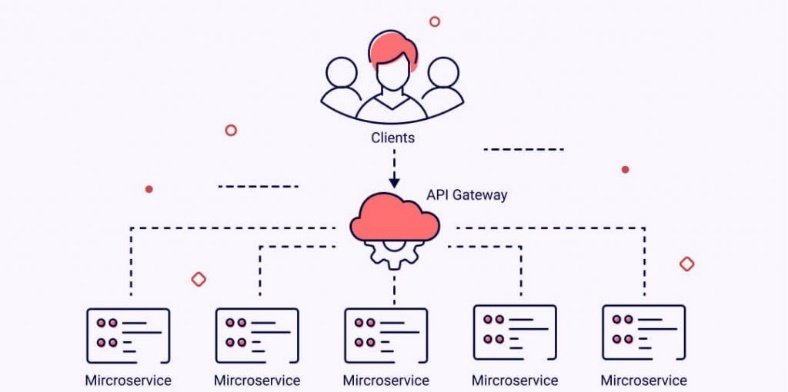


Figure 30: Microservice Architectural pattern

#### **3.3.2.3 Project Structure:**

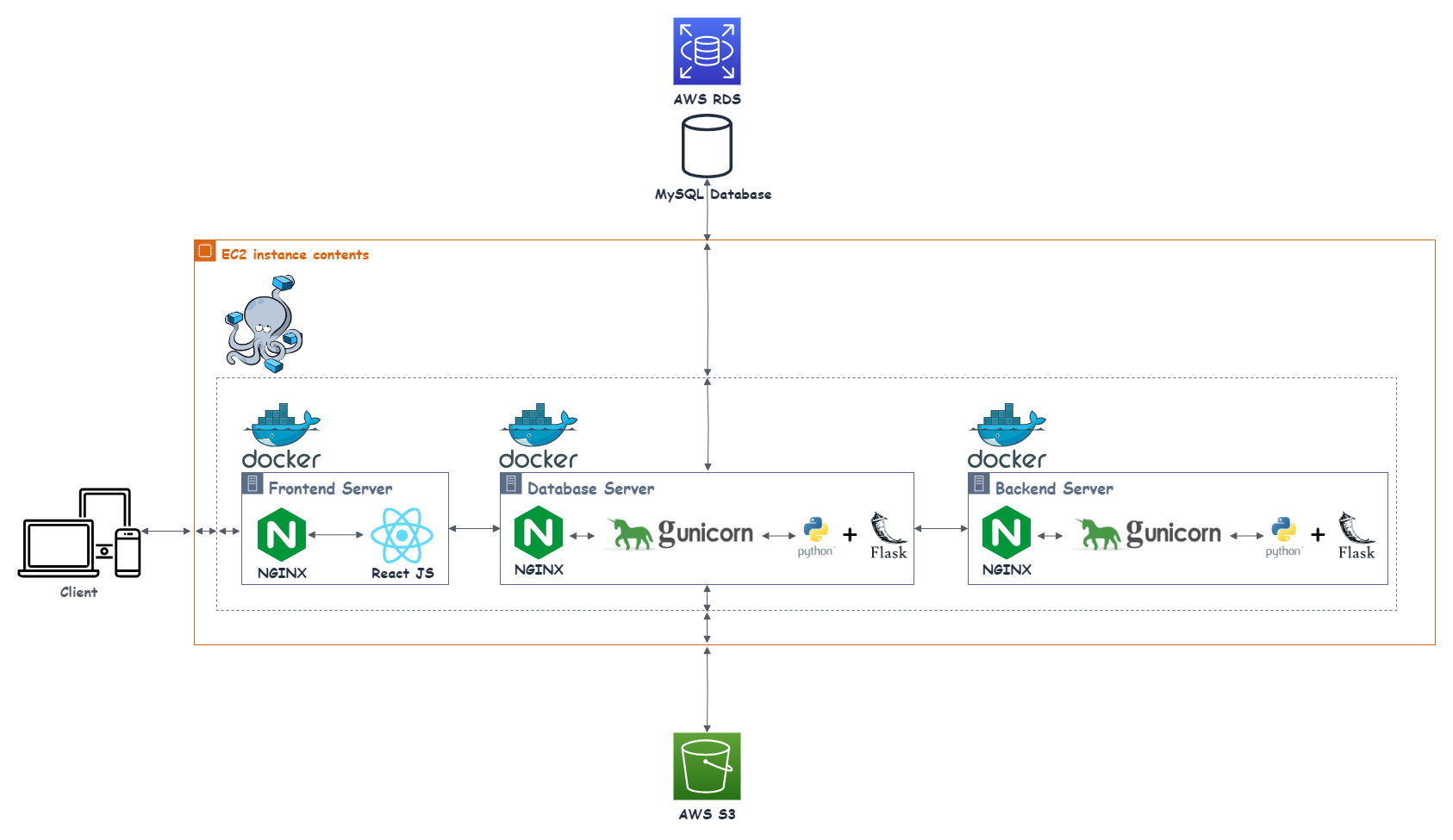
****

Figure 31: Project Structure

We divide the project into 3 containers:

1. Frontend container: Handles client requests and returns the UI to the client.
2. Database container: Manages requests from the Frontend container. If the request is related to code operations, it passes the request to the Backend container and returns the response to the Frontend container.
3. Backend container: Handles code operation requests, such as compiling and running code.

In the deployment structure for the frontend, Nginx serves as a crucial web server, playing a key role in handling the dynamic runtime of the React application.

In the database and backend deployment structure optimized for APIs, Flask serves as the foundational framework, responsible for handling application logic and dynamically generating content. Gunicorn is employed as the WSGI server, efficiently managing communication and concurrent requests through multiple worker processes. While Nginx, traditionally recognized as a reverse proxy for web applications, is considered optional in this API-centric setup, it remains a valuable component for potential load balancing and additional security measures. The refined architecture emphasizes a modular separation of concerns, with Flask managing API routes, Gunicorn overseeing WSGI interactions, and Nginx, when utilized, contributing to load balancing and potential security enhancements. This streamlined structure establishes a dependable, scalable, and secure foundation specifically designed for deploying a Flask backend focused on API functionalities in production environments, combining the strengths of Flask, Gunicorn, and Nginx for optimal performance and security.

## 3.3.3 Security:

#### **3.3.3.1 Authentication:**

To use the web application, you must have an account. To obtain one, you need to register on the system and confirm your registration by entering the valid verification code received via email. When a user logs into the system, we authenticate their information. If the authentication is successful, we generate a token and return it to the frontend.

#### **3.3.3.2 Autherization:**

After logging in, each request to the backend should include a token. In the backend, the system first checks the validity of the token. If the token is valid, it is passed to the API; otherwise, an unauthorized response is returned. Upon receiving a request, the API checks the user's role, which is extracted from the token. If the user has the necessary access rights to the API, the request is processed; otherwise, an unauthorized response is returned.

#### **3.3.3.3 CORS policies:**

In the backend, we enable the CORS policy for the frontend address, so any received request from another address will be rejected.

#### **3.3.3.4 : Library used:**

|  |  |  |
| --- | --- | --- |
| Figure 25: Flask Mai | Figure 26: pyJWT | Figure 27: Flask-Cors |

# 3.4 Implementation:

## 3.4.1 Sign Up:

# Chapter 4: Result and Analysis

Our Braille printer produces printed Braille papers of good quality that adhere to the rules of the Braille language, ensuring correct distances between dots within each character and between characters. It's important to note that achieving absolute distance accuracy between dots can be challenging due to various factors. For instance, the solenoid's precision may not be optimal and on the other hand, handling the paper presents difficulties due to its extremely small thickness and sensitivity to external factors. Despite these challenges, the overall output results are good.

# Chapter 5: Discussion

Individuals with visual impairments encounter numerous hurdles when seeking to print text in Braille. Among these challenges, a prominent one is the considerable expense associated with purchasing Braille printers, rendering them inaccessible to a substantial portion of the population that could greatly benefit from their utility. Furthermore, the availability of Braille printers presents an additional obstacle.

We've developed an affordable Braille printer that's easily accessible whenever required. This printer is made up of readily available basic parts found in the market. This creative solution enables people with visual impairments to conveniently obtain the resources they need.

# Chapter 6: Conclusions and Recommendation

We have constructed a Braille printer with a low cost that is readily available when needed. This printer is composed of simple components that are readily available on the market. This innovative solution allows individuals with visual impairments to easily access to the resources.

After studying the project during the design stage, we chose the most efficient approach to its implementation. Our goal was to achieve high performance based on the hardware components and project design, while also ensuring cost-effectiveness and high availability.

We gained valuable insights into effectively working with hardware components by thoroughly studying datasheets and component documentation. This comprehensive approach allowed us to understand the intricate details of each component's functionality and integration. In addition to mastering the technical aspects, we honed our skills in designing a system that ensures stability and reliability.

By delving into the datasheets, we were able to get information about pin configurations, electrical characteristics, and communication protocols.

Furthermore, our journey involved not only understanding individual components but also orchestrating their harmonious collaboration. By thoughtfully addressing these aspects, we were able to create a system that remains good and dependable under varying conditions.

**Future Works :**

1. Make the Web application accept formats such as .pdf, .docx, …etc.
2. Make the printer accept different sizes of papers such as A5, A3.
3. Add the scanner that scan the text from papers then send it to printer to print in in Braille.

# References

* **Pharmaceutical Braille :** The Braille representation. Available:

<https://www.pharmabraille.com/pharmaceutical-braille/the-braille-alphabet/>.

* **WeCapable** **Braille translator**. Available:<https://wecapable.com/braille-translator/english-to-braille-converter/> .
* **AccelStepper library for Arduino**. Available: <https://www.airspayce.com/mikem/arduino/AccelStepper/classAccelStepper.html> .
* **LiquidCrystal.** Available:<https://www.arduino.cc/reference/en/libraries/liquidcrystal/> .
* **Servo**. Available: <https://www.arduino.cc/reference/en/libraries/servo/> .
* **SoftwareSerial Library**. Available: <https://docs.arduino.cc/learn/built-in-libraries/software-serial> .
* **ESP8266WiFi library**. Available: <https://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/readme.html> .
* **HTTPClient Class Reference**. Available : <https://links2004.github.io/Arduino/dd/d8d/class_h_t_t_p_client.html> .
* **WiFi**. Available: <https://www.arduino.cc/reference/en/libraries/wifi/> .
* **FirebaseESP8266**. Available: <https://github.com/mobizt/Firebase-ESP8266> .
* **React JS Documentation**. Available: <https://react.dev/> .
* **Firebase Documentation**. Available : <https://firebase.google.com/docs> .

# Appendices

* **Arduino Mega code :** <https://github.com/Momen-Odeh/Braille-Printer-GP2/tree/main/GP2> **.**
* **ESP8266 D1 mini code :** <https://github.com/Momen-Odeh/Braille-Printer-GP2/tree/main/WIFI_ESP86266> .
* **Web application code :** <https://github.com/Momen-Odeh/Braille-Printer-Web-Page> .