KOKOMO

INDIANA UNIVERSITY KOKOMO
DEPARTMENT OF COMPUTER SCIENCE & INFORMATICS

Diamond Finish Services Report

Submitted By:
Jackson McCullough, B.S Computer Science, Minor Mathematics
Enrique Lopez, B.S Informatics
Gavin McCrosson, B.S Informatics, Minor Coaching

Course:
System Development & Design (INFO-I451)

Instructor:
Dr. Aktaruzzaman

Kokomo, IN
April 29, 2026

Acknowledgments

This is an acknowledgment for the people that have put in the efforts to finish and create the best project possible with the resources given. Diamond Finish Detailing was originally the one company that we created the website for, but over time the company merged with another smaller company, creating Diamond Finish Services. This is to show Diamond Finish Services appreciation for the opportunity to create a website for them and for all of the help and information given along the way, along with the flexibility they have had when it comes to catering to our schedule. Also, for all of the hard work throughout the entire school year from all of Group Three (Gavin McCrosson, Enrique Lopez, & Jakcson McCullough) to make this project the best it could be. Finally, a great shout out and acknowledgment to Dr. Aktaruzzaman, the professor who helped and taught the class for an entire year to guide all of the students to help create the best project they could.

Abstract

This project focuses on the design and development of a professional website for Diamond Finish Detailing, a car detailing business. Goals of the development of this website include improving customer outreach, service accessibility, and overall brand visibility. Small businesses rely heavily on an online presence to attract and retain customers. The purpose of this project is to create a user friendly, visually appealing, and functional website that effectively represents the services offered while providing a seamless experience for potential clients.

The development process followed a structured approach. Key features were identified, including a service menu, pricing information, appointment booking functionality, customer testimonials, and a photo gallery showcasing completed work. While something that was told to us to leave out that most websites include in this day and age is the ability to order and pay online. The website was then designed utilizing Full-Stack development with an agile approach. Many different programming languages were utilized throughout the development of the website. These languages include HTML, JavaScript, CSS, MySQL, and handlebars was implemented further down the development cycle for efficiency related purposes. Other tools used was Argon2 which is an OWASP endorsed hashing algorithm to increase security throughout the website.

The goal of this project was to create a website for Diamond Finish Detailing. The final product includes the ability to schedule appointments, cost of service information, customer testimonials, photos of previously completed work. The final product works as intended and meets all of its intended objectives. There is room for growth within the project, the main space for growth would be if the client wanted payments to be processed through the project as well. Overall this project demonstrates the successful development of a website for a client.

Keywords: Web Development, Full-Stack, Cybersecurity, Data Integrity, OWASP

Contents

Introduction

System Overview

The system developed in this project for Diamond Finish Services is a web-based application designed for a car detailing company that also provides exterior cleaning and landscaping-related services, such as pressure washing. The platform allows customers to browse available services, including interior & exterior vehicle detailing, driveway & sidewalk pressure washing, and other property cleaning services. Users of the webpage can view service descriptions, pricing, and availability, as well as book appointments directly through the system that we have designed with an easy to follow user interface. Allowing for the user experience to be a seamless transition giving Diamond Finish Services the best possible product for the business administrators.

For these business administrators, the system provides tools to manage service listings, track customer bookings, and organize scheduling efficiently. This system addresses the need for small service-based businesses to have a centralized digital platform that improves customer engagement and efficiency. In today's market, having an accessible online booking system is essential for attracting and retaining customers.

This project demonstrates how modern web technologies can be applied to solve real-world business problems, making it highly relevant for students pursuing a degree in Computer Science or Informatics. With the tools and technology more readily available than ever before it makes it easier for those who want to create a start up. Many smaller businesses simply do not have the financial capacity to invest in professional web development services. Partnering with a recently certified graduate offers a mutually beneficial opportunity, allowing for accessible affordable solutions while enabling new professionals to gain practical experience.

Tools & Technologies

The system was developed using modern web technologies to ensure both functionality and security. The backend was built using Node.js and Express.js, which handle server operations and application logic. The front-end was created using React to provide a responsive and user-friendly interface. MongoDB was used as the database to store and manage application data

efficiently.

In addition to these core technologies, several security measures were implemented to protect the system from common online threats. User inputs are carefully checked and cleaned to prevent harmful code from being injected into the system. Protections are also in place to prevent unauthorized access attempts, such as limiting repeated login or request attempts.

The application includes safeguards against browser-based attacks by using secure configuration settings, and all data is transmitted securely using HTTPS to reduce the risk of interception. Sensitive user information is not exposed in publicly accessible areas of the system to prevent data leaks.

Additionally, protections were added to prevent misuse of system inputs and to limit the size of incoming data, reducing the risk of system overload or abuse. Overall, these tools and security measures help ensure that the system is reliable, secure, and suitable for real-world use. All while helping improve the flow of business for Diamond Finish Services and improving their daily business operations.

Project Objectives

The main objectives of this project are:

- To design and develop a web-based service management system for a car detailing and pressure washing business (DFS).
- To create an easy-to-use interface for customers to explore and book services.
- To implement an efficient scheduling and booking system.
- To allow administrators to manage services, pricing, and customer data.
- To integrate multiple service categories (car detailing and exterior cleaning) into one platform.
- To provide the user of the system with a UI/UX friendly experience.
- To build a responsive and scalable application using modern web technologies.
- To improve overall customer experience and business efficiency.

Methods & Development

Started initially with foundational Entity Relationship diagrams to understand the relationships and flow within the system that we wanted to create. The beginning phase was essentially a trial & error phase of what the company truly wanted, and what group 3 decided would be the best way to go about creating the desired system. An agile methodology was utilized when creating this system. This was the phase that was most of the first semester. Understanding the relationships in the websites, how to efficiently make all of the entities communicate with each other, and more.

Going onto the second phase was the "rough draft" phase. This phase was the longest out of all phases. This phase was when the entirety of the HTML, CSS, and JS was created. Once the code was entirely created from scratch (besides the CSS), modification of the code and the "tailoring" of what we could do better to essentially hone this program to the best it can be took place during this phase.

When it came to implementing security frameworks and modern security architecture was its own phase as well. This phase started with research and gathering of information on proper web security. Once knowledge of what to do was gained, then we had to go out and learn how to physically implement those architectures into the website. Finally, the security measures were implemented into the website and tested for validation that they are actually effective. The following security measures were implemented onto the website:

- **XSS Mitigation:** Implementation of email and website restrictions to prevent Cross-Site Scripting (XSS) attacks.
- **HTTP Header Protection:** Enforcement of header restrictions to prevent unauthorized modification of HTTP headers.
- **Injection Prevention:** Utilization of NoSQL structures and input sanitization to prevent SQL and general data injection vulnerabilities.
- **Helmet Middleware:** Integration of the Helmet.js middleware to provide comprehensive protection against common web vulnerabilities, including:

- Clickjacking protection via frameguard.
- XSS filtering and data injection mitigation.
- **HSTS Enforcement:** Implementation of HTTP Strict Transport Security (HSTS) to prevent protocol downgrading and force secure HTTPS connections.
- **Data Leakage Prevention:** Systematic controls to identify and block potential data leakage vectors across the platform.

Finally, was the refining phase. This was where the website was essentially finished, but a few things had to be added in, such as the TLS tokens, encryption keys, and business confirmation emails. In this phase, we also decided to switch to handlebars and add another two files for the landscaping company that Diamond Finish Services brought on with the company. Although this seems hefty, it was less dense than the phase that was previously talked about.

Results & Discussions

The outcome of the website was an overall success and exceeded the expectations of the company. We left the potential for the website to scale even greater over-time as the company grows.

Some difficulties that arose when it came to making the scheduling system was finding the middle ground of simple yet effective. The dilemma was not being able to really have both. If it was more simple, it seemed useless, if it was too complex, it seemed overwhelming for end-users. Realistically, the process of getting through this was trial and error. We would try to implement some design, and take it to the company to get their thoughts. Once feedback was gathered, we would go back to the drawing board and repeat the process of properly balancing these two traits into the system.

This is not necessarily a "difficulty" within the system, but something that could potentially make it even greater would be if the owner wants to start allowing online payments. It is understandable as to why they would not want online payments right now, but if there is a way for them to figure out on the financial side a way to allow online payments, it could greatly add benefit to the user experience and efficiency not just to the website, but the company as a whole. That is how they could fix the issue.

Result validation was a simple task of just running the final prototype by the owner of the company to see if it was up to par and even exceeded the expectations of the company owner. It does satisfy the requirements that were requested by the company.

The limitations lie within the payment method and scheduling of the system. As of now, the time slots are very strict and quite literally hard-coded into the program. This is of request due to the schedule of the owner and worker, so the timing and availability is very limited and does not allow for variability within the schedule. The payment issue is the limitation that was previously discussed.

Conclusion

This project focused on the development of a functional website for Diamond Finish Services. The final product successfully fulfills its primary objectives by allowing users to schedule appointments, view service pricing, read customer testimonials, and view a gallery of completed work. To protect the application from common security threats, several safeguards were implemented. XSS (script injection) was mitigated using CSP nonces and proper HTML escaping for email input fields. NoSQL injection risks were reduced through mongo-sanitize and strict type and format validation. Brute-force and endpoint abuse were handled using rate limiting, while clickjacking and other browser-based attacks were prevented with Helmet security headers. To protect against MITM and downgrade attacks, HTTPS with TLS support and HSTS redirects were enforced. Data leakage was minimized by removing PII from publicly accessible booking endpoints. Additionally, parameter pollution was addressed using HPP, and large payload DoS attempts were prevented through request size limits.

Overall, the website performs as intended and provides a user-friendly experience that meets the client's needs. While the project is complete in its current scope, there is potential for further improvement. One key area for future development would be the integration of an online payment system to allow customers to securely pay for services directly through the website. Adding more availability with the schedule and giving the customers more autonomy on timing would also improve the website, but that would have to be in synchronization with the company owner and workers with their availability. In conclusion, this project demonstrates the successful design and implementation of a professional client-based website, with a solid

foundation for future enhancements.

Appendix

Website Images

DIAMOND FINISH DETAILING

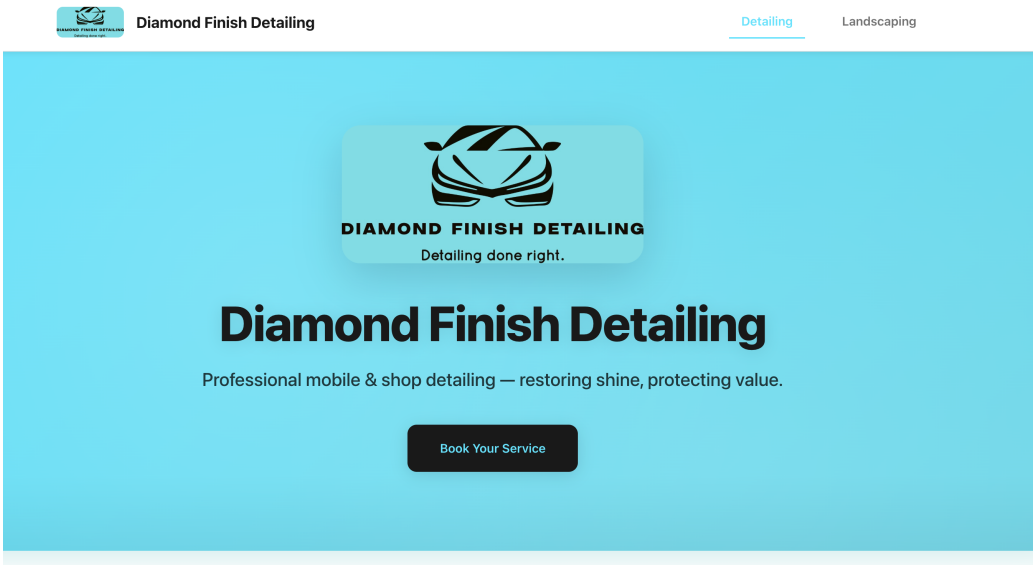

Figure 1: Front page

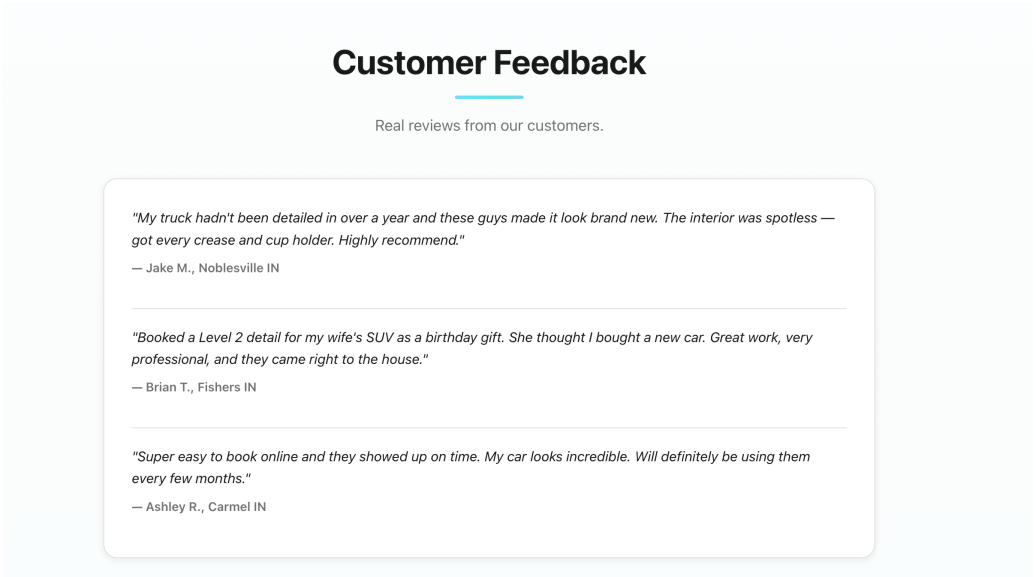

Figure 2: Real feedback from customers on both pages

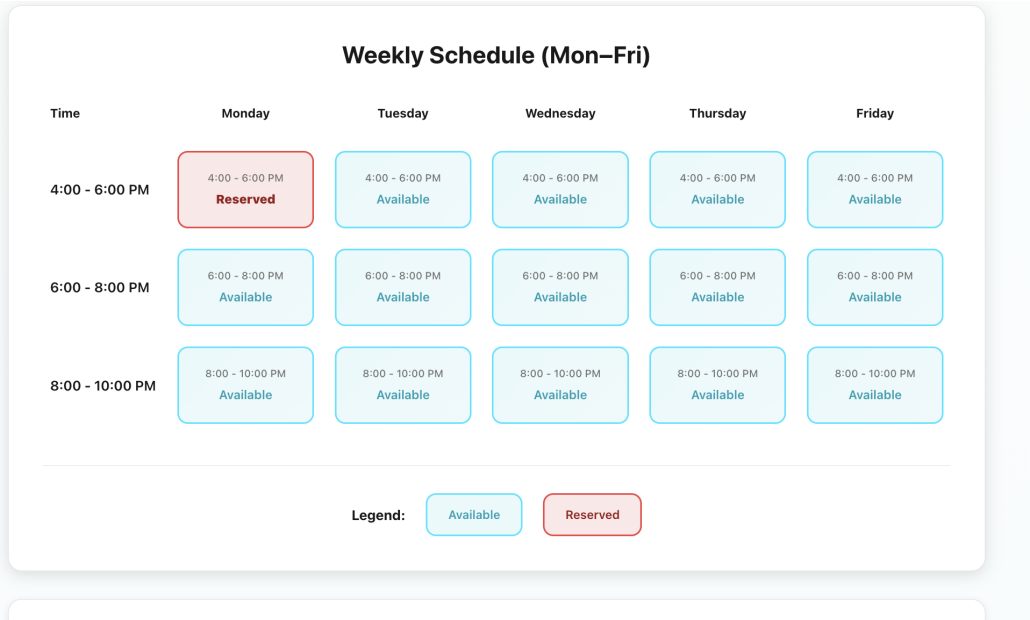

Figure 3: What the reservation system looks like on both pages

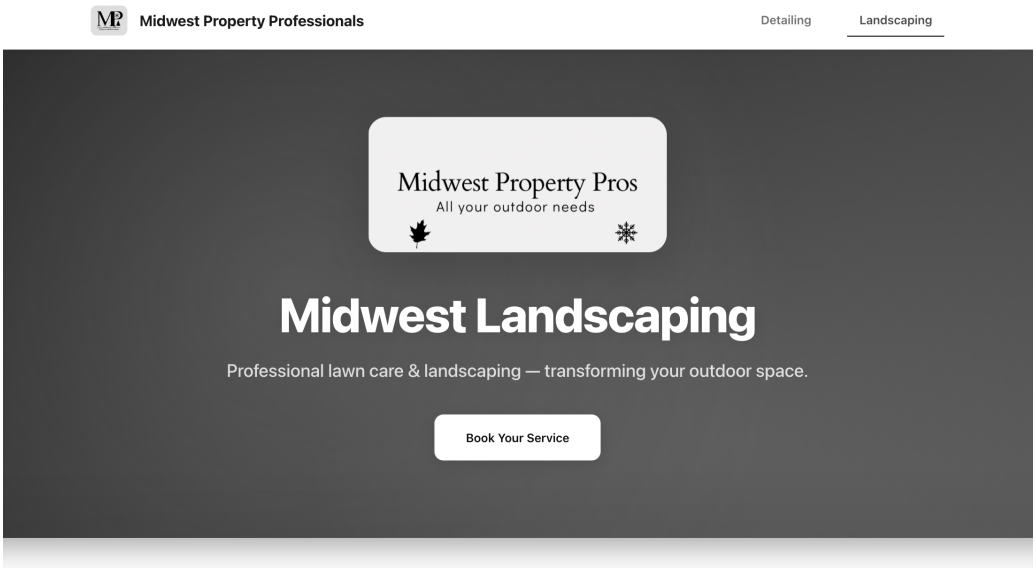

Figure 4: Home Page of the Landscaping company

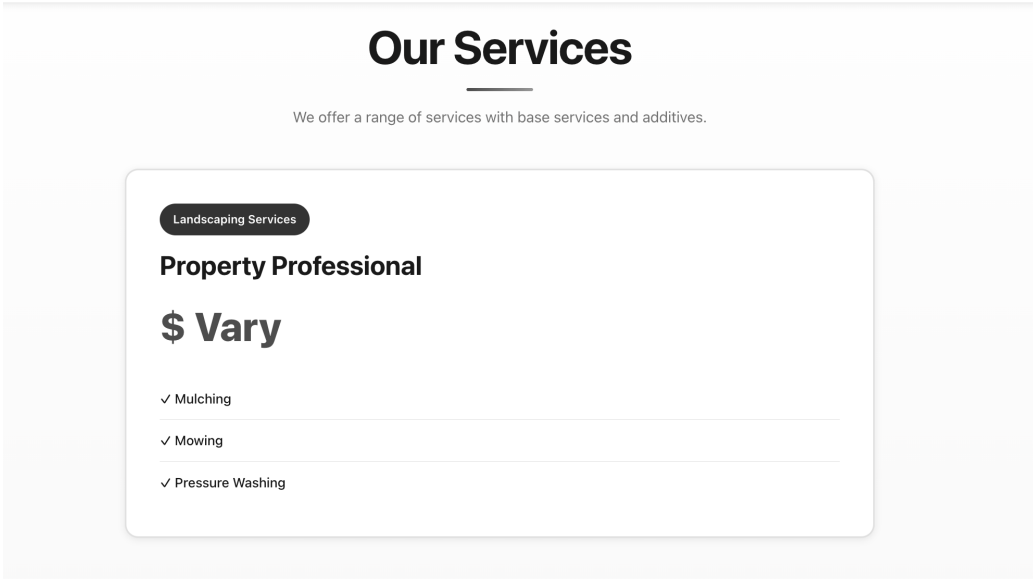

Figure 5: Services labeled for the landscaping page

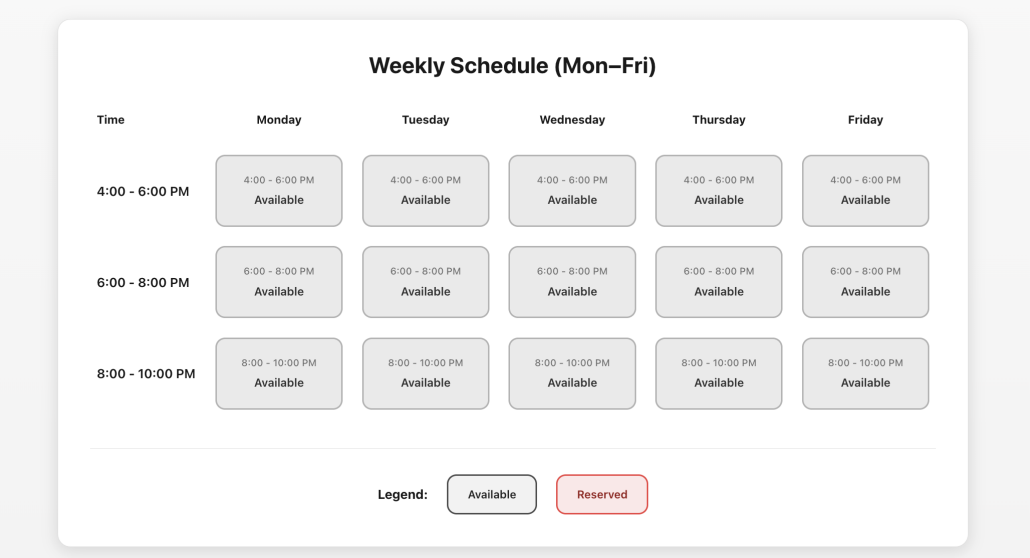

Figure 6: Calendar for landscaping page

REQUESTED INPUT

This section will show all screenshots for the input that the websites ask of the customer.

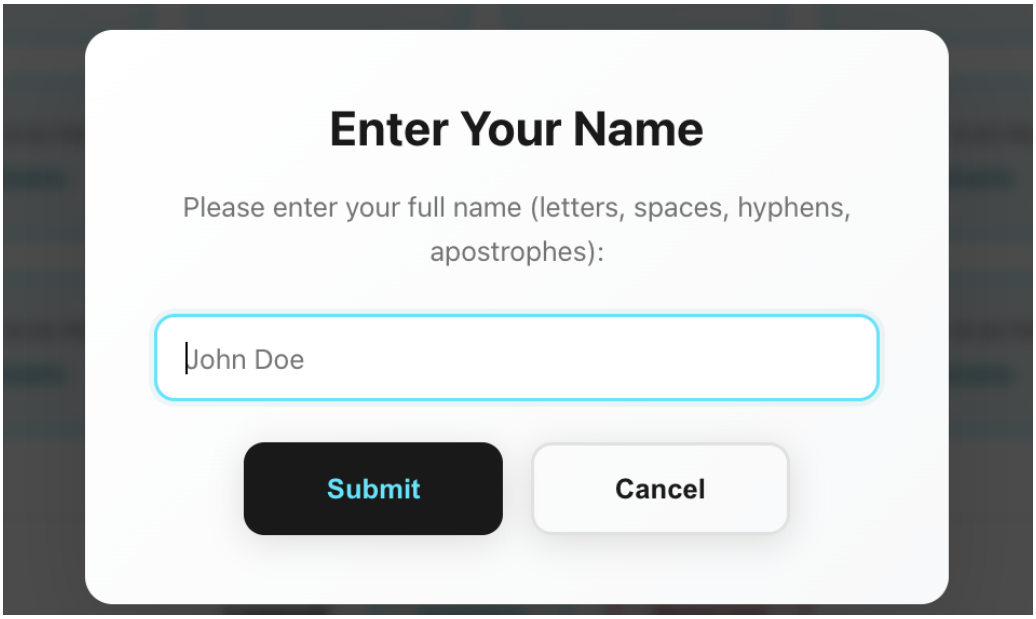

Figure 7: Name Input

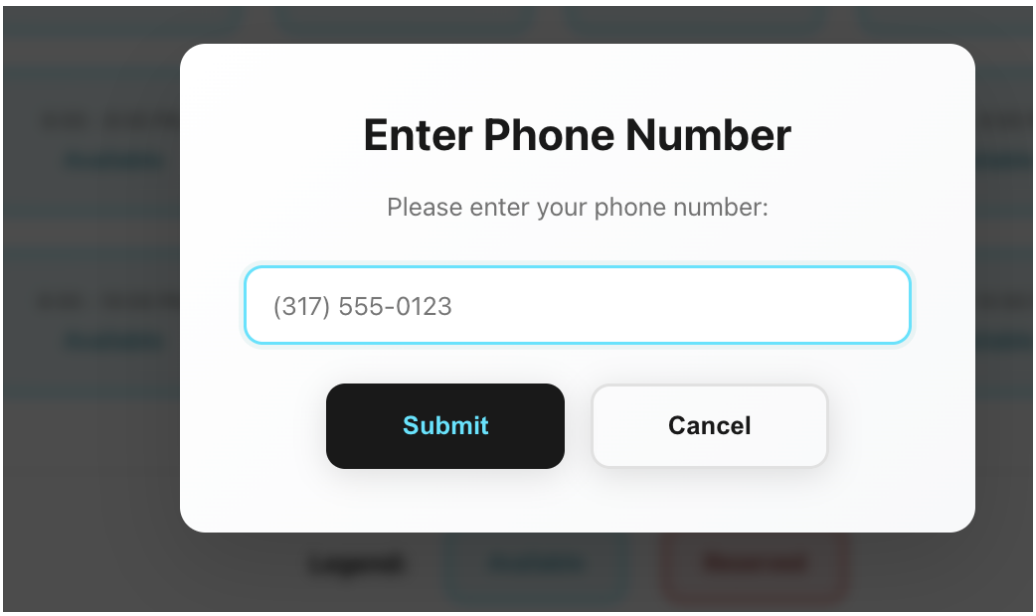

Figure 8: Phone Number input

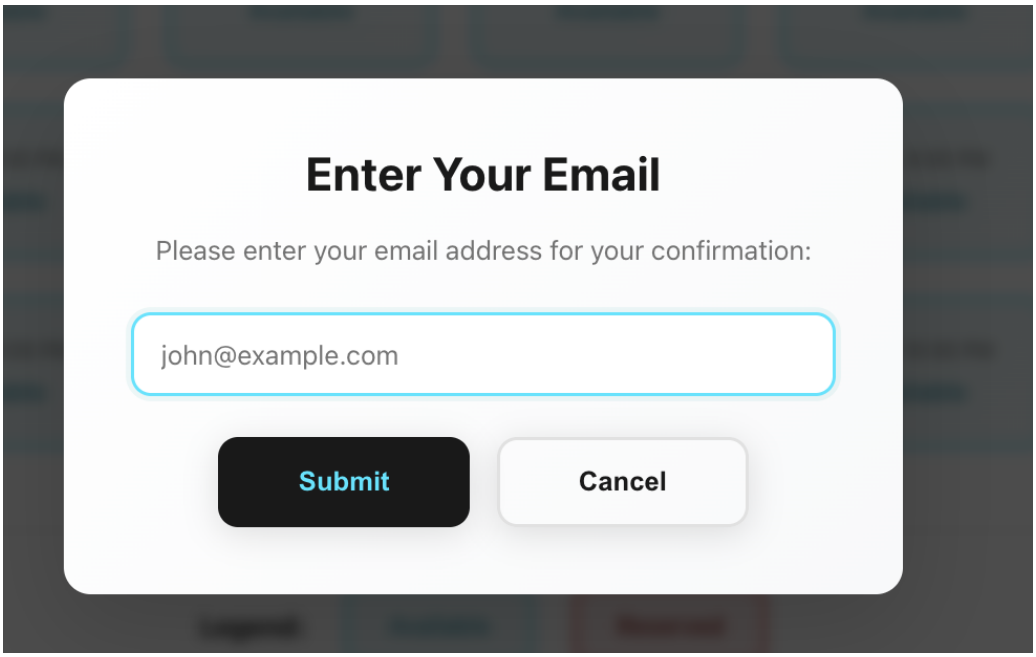

Figure 9: Email input

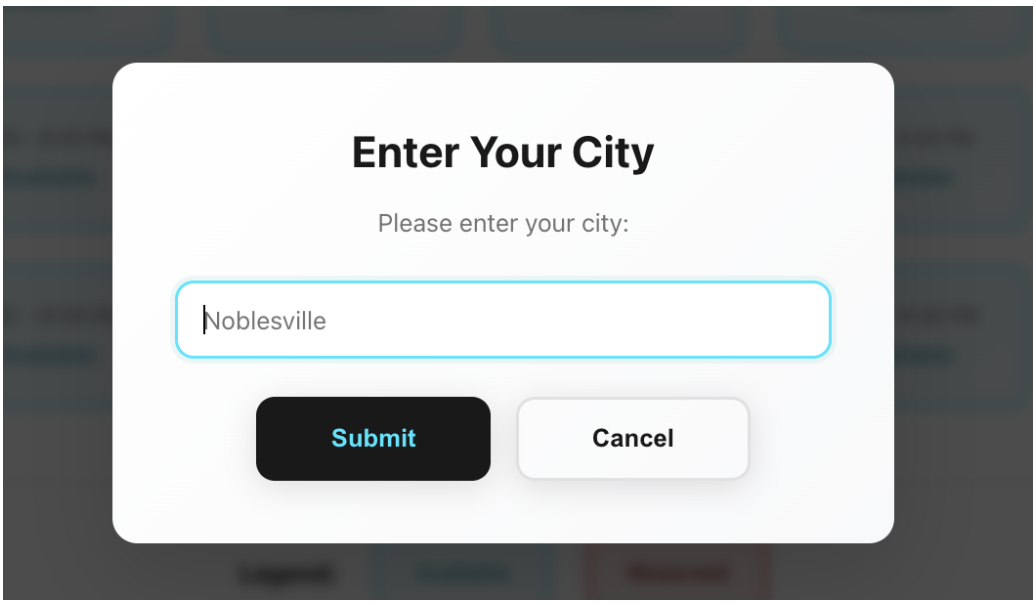

Figure 10: City Input

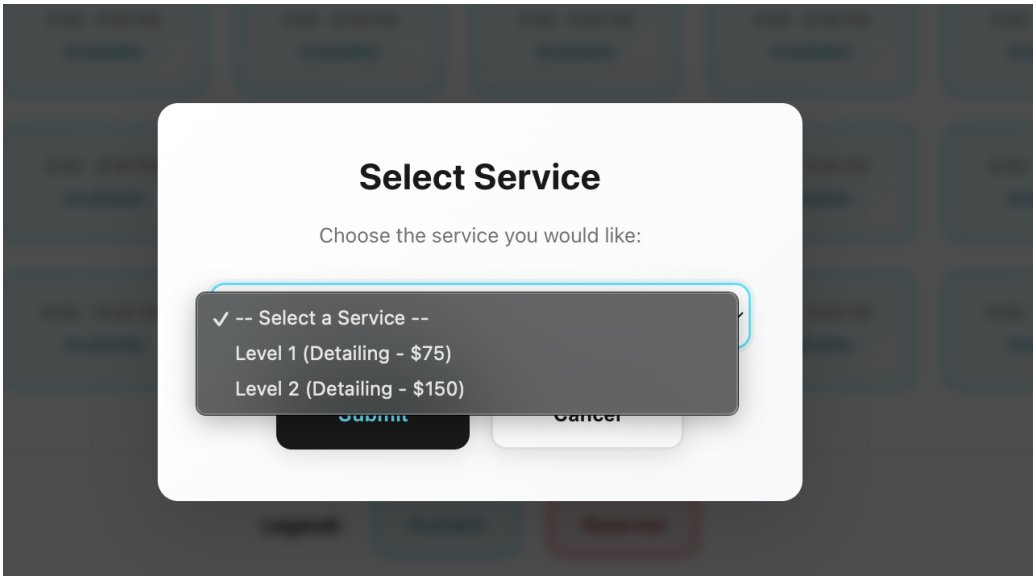

Figure 11: Service Dropdown

SECURITY IMPLEMENTATIONS

```
// HTML escaping — prevents XSS in emails
function escapeHtml(str) {
  return String(str)
    .replace(/&/g, '&amp;')
    .replace(/</g, '&lt;')
    .replace(/>/g, '&gt;')
    .replace(/"/g, '&quot;')
    .replace(/'/g, '&#39;');
}
```

Figure 12: Preventing XSS in email box

```
// ── SECURITY: Strict input validation at system boundary ──
if (!name || !email || !phone || !city || !service || !day || !time) {
  return res.status(400).json({ error: 'All fields are required' });
}
// Type checks — prevents object/array injection
if (typeof name !== 'string' || typeof email !== 'string' || typeof phone !== 'string' ||
    typeof city !== 'string' || typeof service !== 'string' || typeof day !== 'string' ||
    typeof time !== 'string') {
  return res.status(400).json({ error: 'Invalid input types' });
}
// Length limits
if (name.length > 50 || email.length > 254 || phone.length > 20 || city.length > 100) {
  return res.status(400).json({ error: 'Input exceeds maximum length' });
}
// Name format (letters, spaces, hyphens, apostrophes)
if (!/^[A-Za-z][A-Za-z' \-]{0,48}[A-Za-z]$/.test(name.trim())) {
  return res.status(400).json({ error: 'Invalid name format' });
}
// Email format
const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
if (!emailRegex.test(email.trim())) {
  return res.status(400).json({ error: 'Invalid email address' });
```

Figure 13: Regular Expressions for the user input

```
// ── SECURITY: Body size limits — prevents large-payload DoS ──
app.use(express.urlencoded({ extended: true, limit: '10kb' }));
app.use(express.json({ limit: '10kb' }));

// ── SECURITY: NoSQL injection prevention ──
// Strips MongoDB query operators ($gt, $ne, $in, etc.) from user input
app.use(mongoSanitize());

// ── SECURITY: HTTP Parameter Pollution prevention ──
app.use(hpp());

// ── SECURITY: HTTPS redirect in production ──
if (process.env.NODE_ENV === 'production') {
  app.set('trust proxy', 1);
  app.use((req, res, next) => {
    if (req.header('x-forwarded-proto') !== 'https') {
      return res.redirect(301, `https://${req.hostname}${req.url}`);
    }
    next();
  });
}
```

Figure 14: SQL injection & DoS prevention

```
app.get('/api/bookings', async (req, res) => {
  try {
    // ── SECURITY: Strip PII — only expose scheduling data publicly ──
    const bookings = await Booking.find({}, { name: 0, email: 0, phone: 0, city: 0, __v: 0 });
    res.json(bookings);
  } catch (err) {
    res.status(500).json({ error: 'Failed to fetch bookings' });
  }
});
```

Figure 15: Implementing least privilege on who can see the PII

```
// ── SECURITY: Rate limiting — prevents brute-force attacks & DDoS ──
const generalLimiter = rateLimit({
  windowMs: 15 * 60 * 1000, // 15 minutes
  max: 100,
  standardHeaders: true,
  legacyHeaders: false,
  message: { error: 'Too many requests. Please try again later.' },
});
const bookingLimiter = rateLimit({
  windowMs: 15 * 60 * 1000,
  max: 10,
  standardHeaders: true,
  legacyHeaders: false,
  message: { error: 'Too many booking attempts. Please try again later.' },
});
app.use(generalLimiter);
```

Figure 16: Preventing brute force attacks

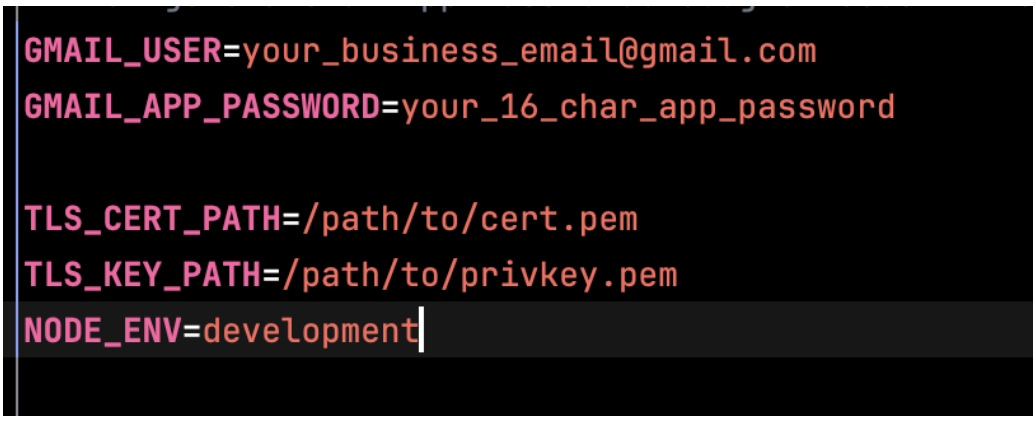

Figure 17: .env file where the TLS tokens and the gmail authentication will take place