Application Architecture Document

Application Name: recruitment portal

Date: 20/09/2023

Prepared by: DevOps Unit

Table of Contents

1. Introduction

- Purpose of the Document

- Scope

- Audience

1. System Overview

- Brief description of the application- careers portal that manages applications and hiring candidates

- High-level goals and objectives

- Key features and functionalities- recieves and processes applications for candidates

1. Architecture Overview

- High-level architecture diagram

- Components and their interactions

- Technologies and frameworks used- nextjs, dotnet core, mssql

1. Architectural Patterns

- Explanation of the architectural pattern(s) used (e.g., MVC, Microservices, Serverless)- microservices

- Justification for selecting these patterns- suits the application needs

1. System Components

- Detailed description of each major component:

- Component name

- Purpose and responsibilities

- Interfaces and APIs- CREATE\_MEETING="http://localhost:5048/api/Candidate/meeting"

CREATE\_COMMENT="http://localhost:5048/api/Candidate/comment"

GET\_JOB\_ROLES="http://localhost:5048/roles/Role/all"

GET\_CANDIDATE\_BY\_ID="http://localhost:5048/api/Candidate/id"

MOVE\_TO\_NEXT\_STAGE="http://localhost:5048/stage"

GET\_JOB\_DESCRIPTION="http://localhost:5048/roles/Role/getJobDescription"

GET\_SKILLS="http://localhost:5048/api/Candidate/skills"

FLAG\_CANDIDATE="http://localhost:5048/api/Candidate/flag"

CANCEL\_APPLICATION="http://localhost:5048/api/Candidate/cancel"

GET\_RESUME="http://localhost:5048/api/Candidate/resume"

GET\_ACTIVE\_ROLES="http://localhost:5048/roles/Role"

GET\_CANDIDATE\_BY\_STAGE="http://localhost:5048/api/Candidate/stage"

HIRE\_CANDIDATE="http://localhost:5048/api/Candidate/hire"

GET\_ALL\_JOB\_ROLES="http://localhost:5048/roles/Role/getJobRoles"

SEARCH\_JOB\_ROLES="http://localhost:5048/roles/Role/search"

GET\_STATUS="http://localhost:5048/status"

CANCEL\_APPLICATION="http://localhost:5048/api/Candidate/cancel"

SIGN\_USER\_IN="http://localhost:5048/api/Candidate/signin"

GET\_MEETINGS="http://localhost:5048/api/Candidate/meetings"

GET\_COMMENTS\_BY\_ID="http://localhost:5048/api/Candidate/getcomments"

FLAG\_APPLICATION="http://localhost:5048/api/Candidate/flag"

ADMIN\_AUTH="http://localhost:5048/api/Candidate/admin/auth"

VALIDATE\_EMAIL="http://localhost:5048/api/Candidate/validate"

GET\_METRICS="http://localhost:5048/api/Candidate/metrics"

GET\_APPLICATIONS\_BY\_JOB="http://localhost:5048/api/Candidate/role"

CREATE\_APPLICATION="http://localhost:5048/api/Candidate/create"

CREATE\_NEW\_USER="http://localhost:5048/api/Candidate/user"

SIGN\_USER\_IN="http://localhost:5048/api/Candidate/signin"

GET\_STATUS="http://localhost:5048/status"

CREATE\_NEW\_JOB\_ROLE="http://localhost:5048/roles/Role"

SEND\_PASS\_RESET\_MAIL="http://localhost:5048/api/Candidate/mail\_reset\_options"

RESET\_PASSWORD="http://localhost:5048/api/Candidate/reset\_password"

GET\_APPLICATIONS\_BY\_FLAG="http://localhost:5048/api/Candidate/flag/candidates"

GET\_APPLICANTS\_BY\_FLAG="http://localhost:5048/api/Candidate/flag/candidates"

CHANGE\_STATUS="http://localhost:5048/roles/Role/status"

GET\_RESUME="http://localhost:5048/api/Candidate/resume"

- Data flow and dependencies

1. Data Architecture

- Data models and schemas

- Database systems and technologies used

- Data storage and retrieval strategies

- Data synchronization and replication (if applicable)

1. Technology Stack

- Programming languages- typescript, html, css, c#, sql,

- Frameworks and libraries- dotnet core, nextjs,

- Databases and data storage solutions- mssql

- Third-party services and APIs

- Development tools and IDEs- visual studio code

1. Deployment Architecture

- Server infrastructure (physical or cloud-based)- physical

- Scalability and load balancing strategies

- Security measures

- Disaster recovery and backup solutions

1. Communication and Integration

- Communication protocols (e.g., REST, WebSocket)- REST

- External services and APIs integrated- zoom api

- Messaging systems (e.g., Kafka, RabbitMQ)

1. Security

- Authentication and authorization mechanisms- uses dotnet cores policies for authentication and authorization

- Data encryption (in transit and at rest)- uses AES encryption for encrypting data in transit

- Security best practices and compliance with industry standards

1. Performance and Scalability

- Performance optimization techniques

- Scalability strategies (horizontal and vertical)

- Resource utilization monitoring

1. Testing and Quality Assurance

- Testing methodologies (e.g., unit testing, integration testing)

- Continuous integration and deployment (CI/CD) pipelines

- Quality assurance processes

1. Monitoring and Logging

- Tools and technologies for monitoring application health

- Logging and error tracking mechanisms- errors are writen to a file and saved for later

- Alerts and notifications setup

1. Documentation and Knowledge Sharing

- Code documentation standards

- Knowledge transfer processes for onboarding new team members

1. Maintenance and Support

- Post-launch maintenance plan

- Bug tracking and issue resolution

- Version control and release management

1. Appendices

- Additional diagrams (sequence diagrams, ER diagrams)

- Glossary of technical terms and acronyms

- References and external resources

Conclusion

Summarize the key points from the document and reiterate its importance in guiding the development, maintenance, and scaling of the application.

Revision History

Maintain a record of changes made to the document over time.

Approvals

Head, DevOps

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Head, Quality Assurance

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Head, IT Solution and Innovation

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_