SPOT-A-DOC

PLATFORM ECONOMY: WEB-BASED CLINIC-PATIENT MATCHING SYSTEM

MAK HO YIN

3035693337
BACHELOR OF ENGINEERING (COMPUTER SCIENCE)
FINAL PROJECT REPORT

Supervisor: Dr. Tsz Hon Yuen, John
Date of Submission: 17-04-2023
Abstract

This paper proposes, designs, and implements an electronic commerce platform for Hong Kong’s commercial clinic industry to improve the doctor’s appointment reservation process and to make medical certifications more trustworthy. This is in view of the obsolesce in the telephone-oriented process of doctor’s appointment reservation and the matter of forgery and misuse of paper-based medical certificates in the commercial medical sector in Hong Kong. The realization of the project goals involves the use of modern web technologies and applied cryptography, as well as well-established software engineering frameworks and DevOps practices like version control and CI/CD. In particular, the development of the platform uses the MERN stack which are JavaScript-based technologies that involve concepts like non-relational databases. Topics such as electronic commerce platform considerations, digital signature implementation, responsive web design etc. are discussed. The project is successful in that the project objectives for the development of the e-commerce platform with the proposed features has been realized.
Acknowledgment

I would like to express my sincerest gratitude to my project supervisor, Professor John Yuen, for his invaluable insights and mentorship throughout the development of the project deliverables. He has selflessly shared his expertise in e-commerce technologies and cryptography that are instrumental to the success of my project.

I would also like to thank my colleagues at the IT company where I interned for their precious guidance and support. They provided invaluable insights on how the project team could apply industry best practices to make our final deliverable production-ready, and suggested ways to tackle various technical obstacles.

It is also my great privilege to work with my dear teammate, Miss Winky Wong, who has contributed tremendously to key aspects of the deliverables, especially the design and implementation of the product’s user interface.

I also wish to thank Mr. John Ng, my course-mate who is already an experienced cloud practitioner, for provisioning advice regarding the measurement of web performance metrics.

All in all, I feel immense gratitude for the opportunity to work with such an esteemed group. Their mentorship, guidance and support have been instrumental to my learning and growth over the course of this project.
# Table of Contents

Abstract ....................................................................................................................... 2  

Acknowledgment ........................................................................................................ 3  

Table of Contents ........................................................................................................ 4  

List of Figures ............................................................................................................... 6  

List of Tables ............................................................................................................... 8  

1. Introduction .............................................................................................................. 9  
   1.1 Project Contributions ......................................................................................... 9  
   1.2 Background and Literature Review .................................................................. 10  
      1.2.1 Modern E-Commerce with Two-sided Platforms ................................... 10  
      1.2.2 E-Commerce in the Commercial Clinic Industry .................................... 10  
      1.2.3 Use of Digitally Signed Medical Certificates ........................................ 12  
      1.2.4 The Two-sided Platform Solution for Hong Kong .................................. 13  
   1.3 Objectives .......................................................................................................... 14  

2. Methodology ............................................................................................................ 15  
   2.1 Aspects in Electronic Commerce ..................................................................... 15  
      2.1.1 Personalization ......................................................................................... 15  
      2.1.2 Revenue Models ..................................................................................... 16  
   2.2 Medical Certification using Digital Signatures .................................................. 17  
   2.3 Software Architecture and Design .................................................................... 20  
      2.3.1 Adoption of the Three-Tier Architecture ............................................... 21  
      2.3.2 Adoption of the MERN Stack .................................................................. 22  
      2.3.3 Adoption of Cloud Services .................................................................... 23  
      2.3.4 Choice of Cloud Providers ...................................................................... 24  
      2.3.5 Software Design Considerations ............................................................. 25  
      2.3.6 Security Considerations .......................................................................... 26  
   2.4 UI/UX Considerations ......................................................................................... 26  
   2.5 Data Modelling in NoSQL .................................................................................. 27  
   2.6 Software Engineering Methodologies ............................................................... 29  
      2.6.1 Adoption of Agile Development Methodology ........................................ 29  
      2.6.2 Version Control and Collaboration ........................................................... 30  
      2.6.3 Continuous Testing – Unit-testing on Postman ......................................... 31  
      2.6.4 Continuous Integration and Deployment (CI/CD) ...................................... 33  

3. Project Results ......................................................................................................... 34  
   3.1 Elaboration on Completed Features .................................................................. 34  
      3.1.1 Main User Interface (UI) Design ............................................................... 34  
      3.1.2 User Accounts ......................................................................................... 37  
      3.1.3 Clinic/Doctor Listings and Reviews ............................................................ 41  
      3.1.4 Appointment Management ...................................................................... 47  
      3.1.5 Health Articles ........................................................................................ 55  
      3.1.6 Admin Panel ............................................................................................ 56  
   3.2 Software Implementation ...................................................................................... 59  
      3.2.1 Domain Name Service Configurations .................................................... 59  
      3.2.2 Frontend Implementation ........................................................................ 60  
      3.2.3 Backend Implementation ......................................................................... 62  
      3.2.4 DevOps and Testing ................................................................................ 67  
   3.3 Difficulties Encountered ...................................................................................... 68
4. Conclusion........................................................................................................................................69
   4.1 Project Summary............................................................................................................................69
   4.2 Limitations ....................................................................................................................................

References ............................................................................................................................................. V

Appendix I – UX Wireframe Designs ..................................................................................................... VIII
List of Figures

Figure 1: The page showing the list of doctors under the General Practice category. As shown in the figure, the only means for a user to approach the doctor would be to make a phone call or walk-in. .......................................................... 11

Figure 2: The SeeDoctor platform in Hong Kong. Although clinics are able to upload themselves onto the platform, there is no online reservation functionalities provided. Information is unidirectional.......................................................... 12

Figure 3: The Medical Certificate Verification System launched by the Hospital Authority in February 2023. .............. 13

Figure 4: RSA Key generation flow during doctor registration using the doctor’s supplied passcode. .......................................................... 19

Figure 5: Process of signing a piece of appointment data. DPW is input by doctor and checked against database’s record for identity verification. It is then used for regenerating the key pair to obtain the private key for signing. .......................................................... 19

Figure 6: Process of verifying a signature for an appointment. The SK is not involved in this process. .......................................................... 20

Figure 7: An illustration of the three-tier-architecture. A more complete picture will be delivered in section 2.3.3.......................................................... 21

Figure 8: A diagram in UML fashion depicting how cloud computing, the MERN stack, and the three-tiers are architected. Another diagram at a lower level regarding software design will be presented in section 2.3.5.......................................................... 24

Figure 9: A conceptual illustration to the inner design of the architectural tier.......................................................... 25

Figure 10: An illustration of relationships between our MongoDB collections. The Doctor collection is identified jointly by Clinic’s _id field and its own full_name field, which can be thought of as a weak entity in RDBMSs. .......................................................... 28

Figure 11: An example unit-test prototype for illustration. This unit-test contains the set-up and tear down processes with API calls, and within them are specific features to be tested. ............... 32

Figure 12: Test assertions can be written using JavaScript in Postman. Environment variables can also be set to assist in test case construction. .......................................................... 32

Figure 13: The CI/CD workflow for the backend development of our B2C platform. PR stands for “Pull Request” .......................................................... 33

Figure 14: Home Page of Spot-A-Doc.......................................................... 35

Figure 15: The Popular Clinics panel on the Home Page of Spot-A-Doc .......................................................... 35

Figure 16: The Popular Doctors panel on the Home Page of Spot-A-Doc .......................................................... 36

Figure 17: The Health Articles panel on the Home Page of Spot-A-Doc .......................................................... 36

Figure 18: Responsive Design manifested by implementing a side navigation panel for smartphone users.......................................................... 37

Figure 19: Registration page for Client users.......................................................... 38

Figure 20: Registration page for Clinic users.......................................................... 38

Figure 21: The access_token and refresh_token stored in local storage after authentication.......................................................... 39

Figure 22: Login Page of Spot-A-Doc.......................................................... 39

Figure 23: The Check Registration Status page for clinic users with not yet approved registration requests to check request status. .......................................................... 40

Figure 24: Page where clinic users having their requests not yet approved (so that they cannot yet login normally) can view the status of their request here. .......................................................... 41

Figure 25: The Doctors Listing page with entries showing the doctors that are available on the platform.......................................................... 41

Figure 26: The Clinics Listing page with entries showing the doctors that are available on the platform.......................................................... 42

Figure 27: The District attribute is selected for filtering. A separate field selector pops up to the right and allows users to choose from the available options, such as "Central and Western" in this case. The Filter button to the right shall be clicked next.......................................................... 44
Figure 28: The result of the above filtering action. Since this is a Single-Page-Application (SPA), the results are generated immediately without the need for page reloads. As shown, all clinics are in Central and Western. .......................................................................................................................................................................................... 44

Figure 29: Sorting of the listings when the user has "favorited" a doctor, and the ratings sorted in descending order. .......................................................................................................................................................................................... 45

Figure 30: The Doctor Detail page of the doctor Man Yuet-Tsin. .......................................................................................................................................................................................... 46

Figure 31: The Reviews section of the Doctor Detail page of the doctor Man Yuet-Tsin. The overall rating of this doctor is the average of all its ratings. This overall rating value will propagate its effect to the overall rating value of the entire clinic it belongs to. .......................................................................................................................................................................................... 47

Figure 32: Schedule Booking Now page.......................................................................................................................................................................................... 48

Figure 33: The state when the parameters have been selected, and the "CONFIRM BOOKING" button is ready to be clicked. .......................................................................................................................................................................................... 48

Figure 34: Calendar view of the Booking History page. The bookings that will happen in the future are indicated by a blue dot.......................................................................................................................................................................................... 49

Figure 35: Booking Record page showing the details of and controls for a specific booking item. .......................................................................................................................................................................................... 50

Figure 36: The indicator's color has turned orange after the client user cancels the booking. .......................................................................................................................................................................................... 50

Figure 37: The Booking Record page shown to clinic accounts. .......................................................................................................................................................................................... 51

Figure 38: The Booking Detail page shown to doctors. Since this appointment is not cancelled by the client, the doctor will be able to mark it as "Attended" or "Absent". .......................................................................................................................................................................................... 51

Figure 39: Doctor marking an appointment as "Attended". The four extra fields for the doctor to input will be displayed after selecting the status. .......................................................................................................................................................................................... 52

Figure 40: Medical Certificate issued via Spot-A-Doc. The QR code printed can be scanned, and it redirects the user to the medical certificate verification system for this medical certificate. .......................................................................................................................................................................................... 53

Figure 41: Successful verification of the digital signature. .......................................................................................................................................................................................... 54

Figure 42: Failed verification of a wrong digital signature. .......................................................................................................................................................................................... 55

Figure 43: One of the demo articles on the site advocating the importance of a good night sleep. .......................................................................................................................................................................................... 55

Figure 44: The Health Articles Listing page. .......................................................................................................................................................................................... 56

Figure 45: Admin Panel's Login page. .......................................................................................................................................................................................... 56

Figure 46: The All-Clinics section in the Admin Panel. .......................................................................................................................................................................................... 57

Figure 47: The Clinic Requests section in Admin Panel. The state of each clinic request is also displayed alongside each request entry. .......................................................................................................................................................................................... 58

Figure 48: The All-Articles section in the Admin Panel. There are buttons in each entry that lets admins perform certain actions including View, Modify and Delete. .......................................................................................................................................................................................... 58

Figure 49: Configurations of DNS for fyp22008.online .......................................................................................................................................................................................... 59

Figure 50: Part of the entire file hierarchy of the frontend React application. The separation of concerns principle is employed in its structure. .......................................................................................................................................................................................... 60

Figure 51: Code in the Express application (at the Presentation Tier) responsible for routing browser URL requests to index.html. .......................................................................................................................................................................................... 61

Figure 52: ProtectedRoute applied in the React application to control user access to specific groups of pages. .......................................................................................................................................................................................... 62

Figure 53: Project hierarchy of the Express application in the Application/Logic Tier. .......................................................................................................................................................................................... 65

Figure 54: Extensive use of environment variables in the backend provides protection of credentials and ease of modification. .......................................................................................................................................................................................... 65

Figure 55: node-forge used for RSA key generation. .......................................................................................................................................................................................... 66

Figure 56: Crypto library used for signing and verifying medical certificates. .......................................................................................................................................................................................... 66

Figure 57: Use of Postman for unit-testing .......................................................................................................................................................................................... 67

Figure 58: Successful run of Postman CLI on GitHub Action's CI/CD pipeline. .......................................................................................................................................................................................... 68
List of Tables

Table 1: Contributions and Roles of Different Project Team Members

Table 2: Wordings to Explain in the Below Flow Diagrams

Table 3: A pivot-table summarizing the characteristics of how accounts of different roles are handled.

Table 4: Description of the Purposes and Functionalities of the Admin Panel Sections

Table 5: List of API endpoints developed in the backend as the Application/Logic Tier
1. **Introduction**

As a computer science final year project at the University of Hong Kong, this work designs and implements an online B2C two-sided platform to enhance the operational processes of commercial clinics. The current methods of manual telephone reservation of doctor’s appointment and paper-based medical certificates have proved inadequate and inefficient, hindering user experience and the possible developments of the commercial clinic industry using information technology. The literature of the circumstances described will be elucidated in the Background and Literature Review section of this paper.

This project aims to provide a solution to these shortcomings by introducing an electronic commerce platform for commercial clinics, named by the project team as **SPOT-A-DOC**, implemented using modern software engineering tools, frameworks and project management methodologies. It will facilitate convenient online appointment reservation, as well as cryptographically verifiable, digitally signed medical certificates, in the commercial realm. Doctors and clinics can leverage the platform to enhance management, reduce administrative burden and build trust. Patients, in turn, will benefit from an easy, seamless and paperless experience across their medical journey. Appointments can be booked on-demand based on real-time availability, and certificates are digitally accessible anytime without risks of loss, damage or forgery.

By designing and implementing this platform, the project envisions technology and practice integrating to the mutual benefit of healthcare organizations, practitioners and public health.

### 1.1 Project Contributions

The table below depicts the different software engineering roles that the project team members each undertake, and their corresponding contributions in percentage.

*Table 1: Contributions and Roles of different project team members.*

<table>
<thead>
<tr>
<th>Student Name (UID)</th>
<th>Contributions</th>
<th>Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mak Ho Yin (3035693337)</td>
<td>50%</td>
<td>Backend, Architecture, DevOps</td>
</tr>
<tr>
<td>Wong Wing Kei (3035695189)</td>
<td>50%</td>
<td>Frontend, User Interface/Experience</td>
</tr>
</tbody>
</table>
1.2 Background and Literature Review

1.2.1 Modern E-Commerce with Two-sided Platforms

Electronic commerce has become increasingly prominent as the technologies that form the basis of the modern internet mature over time. Countless number of businesses have grasped such an opportunity to expand their virtual presence. Particularly, e-commerce in the form of two-sided platforms has gained traction in the twenty-first century. Firms running under this model facilitate transactions between businesses and consumers and harness the power of the Network Effect to sustain their operations. This model has been deemed successful in numerous industries globally such as video streaming sites, restaurant and hotel reservation, food delivery, and even virtual marketplaces. Their respective notable examples include Netflix, OpenTable, Agoda, Uber Eats and Amazon.

In Hong Kong, numerous well-known online two-sided platforms have also been widely adopted by firms to grow their business and consumers to improve their quality of life. OpenRice.com, for example, provides a platform for restaurants to create an online presence and allows consumers to directly make reservations via the platform. Price.com.hk, on the other hand, provides a platform for retailers of electronic appliances to advertise their goods and allows customers to compare the prices of the same item across sellers, and directly make purchases via the platform. These all show that there are many possibilities for business development under the model of two-sided platforms. However, it is observed that there is one industry where such model is highly applicable but still nonexistent in Hong Kong – the commercial clinic industry.

1.2.2 E-Commerce in the Commercial Clinic Industry

Commercial clinics (or private clinics) revolve in the everyday life of Hong Kong citizens. As public medical services become gradually inadequate, commercial clinics will play an increasingly important and dominant role in the city [1]. Yet, the reservation of doctor’s appointments nowadays still relies mainly on traditional means, such as telephone booking. Take Town Health Medical Services\(^1\) as an example. Despite that it is one of the largest private clinic brands in Hong Kong, its website does not provide any functionalities that allow prospective patients to directly make appointments [2]. Instead, only a list of addresses and phone numbers are displayed, and

\(^1\)Town Health Medical Services Limited (Chinese: 康健醫療服務有限公司) is “one of the largest and most extensive local chain medical service networks”, according to the firm’s official website. https://townhealth.com/en/investor-relations/performance-highlights
users are supposed to make phone calls to reserve an appointment or enquire the doctors’ availabilities. They can also walk into the clinic directly based on the address printed on the web page. It appears that commercial clinics themselves are not catching up the pace of technological advancement in the realm of web technologies, and they still rely on rather obsolete practices to fulfill their operational needs.

Figure 1: The page showing the list of doctors under the General Practice category. As shown in the figure, the only means for a user to approach the doctor would be to make a phone call or walk-in.

Besides the inadequacy and obsolescence of the booking functionalities proprietarily provided by existing clinic brands, the existing clinic-related two-sided platforms in Hong Kong are also unable to deliver the experience consumers would expect nowadays being B2C e-commerce presences. One example would be Seedoctor.com.hk, a popular platform for prospective patients to search for doctors in different medical categories [3]. Although doctors can submit and update their public information on the platform, prospective patients would only be content consumers, where the flow of the content is unidirectional. Such an application of two-sided platforms has not fully harnessed the possibilities of the modern World Wide Web, and it seems to lack behind e-commerce platforms in other industries, such as the restaurant dining industry.
Contrary to the condition in Hong Kong, the development and adoption of two-sided platforms that provide online reservation service of doctor’s appointments are already prominent and mature in numerous other countries. In the United States, for example, the two-sided platform Zocdoc has accumulated more than 68 million patients in 2022, and the platform has been providing online doctor’s appointment reservation services for more than a decade [4]. According to Zocdoc, the platform was developed initially to simplify the process of clinic booking so that the public no longer needs to rely on telephone bookings [5]. This had been a big step forward for years compared to Hong Kong’s commercial medical industry. It would be beneficial to most clinic businesses and client patients when a platform for commercial clinics with such capabilities is brought about into the city.

1.2.3 Use of Digitally Signed Medical Certificates

Another topic in the medical industry that has been of concern would be the forgery of doctor issued medical certificates. In Hong Kong, it is still common that traditional paper medical certificates are issued by registered medical practitioners so that employees could prove to their
employers the necessity of a sick leave. Although the certificates are mostly pen-signed and handed to patients, they serve critical purposes such as a prerequisite for an employee’s eligibility of paid sick leave or maternity protection, according to Cap. 57 Employment Ordinance of Hong Kong Legislation [6]. Paper medical certificates are not only vulnerable to forgery, but are also prone to physical deterioration, damage or loss. Registered practitioners also lack oversight over the medical certificates they issue.

To address the downsides of paper medical certificates, cryptographically verifiable medical certificates could be incorporated into medical systems in Hong Kong. These certificates would be signed by medical practitioners using electronic means, where only the right person can be involved in the signature process. Moreover, the tempering of the information in the database or the signature, if implemented correctly, would invalidate the issued medical certificate. These certificates are also always accessible by the patients and the doctors whenever needed and can be publicly verified by any persons concerned. The Hospital Authority (HA) in Hong Kong has rolled out their own version of digitally signed medical certificates for public hospitals and clinics in February 2023 [7], and they require all certificates issued March onwards to be digitally signed [8]. However, the HA’s medical certificate system is, to date, only deployed for the public medical system. Commercial clinics, especially those with smaller business scales, would not have the incentive or resources to incorporate such a feature into their system.

![Figure 3: The Medical Certificate Verification System launched by the Hospital Authority in February 2023.](image)

### 1.2.4 The Two-sided Platform Solution for Hong Kong

To sum up, while technological advancement of the Web has transformed numerous industries, the commercial clinic industry in Hong Kong remains relatively outdated in its mode of operation and service delivery. Existing means of doctor’s appointment reservation and the issuance of
medical certificates seem to undermine user experience and business sustainability. At the same time, promising solutions such as two-sided platforms and digitally signed medical certificates have not been adequately adopted. There are clear opportunities for progress by learning from the experiences of other industries. By developing an online B2C two-sided platform for appointment reservation and integrating cryptographically verifiable medical certificates, the commercial clinic industry can better harness web technologies, improve operational efficiency, enhance service quality and build trust with users. The proposed solution aims to propel the commercial clinic industry in Hong Kong into the digital age and remedy the shortcomings of traditional practices.

1.3 Objectives

This project aims to provide a modern alternative to the traditional practices of doctor’s appointment reservation and medical certification in commercial clinics through the design and implementation of a B2C two-sided platform using modern software engineering techniques. The platform will benefit commercial clinics by introducing a modern way to perform booking, improving the user experience through the integration of real-time availability and advanced booking features, and establishing a system for the issuance of verifiable medical certificates that is secure and works for these clinics.

Key objectives include:

- Design, implementation and deployment of an online B2C two-sided platform with the following functionalities:
  - **Core features**: Find/book available doctors for appointments and issue trusted medical credentials.
  - **Appointment management features**: User authentication, user accounts, real-time doctor availability, calendar management and on-demand appointment booking.
  - **Electronic commerce features**: Search/filter/browse clinics and doctors, read/add ratings and comments, browse platform-managed health articles.
  - **Medical certification features**: Tamper-proof medical certificates with digital signatures for verifiability.
• Leverage cloud computing services, software frameworks and DevOps best practices for a secure, scalable and reliable solution.

• Utilize agile methodologies, version control, unit-testing and CI/CD pipelines to continuously deliver high quality code.

In short, through innovative use of technology and industry-established practices, this project delivers a solution that revolutionizes how patients interact with healthcare.

2. **Methodology**

This section describes thoroughly the methodologies adopted in attempt to accomplish the project objectives, including the considerations on aspects in electronic commerce, the digital signature algorithm for implementing medical certifications, the architectural and design principles in web development to be involved in the implementation, and the software engineering methods and practices to be adopted.

2.1 **Aspects in Electronic Commerce**

The solution to be delivered in this project is intended to be a Business-to-Consumer (B2C) two-sided platform. The characteristics of electronic commerce websites with relevance to the realization of our doctor’s appointment reservation platform, including the adoption of measures that will facilitate personalization for consumer users and the revenue models for maintaining the platform’s operability, will be discussed.

2.1.1 **Personalization**

Personalization is an important aspect in customizing the experience of consumers when they visit the electronic commerce platform. According to a business guide on Indeed, personalization induces multiple benefits to platform businesses including increased sales, better engagement and more customer recommendations and referrals [9]. Some aspects of personalization, according to the guide, are to utilize user-generated content, create customer profiles, gather customer feedback, and even to adopt analytics tools to acquire knowledge about customers’ behaviors. This project will adopt some of these measures by integrating them into the appointment booking system.
In particular, doctors that are present on the platform can be reviewed by clients, where each piece of review will contain a score rating from 1 to 5, and optionally some comments. A client will also be able to label specific doctors and include them into his personal favorite-doctors list. With these mechanisms, the listing of the doctors and clinics in the search and browse sections of the platform will be able to generate results sorted and filtered in accordance with the client’s own booking history, own favorite-doctors list, as well as each clinic’s overall score ratings. This makes the search results of the doctors more relevant and useful to clients on the platform and ultimately bolsters the appointment reservation experience (to clients as prospective patients) and performance (to clinics as businesses).

There exists many more strategies and techniques that are possible to be incorporated into our platform, including the use of machine learning and cross-domain cookies for performing analytics of user data. These methods will likely deliver broader and more accurate doctor recommendation results on the browse page relative to the measures described in the previous paragraph. However, the use of these technique is not a focus of this project, and therefore will not be included in the final product.

2.1.2 Revenue Models

To sustain the business that operates the platform, it is crucial that one or more revenue models are adopted on the platform to generate income. Some common revenue models adopted on electronic commerce websites include the Catalog Revenue Model, Fee-for-Content Revenue Model, Advertising-Supported Model, Fee-for-Transaction Model, and the Fee-for-Service Revenue Model [10]. These models are all applicable in the realm of B2C platforms, and they all include some kind of income source from the business or the consumer, or even parties that are not directly involved in the platform. For example, modern online platforms monetize their websites through the Advertising-Supported Model by embedding JavaScript plugins from Google AdSense [11].

In the context of this project, a combination of the Fee-for-Transaction model and the Advertising-Supported Model can in principle be acquired. A clinic can be charged a certain amount of commission after the completion of each transaction (successful appointment reservation) through the platform, and spaces on the user interface (specifically the page for browsing lists of doctors and clinics) can be allocated for displaying advertisements fed through the Google AdSense
service for monetization. Other revenue models may be less suitable for application in our B2C platform. For example, the Catalog Revenue Model is more tailored for e-marketplaces that involve purchase of products that directly incur sales on the platform, while our doctor’s appointment reservation system acts as an intermediary that matches businesses and consumers.

Unfortunately, since Google AdSense requires that the web page be compliant to the Google Publisher Policies and be reviewed and approved by Google, and the incorporation of online payment gateways into our product involves legal and compliance issues, which are not the focus in the context of this final year project, the revenue models mentioned applicable will serve merely as a conceptual possibility.

2.2 Medical Certification using Digital Signatures

The RSA asymmetric cryptographic algorithm will be applied for the generation and verification of signatures of the data associated with a completed appointment. RSA is a widely adopted algorithm for the generation of a public-private key pair for encrypting and decrypting messages, and also for signing and verifying a message’s authenticity, depending on how the generated key pair is utilized [12]. When it is used for the purpose of digital signatures, the message will first be hashed using a hash algorithm such as the SHA-1 Message Digest Algorithm, and then be encrypted using the private key, where the private key is kept secret by the message signer and the public key is disclosed to the public so that whoever wishes to verify the message’s authenticity can do so. To perform verification, the plain message is hashed using the same hash algorithm used in the signing process again, then the public key will be used to decrypt the signature. If the decrypted hash matches the just-computed hash, that means the message is indeed what is sent out by the message signer who possesses the private key, assuming the private key is not leaked.

In the context of the project, where the signing of doctor’s appointment data is desired, the RSA algorithm will be used in a way similar to above. When a doctor registers an account in the system or changes his passcode, the passcode will be used for subsequent signing of medical certificates. By default, most RSA implementations rely on pseudorandom number generators (PRNG) to generate the seed responsible for deriving a prime-number pair, and a higher entropy seed makes the generated key more unbreakable [13]. Since a doctor will practically need to sign multiple pieces of medical certificates, it would be desirable that the same passcode will yield the same
public-private key-pair in a fashion that does minimal compromise to the security brought about by the PNRGs.

The methodology to achieve this would be to implement password-based cryptography based on the RFC8018 specification [14]. Firstly, a Key Derivation Function (KDF) will be used to generate seeds from doctor passcodes. The result generated by the KDF will replace the seed generated by the PRNG, thus resulting in a deterministic pair of public-private keys. Moreover, to avoid rainbow table attacks induced by password-derived deterministic seeds, each doctor will be assigned a pseudorandom salt value whose generation is independent of password provided (or any information associated with the doctor), used also as input to the KDF. The KDF, taking the user password and a salt, will also be configured an iteration count of $100000$, which strives a balance between protection against trial-based attacks and user-perceived performance. These increases the entropy of the key pair generated and makes it difficult to crack while provides the ability for us to implement password-based cryptography for medical certification.

The below flow diagrams depict the conceptual flows of which the RSA key-pairs are generated, the appointment data is signed, and the signature is verified, for the medical certification feature of our e-commerce platform. Some terminologies and wordings that appear in the flow diagrams are clarified in the following table.

\textit{Table 2: Wordings to explain in the below flow diagrams.}

<table>
<thead>
<tr>
<th>Wording</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{DPW}</td>
<td>Doctor’s passcode for key generation and signature.</td>
</tr>
<tr>
<td>\textit{AData}</td>
<td>Appointment Data (to be signed by a doctor).</td>
</tr>
<tr>
<td>\textit{PK}</td>
<td>Public Key.</td>
</tr>
<tr>
<td>\textit{SK}</td>
<td>Private Key (SK stands for Secret Key).</td>
</tr>
<tr>
<td>\text{GenerateSalt()}</td>
<td>A function that outputs a pseudorandom salt value.</td>
</tr>
<tr>
<td>\text{GenerateRSAKeyPair()}</td>
<td>A function for generating a pair of RSA keys using PDW and the salt value that belongs to the doctor. KDF is assumed to be handled already in this function.</td>
</tr>
<tr>
<td>\text{Hash()}</td>
<td>A hash function that takes an input and maps it to a fixed length string representation.</td>
</tr>
<tr>
<td>\text{Encrypt()}</td>
<td>An RSA encryption function.</td>
</tr>
<tr>
<td>\text{Decrypt()}</td>
<td>An RSA decryption function.</td>
</tr>
</tbody>
</table>
Figure 4: RSA Key Generation flow during doctor registration using the doctor's supplied passcode.

Figure 5: Process of signing a piece of appointment data. DPW is input by doctor and checked against database’s record for identity verification. It is then used for regenerating the key pair to obtain the private key for signing.
As shown in the above process illustrations, for the certificate signing process to work, the correct doctor’s password (same as that provided in the key generation process) must be provided every time for the private key to be regenerated for signing, ensuring that the signatures must have been produced by the right person, and there will be nowhere the private key is persistently stored in the system since it can only exist with the intervention of the doctor holding his DPW.

In short, the design of the flows together with the use of password-based asymmetric cryptography in accordance with the RFC8018 specification will serve as the foundation to our platform’s electronic medical certification feature. Its implementation in our platform, specifically the parts denoted by the GenerateRSAKeyPair(), Hash(), Encrypt() and Decrypt() functions will rely on industry-trusted software libraries, such as modules that are based on OpenSSL and are widely adopted by enterprises.

### 2.3 Software Architecture and Design

To achieve the design of a modern and robust B2C e-commerce platform, the Three-Tier Web Architecture with the incorporation of cloud computing services will be utilized. This section also introduces the choice of specific technologies that will be involved in each architectural tier, and also discusses some software design and security considerations.
2.3.1 Adoption of the Three-Tier Architecture

The e-commerce platform in this project will employ a three-tier architecture. The gold standard in today’s industry for building quality software applications on the web would be the three-tier architecture, where the entire application is partitioned into three separate infrastructural components – the Presentation Tier (the frontend application running on a web server), the Application/Logic Tier (the backend application running on a server to orchestrate data and serve frontend API requests), and the Date Tier (the database server for storing user-generated data) [15]. These tiers are separately deployed and interact with one another through HTTP web requests, and such an architecture results in benefits such as improved maintainability and security that makes it superior to its client-server predecessor.

![Figure 7: An illustration of the three-tier-architecture. A more complete picture will be delivered in section 2.3.3.](image)

In terms of maintainability, separating the application into different tiers decreases the dependencies and complexity of the code that serves different purposes, such as interface rendering and business logic processing. The scale of an e-commerce web application like ours will likely enlarge in the future, considering the fact that more e-marketing or other features may be introduced to the platform over time. Putting separation of concerns of the architecture into account, rather than designing it in a monolithic manner, will ensure our code’s quality and increase its longevity, reducing efforts in conducting code revamps or the potential need to rewrite the entire application, which may be very costly. On top of that, in the context of our project, which is a teamwork, different team members will be made responsible for the development of particular tier(s), where the collaboration of our work relies on a defined and negotiated API instead of pieces of actual code in the software, speeding and easing development efforts.

Security is also a crucial aspect of the design of an e-commerce application like ours. Since our database server will house sensitive medical information such as our clients’ HKID number, it is desirable that a middle tier exists to safeguard the application’s access to the data. Using the three-tier architecture ensures that the Presentation Tier, which may more easily be prone to different
kinds of web attacks, will not have direct access to the database, and that all instances of data access are managed by the Application/Logic Tier in the middle.

2.3.2 Adoption of the MERN Stack

In terms of the technologies that will be acquired to implement the platform, our team will be using what is commonly known as the MERN stack, which includes four technologies – MongoDB, Express, React, and NodeJS. MongoDB is a non-relational NoSQL database running on the Data Tier that stores data in documents instead of rows and collections instead of tables. The data modelling aspect of the software will be discussed in later sections. Express is a JavaScript web framework with concepts such as middleware and routers running on the Application/Logic Tier. React is a JavaScript library used for building single-page applications on the Presentation Tier. NodeJS is a JavaScript runtime environment that is responsible for running the Express and React applications. These technologies each have their own responsibility in the three-tier architecture, and they collective allow the implementation of a full-fledge web application like our B2C platform [16].

Using the MERN stack is advantageous to building our e-commerce platform. To begin with, the communication between the Presentation Tier and the Application/Logic Tier commonly relies on transmitting data in JSON format. Concerning the use of MongoDB, since it stores data in a similar fashion using documents, where this data structure can be easily interpreted as JSON, the backend does not have to perform a lot of data manipulation work before passing data in and out, to and from the Presentation Tier and the Data Tier [17]. This eases our development, increases the flexibility in data handling, and also makes our code cleaner. On top of that, unlike relational databases where empty values will be populated with NULLs, MongoDB does not have a fixed schema, making it a good fit for e-commerce applications. This is because a product record, in our case a clinic/doctor record, is not likely to possess all fields that are defined in a clinic/doctor schema. This results in a cleaner set of data that consumes less storage space yet retains the same amount of information.

Moreover, from an architecture-wise standpoint, all technologies involved are JavaScript based, meaning the team does not have to possess knowledge of multiple languages to successfully implement the application [17]. The communication of ideas between team members will also be more efficient because the concepts involved will likely be based on JavaScript. Lastly, React
supports the creation of reusable and dynamic UI components that makes it more ideal for building e-commerce websites like ours that will have numerous pages with similar designs. It eliminates the need to write complex Vanilla JavaScript event handlers coupled with CSS components that may be harder for frontend developers to manage. Also, component libraries packed with already-built UI components exist in the React community, including the MUI component library which will be adopted to build our platform’s interface. This reduces the effort and necessity in designing and implementing individual components from scratch.

2.3.3 Adoption of Cloud Services

Instead of using physical servers to host the tiers of our e-commerce platform, cloud services of different kinds, including Platform-as-a-Service (PaaS) and Database-as-a-Service (DBaaS), will be adopted in our project. These two schemes in cloud computing utilize containerization technologies to abstract away the underlying infrastructural complexity that is possessed in traditional virtual machines offered by Infrastructure-as-a-Service (IaaS) schemes, making it more efficacious for the team to develop our e-commerce platform. Moreover, these services have promoted transitions in the commercial realm from in-house infrastructures to cloud computing technologies in the last decade by offering advantages in multiple aspects such as availability, scalability, data security, and cost, according to a recent paper analyzing cloud computing advantages in e-commerce [18].

In the context of our doctor’s appointment reservation system, these advantages have significant relevance. For example, in terms of scalability, the unused computational resources such as virtual machines can be disabled, and more computational resources can be allocated during high usage scenarios. This has huge implications to health-care e-commerce platforms like ours, where usage of the platform will likely increase during winter when seasonal influenza is more active such that more individuals will be seeking medical services. The team operating the doctor’s appointment reservation platform will not be required to manually purchase and set-up physical infrastructure to scale-up, and there will not be unused hardware that are idle during off-peak seasons.

On the other hand, cloud service providers (vendors) will also guarantee infrastructural upgrade, support and physical protection. The burden introduced by these work is called “undifferentiated heavy liftings” in Amazon Web Service’s whitepaper [19], and they argue that the provision of these services by cloud providers allows software teams to focus on the implementation of core
business functionalities of their application rather than having to deal with infrastructural matters. In our project, the ability to speedily deliver web application features can determine the competency of our B2C platform in the commercial clinic industry, as we may have an advantage over our competitors in terms of infrastructural burden involved in rolling out features.

Therefore, with the multiple advantages of cloud adoption in mind, this project will utilize cloud services in the deployment of the technologies utilized in different tiers of the three-tier web architecture.

### 2.3.4 Choice of Cloud Providers

Cloud computing offerings from multiple cloud service providers (CSP) will be adopted for the implementation of the three-tier web architecture using the MERN stack. In particular, two CSPs, namely Heroku and Microsoft Azure, will be used by in the project to facilitate cloud adoption. The React and Express applications will each be deployed to a Heroku Dyno, which is a PaaS container service. The MongoDB cluster will be set-up on MongoDB Atlas, which is a DBaaS platform, and it will be configured to run on Azure’s data centers [21]. Lastly, the static resources such as the profile icon images uploaded by the doctors to our platform will be stored using Azure Blob Storage, which is a persistent object storage service.

The complete picture of how the three-tier architecture for our B2C platform is realized using the MERN stack together with their corresponding cloud technologies is depicted in the above figure. It would be worth noting that the three architectural tiers are physically isolated from one another.

---

*Figure 8: A diagram in UML fashion depicting how cloud computing, the MERN stack, and the three-tiers are architectured. Another diagram at a lower level regarding software design will be presented in section 2.3.5.*
and are only interacted through network requests. This is different from a layered architecture where the components are run within the same physical infrastructure [22].

2.3.5 Software Design Considerations

Designing a software with quality code increases its longevity while reduces the costs and efforts in subsequent maintenance. Our implementation of the doctor’s appointment reservation system, no matter which architectural tier, will take a Strategic Programming Approach rather than a Tactical Programming Approach. The former approach is when the primary goal of software development “must be to produce a great design, which also happens to work”, rather than the latter approach which only focuses on completing the implementation of the intended feature at speed, which is simply to produce working code without thinking ahead [23].

In our project, multiple design principles including object-oriented design, separation of concerns and modularization will be of concern and be incorporated into the final product. These principles together will decrease the complexity of the code and makes it easier for future understanding and manipulation to the codebase.

As shown in the diagram above, both the Presentation Tier and the Application/Logic Tier employ the separation of concerns principle that makes the role of different components more distinct. For example, the routing logic and the style definition logic are housed separately in the React app, and the routing logic and the models that control the access to the database are also handled.
separately. On the other hand, the Application/Logic Tier will also adopt modularization such that functions that facilitate the implementation of specific features will reside in its own module. There will also be a module that places utility functions for dealing with general functionalities, such as for error handling. Moreover, an Entity class will be defined in the Express app to abstract the process of manipulating the database. The Express route logic will therefore not need to contain code that accesses the database models, but rather through the instantiated object representing the model, with class methods that help accomplishing the data manipulation flows.

2.3.6 Security Considerations
The implementation of our B2C platform will incorporate several measures to safeguard it from various kinds of web attacks, particularly the eavesdropping of web packets and code injections.

The first security mechanism we adopt will be the HTTPS protocol – the secure version of HTTP. All the information transmitted between the three tiers are encrypted using cryptographic algorithms and signed using digital signatures under the Public Key Infrastructure scheme\(^2\). This ensures that the sensitive data transmitted among system components, including our client users’ medical history and their credentials, will not be easily accessible by illegitimate parties.

The second security mechanism will be input filtering and sanitization – the act of checking against and cleaning inappropriate and potentially harmful user input. This is to mitigate the risk of code injection attacks, where adversarial users may embed harmful JavaScript or SQL scripts in their input to compromise the system. Despite that MongoDB (which is a NoSQL database) is adopted in the design of our B2C platform, the data stored may be loaded subsequently to business intelligence systems that make use of SQL for data querying, such as RDBMS data warehouses. The worst scenario could have the entire database in the data warehouse dropped, destroying all historical data if incremental backups have not been prepared.

2.4 UI/UX Considerations

For an e-commerce platform to be successful, its ease of use, aesthetics and perceived performance will need to be well and thoughtfully designed. A good user experience (UX) and user interface

\(^2\) The “Digital Signature” here is the one provided via Public Key Infrastructure to facilitate secure transmission of web data rather than for the signing of medical certificates. They are two different applications of the digital signature concept applied in this project.
design should be customer-centric, meaning that the users’ perceived experience, such as how they will navigate different links, is of utmost importance. A bad design could push customers away from the B2C platform, undermining its business performance. User experience design also caters aspects such as how accessible the information on the platform is, how long a web request is blocking the user interface (sometimes by displaying a loading animation), and how adaptive to different display sizes the user interface is.

In terms of UX, the workflow that the users interact with the functionalities once they launch the home page of the platform and how the elements on the pages will be structured are depicted in the Wireframe document in Appendix I. Besides this, the rendered HTML content in the frontend will be made interpretable by screen readers so that users with visual impairment will not be disabled from accessing the information and the features. Also, the sizes of the images to be loaded onto the site will be kept at a size limit and compressed so that the HTTP requests for downloading static content will not undermine the loading time of other content.

In terms of UI, the Responsive Web Design principle will be considered. This principle stipulates that the user interface of a web application should be adaptive to the screen size of the device of which the browser is opened. For instance, since the display of smartphones is much smaller than that of desktop computers, it would be impractical and not user-friendly for the UI’s desktop-version to be directly served to mobile browsers. To cater this, our B2C platform will utilize breakpoints in the React MUI component library (which is a React equivalent of CSS’s media queries) to rearrange the layout so that it fits the device with the corresponding screen-width ranges defined by threshold values. An example of a planned responsive design feature would be that the top navigation bar will become a togglable side panel when the application is opened using a mobile browser.

2.5 Data Modelling in NoSQL

In this project, MongoDB will be adopted for the storage of the platform’s data. Unlike traditional RDBMSs, MongoDB is a NoSQL, Document-based DBMS which supports tunable consistency. This makes the database suit different use case scenarios, such as transaction processing that requires stricter consistency, as well as big data processing that values speed more than accuracy. In the context of our doctor’s appointment system, the database will be configured to be using only
one primary node in the Replica-Set, where all secondary nodes will be responsible for fail-over recovery processes. This also makes our system more reliable in terms of data storage and access.

In terms of data modelling, MongoDB offers flexibility and also a different mechanism and principle to achieve so. Instead of relentlessly performing normalization and defining foreign key constraints to logically link database objects, schema designers are allowed to choose between embedding documents in other documents and using foreign key constraints to reference attributes in other schemas. The factors that affect the consideration of using which method would be data access patterns and the quantity of the data in the many-side of one-to-many relationships. In MongoDB, one-to-many relationships can be categorized into One-to-Few and One-to-Squillions relationships (to distinguish the difference between the case when there are only a few documents and that of a large quantity of documents) [24].

In our project, we will be following the best practices of schema design provisioned by the official MongoDB documentation. Since the project will be adopting an agile development approach, the schema design for the system is accomplished in a high-level manner, and attributes may be added or removed based on the needs during actual development. Nonetheless, the high-level design and the subsequent development process will still adhere to the schema design practices as mentioned.

![Diagram of MongoDB Collections](image)

*Figure 10: An illustration of relationships between our MongoDB Collections. The Doctor collection is identified jointly by Clinic’s _id field and its own full_name field, which can be thought of as a weak entity in RDBMSs.*
As shown in the above diagram, since User and Client have a One-to-One relationship such that each User is mapped to exactly one Client (if the User object is of a Client role, otherwise 0 Client, and must be mapped to 1 Clinic), it would be natural to embed the Client document into the User document. Doing so allows the two documents under two schemas to be retrieved and updated in a single query, such as via the find-one-and-update query, which eliminates the need to perform multiple update queries in such scenario. On the other hand, since one client may be related to many Appointment documents, which the quantity is not bounded, using key referencing ensures that there won’t be some documents that are very large in size due to an excessive number of documents embedded. Moreover, since the Appointment schema is related to two other schemas (Doctor and Client), it would be inappropriate to embed one Appointment document to both the Doctor and the Client documents. Doing so induces the need to run multiple queries to update the same document, which are embedded in two separate documents. This will risk data inconsistency, and the software code for performing such data manipulations will also be more complex, leading to degraded software quality.

2.6 Software Engineering Methodologies

To achieve the successful development of our B2C platform, it is crucial that the project will adopt a set of professional software engineering methodologies, including project management and DevOps. This section discusses the principles and frameworks that will be adopted in our project.

2.6.1 Adoption of Agile Development Methodology

This project runs in Agile and is supported by Scrum. Agile is a project management framework and Scrum is one of the Agile methodologies used to break a project into different stages called sprints. This framework and those alike have been utilized by an increasing number of industry professionals in software engineering projects. It guarantees the quality and functionalities of the product in fulfilling the needs of its users and allows room for flexibility during the course of its development.

One crucial aspect of this approach is the ability for the project team to figure out the underlying issues of the product’s design quickly after each sprint (iteration) and be able to correct them accordingly. This ensures that our final deliverable will, to the greatest extent, deliver practical value and achieve its intended objectives. Since our B2C platform contains multiple core features
with substantial complexity, its design and planning at the preliminary stage may not be sufficiently comprehensive and future-proof, and issues may arise during its implementation and testing phases. Adopting Agile and Scrum increases the likelihood in the project’s success, and makes sure that the implemented core features, including appointment reservation and medical certification, will be practical and are aligned to the project’s objectives.

To facilitate the use of Agile for project management, Notion, a project tracking tool widely adopted by industry project teams, is employed by the team. Notion comprises multiple features for project management, including collaborative and interactive backlogs, boards and Gantt chart views [25]. It also supports the concept of Sprints in the Scrum framework to be incorporated into our project backlogs, while having much higher flexibility and versatility compared to other tools like Atlassian Jira because of its ability to be used like an ordinary note-taking application, alongside and interchangeably with the project management functionalities. It is ideal for our final year project, where some aspects in our system development and project management processes may be different from a typical one in the industry due to our academic nature.

2.6.2 Version Control and Collaboration

Version control is a widely adopted measure by software development teams to control code versions. The version of the code that a team member is currently working on, the creation of branches that separate different development focuses of the codebase, and the submission of newer code versions for other members to perform code reviews etc. are all easily doable using a set of terminal commands. Specifically, the most popular version control system, Git, will be adopted by our team. GitHub, the platform for teams to host their remote Git repositories, will also be adopted by the team to facilitate collaboration.

Specifically, our team will be adopting the Feature Branch Workflow (FBW) for the development of the B2C platform. The FBW is a workflow where new features are developed in a separate branch from the main branch, where the main branch contains code that is clean and production-ready [26]. Team members are supposed to only commit to the feature branch and not the main branch, and the feature branch will be merged into the main branch when it is ready. The code that is merged and deployed into the main branch must pass all test cases in the Continuous Integration flow, which will be discussed in the next two sections. Such a workflow is beneficial to the production of quality software, and it is also a crucial consideration in the system
development process of our platform. Since the software architecture of the platform is partitioned into three separate tiers so that they are separately developed and are deployed to different environments, there needs to be a version of the code that is ready for performing integrations of the tiers, as well as versions that are undergoing development of new features or bug fixes. For example, there needs to be a version of the code in the backend that are stable enough for the frontend counterpart to make web requests to. Using this workflow poses minimal disruption to the integration of the tiers while allows the continuous development of other features in the feature branch, easing and speeding the software engineering process of our project. A diagram illustrating the workflow incorporated into the continuous testing and CI/CD processes will be shown in section 2.6.4.

2.6.3 Continuous Testing – Unit-testing on Postman

As discussed in the previous section, our project will be adopting the Feature Branch Workflow for version control, and there is a crucial step of merging the feature branch codebase into the main branch. Since the code in the main branch must be at a stable and production-ready state, there needs to be a mechanism that facilitate the testing of all the features and code to ensure their quality and stability before permitting the merge operation. Our team will be adopting the process of unit-testing to perform quality assurance (QA) in the version control workflow.

Specifically, the development of the backend tier will utilize the Postman API Testing platform to conduct unit-testing. Postman facilitates the manual and automated testing of APIs using a graphical interface, and it is also capable of integrating with other DevOps systems [27]. It is a widely-adopted tool in the industry, and a large number of companies adopt Postman in their software development life cycle for API testing purposes³. Since there is a need for our B2C platform to perform unit-testing of the API, this popular and industry-proven tool will be adopted by our team during the development of the platform.

On top of that, the design of the unit-testing pipelines for our software will follow best practice in test design and will adopt the typical “Setup-Test-Teardown” flow of unit-tests. The Setup stage will involve API calls to both or either the database and the backend API to prepare the necessary

---

³ In 2020, Postman had accumulated more than 13000000 developer users globally, and was also acquired by more than 800000 companies. In that year alone, it has drawn more than four million new users, and according to the company, they are looking to guide the software industry through the next generation of APIs. [29]
environment and database state for subsequent test cases. For example, to test whether the certificate verification mechanism is functioning properly and whether the results are within expectation, the database records of doctors, patients, and appointments will be needed, and these records together will have to contribute to a state when the doctor is ready to sign the medical certificate. The Tests stage will involve API calls that are sent to the backend server to mimic the scenario that will happen in real-world usage, including those of unsuccessful scenarios, such as whether the input of an invalid certificate signature will result in a failure response. These tests should ideally run through all lines of the code in our software to achieve high code coverage, but whether to implement the QA test cases that may not be of high importance will depend on whether sufficient amount of time is allowed for our team to achieve so. Moreover, each test case will contain one or more assertions to the results, and the unit-test will be deemed failed if one or more assertions have not been passed. The Teardown stage will be composed of API calls that will access the database to revert the modifications that have incurred due to the execution of the test cases. The figure below shows an example configuration of Postman to achieve the described unit-testing process.

Figure 11: An example unit-test prototyped for illustration. This unit-test contains the Set-up and Teardown processes with API calls, and within them are specific features to be tested.

Figure 12: Test assertions can be written using JavaScript in Postman. Environment variables can also be set to assist in test case construction.
2.6.4 Continuous Integration and Deployment (CI/CD)

Continuous Integration (CI) and Continuous Deployment (CD) are continuous software engineering processes adopted in the industry by development teams to accelerate the delivery of the software products while assuring their quality. Particularly, CI is the practice of merging and integrating new code into the main branch for deployment through rapid build and test processes, and CD is the practice of continuously deploying the software product to production environment [28]. These practices will be achieved as a whole in this project by integrating version control and continuous testing into a pipeline that will be automatically run upon code merge.

![Diagram of CI/CD workflow](image)

**Figure 13: The CI/CD workflow for the backend development of our B2C platform. PR stands for “Pull Request”**.

The above diagram shows two flows in which the CI/CD process will be realized in this project. One flow is to be run during feature branch development, and another will be run during the decision to merge the feature branch into the main branch. To achieve these two flows, the GitHub Actions platform, Postman CLI, and Heroku Webhooks will be set up to work collectively, where each of them serve different responsibilities. The GitHub Actions platform helps setting up and running CI/CD pipelines defined in YAML files on their virtual machines. Heroku Webhooks listens to events broadcasted by GitHub upon repository actions, specifically for the case of continuous delivery, the broadcast of the Push event (the action of pushing one or more commits from local Git environment to the remote one). The web hook pulls the codebase in a specified branch from Git to Heroku and deploys it. Lastly, the unit-test pipelines built on Postman will be triggered via Postman CLI, where the commands for running the tests will be executed via GitHub Actions.
Since the CI/CD flow we designed involves the use of a staging environment and a production environment, we will be acquiring two Heroku Dynos for the deployment of the backend application as well as two logically separated databases on the MongoDB cluster for the deployment of a production database and a test database.

3. **Project Results**

This section elucidates the actual design and implementation of the platform achieved through our year-long project development efforts. It first provides descriptions to the actual functionalities implemented in SPOT-A-DOC (our B2C platform), including the two core features – Appointment Booking and Medical Certification. Screenshots will also be included to demonstrate and showcase the completed features. It then discusses the implementation details, including how the completed software code, configured infrastructures and practiced software engineering principles have realized the project methodologies.

3.1 **Elaboration on Completed Features**

3.1.1 **Main User Interface (UI) Design**

The main user interface of the SPOT-A-DOC platform contains a navigation bar at the top so that users can navigate across different main functionalities and features, such as to search for doctors and clinics, to verify medical certificates, to read health articles, as well as to perform login and registration actions. The navigation buttons on the bar also change if the user is authenticated such that the Login, Registration and Join Us buttons will become Logout. The home page where users launch into when they first access the application contains a slider that each contains a background image to represent specific features, and users will also be able to navigate into the feature when they click on specific slides. Scrolling down the home page, there will be panels showing e-commerce styled recommendations to the user, including a panel for Popular Clinics, another for Popular Doctors and the last one for Health Articles. Without searching, a user will already be able to view some clinic and doctor choices that he may opt for with ease, and users will more likely make reservation decisions, favoring clinic businesses on the platform. Moreover, the page does not contain too many data and information at once, making it easy for users to browse. The design and the colors chosen are also consistent, the text content on the page is readable, and there is not an excessive number of animations that potentially distracts and perplexes the user. Such design has been based on customer centricity principles, and it makes the platform more
appealing to the potential and current client and clinic users concerned. The below figures shows the described main user interface in our implemented platform.

Figure 14: Home Page of SPOT-A-DOC.

Figure 15: The Popular Clinics panel on the Home Page of SPOT-A-DOC.
Figure 16: The Popular Doctors panel on the Home Page of SPOT-A-DOC.

Figure 17: The Health Articles panel on the Home Page of SPOT-A-DOC.
The principles of Responsive Web Design are also manifested on the design of our platform. When the screen width is not wide enough for displaying the navigation buttons, the navigation bar will be collapsed into a \( \equiv \) button so that users can click it to toggle a side panel which contains the navigation elements in wider screen sizes.

![Responsive Design manifested](image)

**Figure 18**: Responsive Design manifested by implementing a side navigation panel for smartphone users.

### 3.1.2 User Accounts

There are three major roles that are involved in the usage of the system – client, clinic and admin. Users of the client and clinic roles will be able to register accounts on the platform and perform authentication using their provided credentials so as to access the appointment reservation and medical certification features. Clients registering an account are required to provide personal details such as full name, HKID number, phone number, gender, and date of birth. A password is also required for subsequent authentication purposes. Clinics, on the other hand, are required to provide more details including its branch name, address, opening hours, and the specifics of each doctor that works in the clinic, including the registration number\(^4\). Each doctor is also required to provide a personal PIN for the signature of medical certificates. The upload of a supporting document is mandatory for the review process of the clinic’s registration request, and the clinic

\(^4\) According to Cap. 161 Medical Registration Ordinance of the Hong Kong Legislation, a registration number issued under the law certifies the eligibility of the medical practitioner to legally exercise his/her professional duties in Hong Kong. The number is provided to SPOT-A-DOC upon registration (alongside a supporting document) to facilitate the review process of the account by platform administrators.
account can only be authenticated once an administrator has approved the registration request. This acts as a safeguard against clinics and doctors that do not have legal rights to operate in Hong Kong. As shown in the below figures, there are two separate pages for the client and clinic users to perform registration. The passwords that have to be entered are required to be of at least 8 characters long, and at least one upper case letter, one lowercase letter, and one special letter, resulting in a key space of at least $26+26+10=62$ characters, and a total number of possible password combinations to $62^8$, reducing the risk of advisories breaking into user accounts through brute-force attacks.

**Figure 19: Registration page for Client users.**

**Figure 20: Registration page for Clinic users.**

Users of any roles are not required to provision a username-password tuple for authentication each time an action is performed (or a web request constituting to performing actions for the user is sent). This will be achieved by implementing token-based authentication, and the session of the user will persist as long as the bearer token has not expired, and a manual logout action has not been performed. The tokens will be momentarily placed in the browser Local Storage, and they
will be granted upon success of username-password authentication. Besides access tokens (stored in the variable `access_token`), which will only be valid for 15 minutes, there is a refresh token (stored in the variable `refresh_token`) that will be valid for 24 hours such that a new access token can be granted for this specific user when the old ones are expired, and the user will be required to reauthenticate only after 24 hours of provisioning his username-password tuple. This mechanism allows the access token which provides direct access to the user features to have a short lifespan, while allowing the user to stay logged in and not having to type in his password every 30 minutes.

![Figure 21: The access_token and refresh_token stored in Local Storage after authentication.](image)

The Login page is universal across user roles, and clients and clinics will be required to select their role and then input their username and password for authentication. The below figure shows how the Login page looks like.

![Figure 22: Login Page of SPOT-A-DOC.](image)
Like registration, client accounts will be allowed to modify their credentials whenever desired, while clinic accounts are required to submit their credential-change requests through the account management page alongside a supporting document. The changes will not take effect until an approval is explicitly granted by an administrator.

Table 3: A pivot-table summarizing the characteristics of how accounts of different roles are handled.

<table>
<thead>
<tr>
<th>Role \ Action</th>
<th>Registration</th>
<th>Modify Account Details</th>
<th>Login</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>Takes effect immediately afterwards.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If a clinic registration request has not yet been approved, the user will be able to click into the Check Registration Status page and enter the username-password tuple provided during registration. The status of the registration request, either pending, approved, or disapproved, will be displayed alongside the information the user provided during registration. The reasons for disapproval will also be returned to the clinic registrant if the request is rejected by an administrator so that the prospective clinic knows what has gone wrong.

Figure 23: The Check Registration Status page for clinic users with not yet approved registration requests to check request status.
3.1.3 Clinic/Doctor Listings and Reviews

The clinics and their corresponding doctors will be visible to the public once the clinic account is approved. In SPOT-A-DOC, there will be two pages purposed for browsing the list of available doctors and clinics respectively. Each entry in the doctors and the clinics list contains an overview of its information, including its branch name, doctor name, and whether the clinic is currently open or closed. These entries will be clickable, and a more detailed pane of clinic/doctor information and the actions that the user can perform against this clinic/doctor will be available.

Moreover, the listings contain not only the practitioners registered through our system, but also the ones scrapped from other online platforms\(^5\), such as the official website of the Medical Council.

\(^5\) In our final implementation, the platform already functionally allows scrapped clinics and doctors to be populated into our database and be queried through backend endpoints. However, due to time constraints, regulations and compliance of the data and the complexity in data cleansing, which are not the focus of this project, the ETL (Extract-Transform-Load) process for this is not implemented. Mock data that represent scrapped clinics and doctors have been involved in the testing and demonstration processes.
of Hong Kong. Online reservation of these scrapped doctors will not be permitted since these scrapped clinics are not bound to a clinic user account. However, the clinic can one day decide to join the platform and claim the ownership of the scrapped clinic record on the system through registration⁶.

A prospective clinic will be able to perform registration using the information already existed on the platform by selecting from a list of scrapped clinics. The user will then be redirected to the registration page with the form populated with scrapped clinics so that modification to the information prior to submission of the registration request will be possible.

⁶ A prospective clinic will be able to perform registration using the information already existed on the platform by selecting from a list of scrapped clinics. The user will then be redirected to the registration page with the form populated with scrapped clinics so that modification to the information prior to submission of the registration request will be possible.
Figure 26: The Clinics Listing page with entries showing the doctors that are available on the platform.

The figure above respectively shows the Doctor Listing page and the Clinic Listing page. We can see that there is a Search and Filter function available at the top of both listing pages, and these features allow users to narrow the listing page results to those they desire. Specifically, the filter function is capable of filtering the clinic and doctor results by District, Specialty, Opening Status, Accept Health Care Voucher and Allow Online Booking. On top of that, the Search function performs search on the doctor and clinic names, as well as some of the data attributes such as their address and district. These capabilities are paramount in an e-commerce platform like ours, where it would be desirable that our potential clients (prospective patients) are able to conveniently find
a medical center or a practitioner that fits their needs, resulting in sales conversion (successful appointment reservation). The figures below demonstrate the filter and search functions.

Figure 27: The District attribute is selected for filtering. A separate field selector pops up to the right and allows users to choose from the available options, such as "Central and Western" in this case. The FILTER button to the right shall be clicked next.

Figure 28: The result of the above filtering action. Since this is a Single-Page-Application (SPA), the results are generated immediately without the need for page reloads. As shown, all clinics are in Central and Western.

After obtaining the search results, the user may clear the results using the button CLEAR SEARCH/FILTER. This resets the listings, and the user can continue browsing the listings without any filtering applied.
On the other hand, there have been personalization efforts put in the sorting of the doctor and clinic listings. Since client users can flag a doctor as one of his favorite doctors, these doctors will appear at the top of the list in the search results. Then, the subsequent non-favorite doctors will be sorted in descending order of the ratings. Such sorting scheme allows more relevant doctors and doctors that will more likely attract users to appear earlier\(^7\), thus making it more convenient for prospective patients to seek doctors that they will probably like. As shown in the figure below, this sorting scheme has been applied to the listings.

\hspace{1cm} Figure 29: Sorting of the listings when the user has "favorited" a doctor, and the ratings sorted in descending order.

\(^7\) It would be worth admitting that this may result in an effect like the “Content Bubble” effect, where doctors with higher ratings will always win the game, and those with lower ratings will be at the bottom, and it will be much harder for them to get back to the top. This is conceptually solvable by implementing search results and filtering based on more advanced methodologies such as Machine Learning algorithms that conduct analytics per user. However, due to time constraint, this is not implemented, as stated in the Methodology section, but it is definitely achievable in our platform in the future beyond this final year project.
When an entry is clicked into, a page showing the complete detail of the clinic or doctor will be displayed. In particular, in the Doctor Detail page, client users will be able to leave comments and ratings as Reviews. The ratings will be a number between 1 and 5, where the default ratings of a clinic will be 0. The overall rating of a clinic is calculated by the average of all the ratings belonging to its doctors. To prevent some particular client users from abusing this feature to “push” a clinic or doctor to the top of the rank, each client user will only be allowed to leave one review, and leaving a new review will override the previous one. Moreover, if the doctor belongs to a clinic which already is an approved registered clinic account in the system, there will be a “SCHEDULE AN APPOINTMENT” button, where clicking into it allows accessing the appointment reservation features of this doctor. Partial details of the clinic the doctor belongs to will also be shown next to the doctor’s details. The “LOOK FOR MORE DETAILS” button directs the user to a Clinic Detail page.

Figure 30: The Doctor Detail page of the doctor Man Yuek-Tsin.
3.1.4 Appointment Management

When a client user wishes to make an appointment, the “SCHEDULE AN APPOINTMENT” button can be clicked on the Doctor Detail page. The Schedule Booking Now page will be displayed, and users can choose the symptoms to be reported to the doctor, and the date of the appointment. Multiple symptoms can be selected from the list of predefined symptoms. Since the doctor may not be present in the clinic every moment, and the clinic may not be open 24/7 every day, the date selector will only show dates when the doctor will have consultation sessions available for booking. After selecting a date, the available timeslots will be displayed for selection. The timeslots that are allowed for booking will be 5 minutes per session in our implementation. It is conceptually possible that the partition of the timeslot is variable and be configured by the clinic user in the future. After selecting the parameters for the consultation session, the “CONFIRM BOOKING” button can be clicked, and the session will be visible to the clinic user.
Figure 32: Schedule Booking Now page.

Figure 33: The state when the parameters have been selected, and the "CONFIRM BOOKING" button is ready to be clicked.
After booking, the client will be redirected to the Booking History page. As shown in the below figure, the appointment that has just been reserved will be displayed in the calendar view of the client’s bookings. This view will display all the bookings, including those that have been completed and those that are in the future. The entries in the calendar are clickable, and the user will be redirected to a page that shows the details of this booking and a “CANCEL BOOKING” button. After the booking is cancelled, the little blue indicator in the calendar view will be changed to orange, and the timeslot that was originally occupied by the user will be released for others to book. A booking can only be cancelled 2 hours before the consultation time.

Figure 34: Calendar view of the Booking History page. The bookings that will happen in the future are indicated by a blue dot.
In terms of the clinic accounts, after a client has made a booking on a doctor, the corresponding clinic user (whom the doctor belongs to) will be able to view the new appointment in the Booking Record section of its account. Since a clinic can have a large number of appointments a day, it would be somewhat impractical to show the calendar view packed with appointments, and the box representing a day in the calendar view would be too tiny to show more than 3 appointments. Therefore, the List view will be set as the default appearance of the Booking Record page for clinic accounts.
Figure 37: The Booking Record page shown to clinic accounts.

Figure 38: The Booking Detail page shown to doctors. Since this appointment is not cancelled by the client, the doctor will be able to mark it as "Attended" or "Absent".
When the clinic user clicks into the entry representing an appointment on the Booking Record page, a Booking Detail page will be displayed, and the doctor will be able to read all information associated with the client whom he is going to meet. The doctor will also be able to mark the appointment as “Attended” or “Absent” after the appointment consultation time. If “Attended” is selected, the doctor will be able to provide the fee of this consultation, the diagnosis, and the days of absence that the patient is granted. The passcode for this doctor has to be provided in order for the appointment to be marked. The moment when the “SAVE” button is clicked, the medical certificate for this appointment session is also signed, and the client user will be able to view and download the PDF medical certificate. The appointment is not editable by either the doctor or the client after any changes made to it, and this will also make sure that the medical certificates issued will not be invalidated due to changes to the appointment data.

Figure 39: Doctor marking an appointment as "Attended". The four extra fields for the doctor to input will be displayed after selecting the status.
The generated PDF medical certificate contains the client’s information, including his/her full name, HKID, diagnosis, number of days issued for his sick leave, and the doctor’s signature. Here, since public key digital signature has been implemented (based on the flow mentioned in the Methodology section), the medical certificate will not contain any hand-signed signature of the doctor, but instead, a sentence reminding that the certificate is digitally signed. The digital signature string is printed in the bottom of the PDF so that it can be copied to the verification system for verification, and displaying it also makes this medical certificate look more trustable. Users possessing a medical certificate can also scan the QR code on the PDF or the printed version of the cert to verify whether the exact corresponding record in the system really exists.

![Figure 40: Medical Certificate issued via SPOT-A-DOC. The QR Code printed can be scanned, and it redirects the user to the medical certificate verification system for this medical certificate.](image-url)
When the QR code on a medical certificate is scanned, or the digital signature string is inputted into the Medical Certificate Verification System on our platform, the system will try to verify the signature by performing business logic in the backend. If the certificate is successfully verified, the message “The medical certificate is successfully verified” and a green tick will be displayed to indicate that there indeed exists such a “Attended” appointment record in the system. The appointment data associated will also be displayed together with the client’s personal credentials for manual comparison and verification. The HKID value will be partially hidden for the protection of client users’ personal privacy. If a wrong or arbitrary digital signature is provided, or there is no record in the system that can verify the digital signature, then the page will display the “No such record” message together with a red cross.

![Figure 41: Successful verification of the digital signature.](image-url)
3.1.5 Health Articles

Articles related to health care will be published by platform administrators so that the platform will not only contain its major appointment reservation features, but also promote health care such that users may visit the site regularly to browse health-care information. This helps turning SPOT-A-DOC to an all-round e-commerce platform, and there will be users willing to get onto the platform regularly, and this is beneficial to the platform in terms of prolonged business presence.
Health Articles

Give babies peanut butter to cut allergy by 77%, study says
James Gallagher

Giving young babies - between four and six months old - tiny tastes of smooth peanut butter could dramatically cut peanut allergies, say scientists. Research shows there is a crucial opportunity during weaning to cut allergy cases by 77%. They say the government's advice...

The Life-Changing Importance of a Good Night Sleep
Adam Smith

Getting high-quality sleep each night is one of the single most important things we can do for our health, well-being, and success in life. Insufficient sleep is associated with a dramatically increased risk of heart disease, kidney disease, high blood pressure, diabetes, stroke, and...

Why we have nightmares and how to stop them
Kristen Rogers, CNN

We leave behind our fears of monsters under the bed as we say goodbye to our childhoods, but one can follow us into adulthood and loom over our heads. Nightmares are more common in childhood, but anywhere from 50% to 65% of adults report having occasional...

3.1.6 Admin Panel

Administrators on the system have their own version of the platform to browse with and perform operation matters of the platform. The navigation bar of the admin version is in brown, making it distinctive to the normal version where the business and customer sides of the B2C platform visit. The admin panel is only accessible via the path /admin, and an authentication process is required.
In the admin panel, administrators will be permitted to add and delete health articles. They will also be responsible for approving and disapproving clinic registration and account modification requests. There are a total of three sections in the admin panel:

Table 4: Description of the purposes and functionalities of the admin panel sections.

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Clinics</td>
<td>To manage and view all clinics on the platform, including those that are scrapped and also those registered on the platform. An EXPORT button is also available for the admin to export the clinic list.</td>
</tr>
<tr>
<td>Clinic Requests</td>
<td>To manage and view all clinic requests that are submitted by clinic users. The requests can be approved or disapproved, and including a reason for disapproval is mandatory upon disapproval of a request.</td>
</tr>
<tr>
<td>Articles</td>
<td>To manage and view all health articles on the platform by adding, modifying and deleting records.</td>
</tr>
</tbody>
</table>

![Figure 46: The All-Clinics section in the Admin Panel.](image-url)
Figure 47: The Clinic Requests section in Admin Panel. The state of each clinic request is also displayed alongside each request entry.

Figure 48: The All-Articles section in the Admin Panel. There are buttons in each entry that lets admins perform certain actions including View, Modify and Delete.
3.2 Software Implementation

The technologies that were proposed in the Methodology section have been utilized for the implementation of SPOT-A-DOC. This section showcases the realization of the platform by providing screenshots together with their explanations. Due to the huge size and quantity of the completed implementations, selected important ideas involved will be discussed. The set-ups and configurations of the systems involved will also be covered.

3.2.1 Domain Name Service Configurations

The domain name “fyp22008.online” has been acquired, and the required host records for DNS routing have been configured as well. An “A Record” at the route “@” subdomain has been configured to point to an online service that redirects “fyp22008.online” requests, together with the entire URL path, to www.fyp22008.online, where the platform’s frontend is really hosted at. Four other “CNAME Records” have been configured to point to the corresponding servers where the applications are hosted. The www subdomain points to the frontend application, the api subdomain points to a proxy server of the test environment, the backend subdomain points to the production backend server, the backend-staging subdomain points to the test backend server, and the postman subdomain points to the postman server for API testing purposes.

![Figure 49: Configurations of DNS for fyp22008.online](image-url)
3.2.2 Frontend Implementation

The user interface and the perceived functionalities are implemented as a React application using mainly the MUI addon for pre-built UI components. This application is deployed to Heroku and serves as the Presentation Tier of the system. The Axios library has been adopted for making and handling API requests to and from the backend server. As shown in the figure below, the React application has adopted a modular approach and an alignment to the separation of concerns principle. The React components of individual pages have been placed in subfolders of the “pages” directory, making the entire code structure easier to manage. Moreover, the “list” directory houses JSON files that store a list of predefined values for category (of doctors, such as General Practice), district (such as Central and Western), and symptom (such as “Ankle Pain”). This serves as the source of truth in the entire project and reduces the risk of inconsistencies across different portions of the code.

![Figure 50: Part of the entire file hierarchy of the frontend React application. The separation of concerns principle is employed in its structure.](image-url)
The React application is not run directly on the Node runtime environment on the Heroku service. It is compiled using the “npm run build” command, where the application is converted into a static HTML project with Vanilla JavaScript that can be executed on any web servers. The application in this format is ready for production, and it is configured to be served by an Express application running on the Presentation Tier server. One point to note is that since React is a Single Page Application (SPA), all actions performed on the interface are handled by React Routers that are processed in the client side (instead of routed via the web server by performing server-side rendering). This would be okay if all users launch the platform first on the main page by typing www.fyp22008.online, but the application will fail to load and result in 404 error returned by the web server if a specific path is typed in the URL bar and navigated to. For example, navigating to www.fyp22008.online/clinic via the address bar will not work, but if we click on the “Clinics” tab in the navigation bar of the platform, the browser will be “redirected” to the page with the address bar content changed, while it is actually only the JavaScript in the DOM modifying the user’s perceived address bar. This is addressed by redirecting all web requests to index.html so that user will be routed to the designated page through the React Routers in the home page.

```javascript
const express = require("express")
const path = require("path")

const app = express()

app.use(express.static(path.join(__dirname, "build")))

app.get("*, (req, res) => {
    res.sendFile(path.join(__dirname, "build/index.html"))
})
```

*Figure 51: Code in the Express application (at the Presentation Tier) responsible for routing browser URL requests to index.html.*

Moreover, the concept of Protected Routes in React have been employed to partition pages by user roles. Since all users accessing restricted pages, such as the Account Page, must be of a specific user role, such as “clinic”, and that the role value is returned from the backend and also available in the access token (which is a JSON Web Token), the React application uses this value to check whether a user is permitted to access a certain page. Although the role value is modifiable in the
Local Storage of the browser to bypass Protected Route restrictions, the user will not be able to perform any action because the backend will also verify the role of the user.

![Figure 52: ProtectedRoute applied in the React application to control user access to specific groups of pages.](image)

### 3.2.3 Backend Implementation

The Backend is implemented using Express and is run on the Node Runtime Environment on the backend server. All the components involved in the backend have two versions deployed, and they respectively serve testing and production purposes. The backend version for testing purpose connects to the MongoDB test database, and the server is deployed to a Heroku Dyno. The one for production connects to the spot-a-doc-production database, and the server is deployed to another isolated Heroku Dyno as well. Both databases, on the other hand, are deployed to the same MongoDB cluster. These ensure that the testing processes during development will not contaminate the production environment or pollute the clean data there.

There are a total of 37 backend API endpoints implemented to serve frontend requests. The endpoints can be categorized by feature, and a considerable portion of the endpoints are RESTful.
The table below lists all the developed endpoints, grouped by the corresponding features and subitems.

*Table 5: List of API endpoints developed in the backend as the Application/Logic Tier.*

<table>
<thead>
<tr>
<th>Feature</th>
<th>Subitem</th>
<th>API Endpoint</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authentication</strong></td>
<td>Login</td>
<td>/auth/login (POST)</td>
</tr>
<tr>
<td></td>
<td>Registration</td>
<td>/auth/register (POST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/auth/unregister (DELETE)</td>
</tr>
<tr>
<td></td>
<td>Logout</td>
<td>/auth/logout (GET)</td>
</tr>
<tr>
<td></td>
<td>Refresh session</td>
<td>/auth/refresh (POST)</td>
</tr>
<tr>
<td></td>
<td>Check session</td>
<td>/auth/checkToken (GET)</td>
</tr>
<tr>
<td><strong>Admin Operation</strong></td>
<td>Clinic</td>
<td>/api/admin/clinics (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/admin/clinic/&lt;cid&gt; (GET)</td>
</tr>
<tr>
<td></td>
<td>Request</td>
<td>/api/admin/requests (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/admin/request/&lt;rid&gt; (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/admin/request/&lt;rid&gt;/[approve</td>
</tr>
<tr>
<td></td>
<td>Article</td>
<td>/api/admin/article (POST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/admin/article (PUT)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/admin/article (DELETE)</td>
</tr>
<tr>
<td><strong>Client Operation</strong></td>
<td>Account information</td>
<td>/api/client/account (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/client/account (PUT)</td>
</tr>
<tr>
<td></td>
<td>Favourite doctor</td>
<td>/api/client/fav (POST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/client/fav (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/client/fav (DELETE)</td>
</tr>
<tr>
<td></td>
<td>Review</td>
<td>/api/client/review (POST)</td>
</tr>
<tr>
<td></td>
<td>Booking History</td>
<td>/api/client/bookings (POST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/client/book (POST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/client/cancelBooking (PUT)</td>
</tr>
<tr>
<td><strong>Clinic Operation</strong></td>
<td>Account information</td>
<td>/api/clinic/account (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/clinic/account (PUT)</td>
</tr>
<tr>
<td></td>
<td>Request History</td>
<td>/api/clinic/requests (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/clinic/request/&lt;rid&gt; (GET)</td>
</tr>
<tr>
<td></td>
<td>Booking Record</td>
<td>/api/clinic/bookings (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/clinic/booking/&lt;aid&gt; (GET)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/clinic/modifyBooking/&lt;aid&gt; (PUT)</td>
</tr>
<tr>
<td><strong>Browsing</strong></td>
<td>Clinic</td>
<td>/api/browse/clinics (POST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/api/browse/clinic/&lt;cid&gt; (POST)</td>
</tr>
<tr>
<td></td>
<td>Doctor</td>
<td>/api/browse/doctors (POST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/api/browse/doctor/&lt;did&gt; (POST)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/api/browse/doctor/&lt;did&gt;/availability (GET)</td>
<td></td>
</tr>
<tr>
<td>Article</td>
<td>/api/articles (GET)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/api/articles/&lt;aid&gt; (GET)</td>
<td></td>
</tr>
<tr>
<td>Medical Certificate Verification</td>
<td>Verify certificate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/verifycert/&lt;aid&gt;/&lt;token&gt; (GET)</td>
<td></td>
</tr>
<tr>
<td>Others</td>
<td>Check username</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/ulookup (GET)</td>
<td></td>
</tr>
<tr>
<td>Check request status</td>
<td>/api/checkRequestStatus (POST)</td>
<td></td>
</tr>
<tr>
<td>Store public image</td>
<td>/api/static/public/img/&lt;container&gt;/&lt;blob&gt; (GET)</td>
<td></td>
</tr>
<tr>
<td>Store private image</td>
<td>/api/static/private/&lt;blob&gt; (GET)</td>
<td></td>
</tr>
</tbody>
</table>

In terms of the software design of the backend Express application, the principles of separation of concerns, modular design and object-oriented design have all been incorporated to achieve a more maintainable and high-quality piece of software product. From the project folder hierarchy, we can see that our implementation has modularized the code that serve different purposes to specific locations, instead of coupling them all into the Express Router files, where the logic that runs handles the incoming web requests are defined and run. Specifically, in the project folder, the `utils` folder stores helper modules that contain code for handling specific features, such as the `cert.js` module for running medical certification related features. The `db` folder stores the `Models.js` module, which is responsible for defining the MongoDB schemas and to initialize their models through the Mongoose library. The `Entity.js` module incorporates object-oriented programming (OOP) principles in that it serves as an abstract class definition to its child classes – `Client`, `Clinic` and `Admin`. These entities are instantiated whenever a user passes checks in the authentication middleware, and the code related to accessing the database (manipulating the Mongoose models) are performed via these entities instead of directly in the Express Routers. This separates the concerns of different portions of the code and makes the entire software easier to interpret and maintain. However, only the `Client` object is successfully implemented in the Express application due to time constraints, and the logic for other entities are housed in the Express Routers, since they were initially developed directly in the routers, and later started scaling the code base to incorporate OOP. The figure below shows the project hierarchy of JavaScript modules in the Express application.
Moreover, environment variables have been largely adopted in the backend code structure. This reduces risk of code inconsistencies amid code revamps and modifications, and it also helps hiding critical credentials such as database passwords.

In particular, the implementation of medical certifications in the Express application utilizes the node-forge library, which is widely adopted by many enterprises in the software industry for
performing cryptographic operations. It is used in the RSA key generation process, where the PBKDF2 implementation by the library is used for generating the deterministic seed value, and the RSA generateKeyPair() function is used for generating the public and private keys in binary. They are then converted to PEM format using the privateKeyToPem() and publicKeyToPem() methods. On the other hand, the crypto module is used for signing and verifying digital signatures for medical certificates.

![Node-Forgo Used for RSA Key Generation](image1)

**Figure 55: node-forgo used for RSA key generation.**

![Crypto Library Used for Signing and Verifying Medical Certificates](image2)

**Figure 56: crypto library used for signing and verifying medical certificates.**
As described in the Methodology section, this digital signature flow achieves a secure flow of implementing medical certificates using password-based public key cryptography.

### 3.2.4 DevOps and Testing

The Postman software has been utilized for development of unit-tests pipelines, and the integrated use of GitHub Actions and Heroku Webhooks have collectively formed an entire CI/CD pipeline for backend development that ensured the quality of the software and eased testing and feature implementation efforts. As shown in the below figure, which is the Postman application panel, unit-test pipelines have been set up, and each test contains scenarios to be tested, together with set-up and teardown flows as described in the Methodology section. Test scripts with assertions have been created to facilitate automated testing. Environment variables have also been widely utilized when using Postman to ease set-up modification.

![Figure 57: Use of Postman for unit-testing](image)

![Postman application panel showing unit-test pipelines with scenarios for testing and assertions for automated testing.](image)
As shown in the above figure, Postman has been integrated into our GitHub CI/CD workflow, and the test run has been successful in that the 1000 assertions across the entire Postman unit-test pipeline have been passed.

### 3.3 Difficulties Encountered

There are several difficulties encountered during the development of the platform.

The first difficulty was the process of solving the 404 Error issue of the frontend application. As mentioned in the frontend implementation results section, the Single Page Application limitation of React made it seem impossible for the browser’s address bar to reach pages developed other than through the main page. This was unforeseen because the team did not have knowledge about such limitation of React as an SPA library. The team got frustrated as it affected the implementation of medical certificate’s QR code feature. However, through extensive researching of the problem, it was discovered that the routing of all requests to the index.html page via the web server would solve this problem, and the team successfully catered it in the final deliverable.

The second difficulty was the asynchronous progress between frontend and backend development. Since the team did not have sufficient experience in conducting software engineering in teams
such that the work of different components of the software, in our project’s case, backend and frontend, are worked out by different members, the integration process of the two components was hindered by obstacles of asynchronous pace in development. It was later discovered that meetings and better documentations have to be conducted to facilitate communication between team members, and these could make our integration process of the backend and frontend smoother and clearer.

4. **Conclusion**

This section concludes the project targets and the advancements accomplished.

4.1 **Project Summary**

This project sets out to ease the problems encountered when people attempt to schedule bookings with doctors. A comprehensive platform, SPOT-A-DOC, has been implemented to provide a better clinic search experience and shorter booking duration for the public. Health articles and cryptographically verifiable medical certificate services have also been included on the platform to address the public’s concerns.

Literature has been reviewed to facilitate the objective of this project, which is to develop a B2C e-commerce platform for doctor’s appointment reservation, together with the ability for doctors to issue digital medical certificates and also for the public to conduct verification. The major functionalities that were proposed in the project objective have all been realized successfully. The software engineering methodologies, use of DevOps and testing to streamline development process have all been incorporated in the project.
References


Appendix I – UX Wireframe Designs