Report of Internship at Spintly India Private Ltd.

An Internship Report for

Course code and Course Title: CSA-652 Industry Internship

Credits: 16

Submitted in partial fulfilment of Masters Degree

in MCA

by

HARSHAL SANTOSH GOSAWI

Seat Number : 22P0320021

Roll Number :2206

ABC ID : 697298799070

PRN : 201800649

Under the Mentorship of

RISHU SRIVASTAV

Goa Business School
Discipline of Computer Science & Technology

GOA UNIVERSITY

Date: April 2024

Examined by: Seal of the School/Dept

DECLARATION BY STUDENT

I hereby declare that the data presented in this Internship report entitled, "**Report of Internship at Spintly India Private Ltd.**" is based on the results of investigations carried out by me in the Discipline of computer science & Technology at the Goa business school, Goa University, under the mentorship of **Mr. Rishu Srivastav** and the same has not been submitted elsewhere for the award of a degree or diploma by me. Further, I understand that Goa University or its authorities/College will be not be responsible for the correctness of observations / experimental or other findings given the internship report/work. I hereby authorize the University/college authorities to upload this dissertation on the dissertation repository or anywhere else as the UGC regulations demand and make it available to any one as needed.

Signature and Name of Student
Seat no: 22P0320021

Date: 27/04/2024
Place: Goa University

COMPLETION CERTIFICATE

This is to certify that the internship report "**Report of Internship at Spintly India Private Ltd.**" is a bonafide work carried out by **Mr Harshal Santosh Gosawi** under my mentorship in partial fulfilment of the requirements for the award of the degree of MCA in the Discipline computer science & technology at the Goa business school, Goa University.

(Rishu Srivastav)
Signature and Name of Mentor

Date :

Signature of Dean of School/HoD School/Department Stamp

Date:

Place: Goa University

CONTENTS

Spintly India Private Limited: Shop No.G1, Vaz Apartments, Murida, Fatorda, Goa - 403602

Office: +91 827 502 2406 **GST Number**: 30ABGCS9874E1Z5 **Email**: sales@spintly.com
Sales: +91 876 681 2888 **Regional Sales**: +91 702 219 7893 **Website**: www.spintly.com

Dated: 04th January 2024

To,
Harshal Gosawi,
H.No. 110/E, Near Merces Chapel, Merces,
Vaddem, Vasco-Da-Gama, Goa-403802.

Subject: Promotion and Increase in Monthly Salary

Dear Harshal,

We would like to take this opportunity to congratulate and recognise you for your contributions and thank you for all your efforts, hardwork and dedication.

Effective from January 2024, you have been promoted and your new designation will be "Senior Software Engineer". Additionally, you will receive an increment and your revised monthly Gross Salary (pre-tax) will be ₹40,000.00, the detailed breakup is mentioned below in Annexure A.

You will report directly to Mr. Rishu Srivastav, Lead - Software Delivery.

Please accept this letter as formal notification of your promotion and salary increment. We look forward to your continued commitment and a fulfilling career with Spintly India Private Limited in the years to come.
With best wishes,

Yours Sincerely,

Rohin Parkar

CEO & Managing Director

ANNEXURE A

Salary Components	Per month	Per annum
Basic Salary	₹20,000.00	₹240,000.00
HRA	₹8,000.00	₹96,000.00
Fixed Allowances	₹12,000.00	₹144,000.00
Total Gross Salary	**₹40,000.00**	**₹480,000.00**
Deductions		
PF contribution by employee	₹1,800.00	₹21,600.00
ESI contribution by employee	₹0.00	₹0.00
PT Deduction (if applicable)	₹0.00	₹0.00
Total deductions (PF+ESI)*	**₹1,800.00**	**₹21,600.00**
In Hand Salary (Gross-Total deductions)**	**₹38,200.00**	**₹458,400.00**
CTC Calculation		
Performance variable	₹0.00	₹0.00
Employer PF contribution + PF Admin charges*	₹1,950.00	₹23,400.00
Employer ESI contribution	₹0.00	₹0.00
Total Cost to Company	**₹41,950.00**	**₹503,400.00**

Terms and Conditions:

Confidentiality:
Matter of your compensation is confidential information of the company. Any discussion or disclosure of your compensation with anybody other than your departmental head or HR will be considered a breach of agreement by you. Your compensation package is unique to you and not for comparison with other employees of the company.

*Please note that the above structure is subject to change in lieu of change in company policy and income tax or Government rules.

Spintly India Private Limited: Shop No.G1, Vaz Apartments, Murida, Fatorda, Goa - 403602

Office: +91 827 502 2406 **GST Number**: 30ABGCS9874E1Z5 **Email**: sales@spintly.com
Sales: +91 876 681 2888 **Regional Sales**: +91 702 219 7893 **Website**: www.spintly.com

INTERNSHIP CERTIFICATE

This is to certify that Mr. Harshal Santosh Gosawi, Student of the Goa university, Goa business school, discipline of computer science and technology, undergoing MCA has successfully completed Internship between 01/01/2024 to 31/05/2024 at Spintly India Private Limited. He actively participated in the activities during the period of internship and learned the skills needed for various activities such as Software development (Frontend & Backend Development).

Malcolm Dsouza

CTO

Spintly India Private Limited

Place: Margao

Date: 29/5/2024

ACKNOWLEDGEMENT

I express my sincere gratitude to everyone who has been instrumental in making my internship at Spintly India Private Limited. a valuable and enriching experience.

Firstly, I would like to extend my heartfelt appreciation to Prof. Hanumant Redkar for granting me the opportunity to pursue this internship at the same company where I was employed part-time. Your understanding and support have been invaluable throughout this journey.

I am deeply thankful to Mr. Malcolm Dsouza, the Chief Technology Officer at Spintly, for believing in my capabilities and offering me this incredible opportunity. Your trust and confidence in me have been a source of motivation.

My sincere thanks go to Mr. Rishu Shrivastav, my manager, for his guidance, encouragement, and the fascinating projects he entrusted me with during my internship. Your patience in clearing my doubts and unwavering support have been instrumental in my growth.

I extend my gratitude to Mr. Dominic Barretto, who interviewed me and provided me with the opportunity to join Spintly. Your belief in my potential has been truly inspiring.

I am grateful to Prof. Ramrao Wagh for his mentorship and valuable insights throughout my academic and professional journey.

Furthermore, I would like to thank the entire team at Spintly for their collaboration, support, and for creating a conducive learning environment. Your collective efforts have contributed significantly to my learning and development.

Last but not least, I am thankful to my classmates in college for their encouragement and support.

I am truly fortunate to have had such a supportive network of individuals who have helped shape my internship experience at Spintly. Thank you all for your unwavering support and guidance.

EXECUTIVE SUMMARY

Spintly India Private Limited is leading the charge in transforming the physical security landscape with its cutting-edge fully wireless, cloud-based access control system. This innovative solution streamlines the deployment process, offering installers significant time and cost savings while delivering a frictionless access experience for users. Spintly's overarching mission is to simplify access control and enhance building intelligence through its wireless mesh platform for smart devices.

During my tenure as a Senior Software Engineer, I assumed a pivotal role in various critical projects essential to Spintly's growth and success. These encompassed a spectrum of tasks, including but not limited to, spearheading comprehensive improvements to the Spintly’s Saas platform, developing an intuitive internal support portal, swiftly addressing production bugs to ensure system stability, crafting dynamic email templates to enhance communication efficiency, and architecting a robust call management system for seamless customer interaction. Each task presented unique challenges, from balancing user-centric design considerations to tackling intricate technical issues with agility and precision. Through these endeavors, I not only expanded my technical prowess but also cultivated a deeper understanding of the intricate interplay between software development and business objectives.

My internship at Spintly served as an invaluable learning journey, providing me with exposure to a myriad of advanced concepts and technologies. I delved into the intricacies of React's advanced features, gaining proficiency in building dynamic user interfaces with enhanced functionality. Embracing the microservices architecture, I acquired insights into scalable and resilient software design principles, essential for managing complex systems effectively. Navigating the vast ecosystem of Amazon AWS, I honed my skills in deploying and managing cloud-based infrastructure, ensuring optimal performance and reliability.

Furthermore, my internship afforded me the opportunity to master the nuances of handling date and time in software applications, a critical aspect often overlooked but indispensable for maintaining data integrity and consistency. Embracing the principles of effective version control with Git, I streamlined collaborative development processes, enabling seamless integration and deployment of code changes. Leveraging server-side events, I gained proficiency in implementing real-time communication channels, essential for building responsive and interactive applications.

Additionally, my internship journey encompassed the integration of Kafka for stream processing, empowering me to harness the power of real-time data processing for scalable and resilient applications. Embracing rigorous A/B testing methodologies, I gained insights into data-driven decision-making, enabling me to iterate and optimize software solutions based on empirical evidence and user feedback.

Despite the multifaceted challenges encountered, including the inherent complexity of managing multiple concurrent tasks and the demand for rapid learning and research, my internship at Spintly was a profoundly enriching experience. It equipped me with the adaptability and resilience required to thrive in dynamic work environments. Moving forward, I am confident that the insights gained and skills honed during my tenure at Spintly will continue to serve as a strong foundation for my professional growth and contributions in the field of software engineering.

CHAPTER 01 : ORGANIZATION/COMPANY

1.1 BIRDS-EYE-VIEW

Spintly emerges as a pioneering force in the realm of access control solutions, spearheading a transformative shift towards wireless, BLE-mesh powered technology. As the world's premier provider of truly wireless smart access and attendance management solutions, Spintly stands at the forefront of innovation, redefining the landscape of physical security within buildings.

At the core of Spintly's ethos lies a steadfast commitment to simplifying access control, a mission underscored by the elimination of over 90% of traditional wired infrastructure requirements. This revolutionary approach not only streamlines deployment processes but also translates into substantial savings, with up to 70% reduction in labor and infrastructure costs.

Facilitating seamless, touch-less access through mobile-based solutions, Spintly caters to a diverse spectrum of applications spanning enterprise, hospitality, and residential sectors. By harnessing BLE mesh-enabled platforms, Spintly empowers users with hassle-free, intuitive access to doors, gates, elevators, and other access points, fostering unparalleled convenience and security.

Central to Spintly's vision is the aspiration to foster smarter, simpler built environments, underpinned by frictionless access control experiences. By leveraging fully wireless mesh platforms, Spintly endeavors to augment the intelligence of buildings, heralding a new era of interconnected smart devices and systems.

Dedicated to leading the charge towards smart buildings, Spintly champions innovation through its cutting-edge IoT solutions, reimagining access management and security paradigms. Embracing a holistic approach, Spintly's solutions cater to an array of building typologies, from commercial offices and educational institutions to healthcare facilities and transportation hubs.

Above all, Spintly is driven by a mission to shape the future of smart buildings, enriching lives through safe, sustainable, and convenient living and working environments. By prioritizing the well-being of individuals and communities, Spintly endeavors to create intelligent spaces that enhance quality of life and contribute to a more secure world.

1.2 PRODUCTS/SERVICES

Spintly spearheads a paradigm shift in access control and attendance management, championing innovation and efficiency. With a suite of cutting-edge products, the company redefines conventional systems, making access control simpler and smarter. From the groundbreaking Spintly UNO, the world's first BLE mesh-based access reader, to the versatile Spintly 3-in-1 and Spintly ZEON solutions, each offering reflects Spintly's dedication to leveraging the latest technology to address diverse needs across industries.

Designed for seamless integration and scalability, Spintly's products cater to a wide array of applications, ranging from enterprise and hospitality to residential environments. By eliminating the complexities associated with traditional wired infrastructure, Spintly not only enhances security but also streamlines operations, saving both time and resources for businesses and organizations. With Spintly, access control transcends mere security measures, becoming an integral part of modern building automation and management systems, promising a future of hassle-free, touch-less access management.

1.2.1 Spintly UNO

The Spintly UNO redefines access control with its innovative BLE mesh-based reader. Its sleek and modular design offers a scalable approach to upgrading conventional doors to smart doors. Leveraging BLE mesh technology, it ensures seamless access management while minimizing installation complexities.

1.2.2 Spintly NUOS-1

Tailored for single-door access control needs, the Spintly NUOS-1 boasts OSDP support for enhanced security and reliability. It provides a flexible platform for managing access to premises, offering robust security features and ease of integration.

1.2.3 Spintly Aura

The Aura sets a new standard in access control readers with its BLE-enabled technology. Designed for optimum security and user experience, it allows seamless door unlocking via smartphones or NFC cards. Its advanced features ensure reliable and convenient access management.

1.2.4 Spintly Outdoor

Built to withstand diverse weather conditions, the Spintly Outdoor is an IP-rated BLE mesh access control reader. Its durable and scalable design ensures secure outdoor access management, restricting entry to authorized individuals or vehicles.

1.2.5 Spintly 3-in-1

Combining fingerprint, smartphone, and NFC card authentication methods, the Spintly 3-in-1 access control system offers a comprehensive solution for upgrading regular doors to smart ones. Its modular and scalable design makes it suitable for various access management needs.

1.2.6 Spintly Mesh IO

The Spintly Mesh IO enhances BLE mesh-enabled IoT networks by extending their range and versatility. It facilitates seamless integration with fire alarm panels and remote sensors, expanding the capabilities of BLE mesh communication.

1.2.7 Spintly Fi-B Gateway

As a bridge between BLE and WIFI, the Spintly Fi-B Gateway enables secure communication between Spintly devices and the cloud. Its elegant design and smart features ensure efficient access data storage and management.

1.2.8 Spintly Face Recognition

Smart face identification and authorization, the Spintly Face Recognition system enhances security within the Spintly ecosystem. Its advanced technology offers reliable and efficient access control solutions.

1.2.9 Spintly ZEON

Spintly ZEON combines QR/Barcode scanning with BLE and NFC access, providing a user-friendly and versatile access control solution. Its seamless operation across various access levels makes it ideal for diverse applications.

1.2.10 Spintly HALO

The Spintly HALO offers fingerprint-based access control with BLE-enabled technology. Designed for maximum security and reliability, it ensures seamless door unlocking using fingerprints, smartphones, or NFC cards, enhancing user experience and access control efficiency.

1.2.11 Spintly SAAMs (Smart Access and Attendance Management System)

Spintly SAAMs revolutionizes access control and attendance management with its cloud-based system accessible via a user-friendly software interface and mobile app. Offering a fully cloud-based and wireless solution, SAAMs simplifies security measures across various environments. From basic smart card technology to advanced smartcell-based access, SAAMs provides a range of solutions tailored to diverse needs. Its intuitive software supports multiple credentials, allowing administrators to manage user access efficiently. Integration with fire alarm systems enhances safety, while the Visitor Management System (VMS) tracks daily visits for enhanced security and efficiency. With SAAMs, organizations embrace seamless access control and attendance management, empowering administrators with comprehensive control and insights.

This is the product that I have worked on and is still currently working on, both on the backend services and the frontend side.

1.3 SECTIONS WITHIN THE ORGANIZATION

Within any organization, the distribution of responsibilities and functions across various departments is crucial for its smooth operation. Each department plays a specific role in contributing towards the overall success of the company. In this sub chapter, we will explore the different sections within our organization, categorizing them into technical (tech) and non-technical (non-tech) fields. This classification will help in understanding the diverse nature of tasks performed across various departments. I will provide insights into both tech and non-tech departments, shedding light on their functions and collaborations within the organization.

1.3.1 Technical Departments

a. Firmware Team

The Firmware Team is responsible for designing and developing embedded software that controls the hardware components of our products. They work closely with the Hardware R&D department to ensure compatibility and efficiency. Their tasks include coding, testing, debugging, and optimizing firmware for devices.

b. Hardware R&D

Hardware Research and Development focus on designing and improving the physical components of our products. This includes conceptualizing, prototyping, and testing hardware solutions. Collaboration with the Firmware Team is essential to integrate hardware and software seamlessly.

c. Software Development

The Software Development department, where I work, is responsible for creating and maintaining software applications tailored to meet the needs of our customers. This includes coding, testing, debugging, and implementing new features or enhancements. Collaboration with other departments such as Firmware, Integration, and QA is vital to ensure software compatibility and quality.

d. QA (Quality Assurance)

QA teams are tasked with ensuring the quality of both hardware and software products. They develop and execute test plans, identify bugs or issues, and work closely with development teams to resolve them. Their aim is to deliver high-quality products that meet customer expectations.

1.3.2 Non-Technical Departments

a. Accounts

The Accounts department manages financial transactions, budgeting, invoicing, and payroll. They ensure compliance with accounting standards and regulations, providing financial insights and reports to aid decision-making.

b. Customer Support

Customer Support teams serve as a bridge between the company and its customers. They handle inquiries, troubleshoot issues, and provide assistance to ensure customer satisfaction. Feedback gathered by the Customer Support department is valuable for improving products and services.

c. HR & Admin

The HR & Admin department oversees recruitment, employee relations, training, and administrative tasks. They ensure compliance with labor laws, maintain employee records, and foster a positive work environment.

d. H/W Delivery

Hardware Delivery teams manage the logistics and distribution of physical products. They coordinate with suppliers, warehouses, and shipping carriers to ensure timely delivery of hardware components to customers or other departments.

e. Marketing

Marketing departments are responsible for promoting products and services to attract customers and drive sales. They develop marketing strategies, conduct market research, and execute campaigns across various channels to increase brand visibility and awareness.

f. Product Management

Product Management teams are responsible for defining product strategies, prioritizing features, and overseeing the product lifecycle from conception to launch. They collaborate with various departments to ensure that products meet market demands and align with company goals.

g. Sales

Sales teams focus on acquiring new customers, managing accounts, and generating revenue. They build relationships with clients, negotiate contracts, and strive to meet sales targets set by the company.

h. S/W Project Delivery

Software Project Delivery teams are responsible for managing the delivery of software projects according to agreed-upon timelines and specifications. They coordinate resources, track progress, and communicate with stakeholders to ensure successful project completion.

CHAPTER 02 : TASK(S) HANDLED

During my internship at Spintly, I undertook various responsibilities as a software developer, primarily focusing on web development using React and Node.js technologies.

Throughout the internship, I was actively involved in the development of a call management system, where I contributed to both frontend and backend functionalities. Leveraging React.js for frontend development, I ensured user interfaces were intuitive and responsive, while utilizing Node.js for backend development ensured efficient server-side logic implementation. Integration of real-time communication features enhanced user experience and streamlined call handling processes.

Another significant aspect of my role involved the development of customized email templates to address diverse communication needs within the organization. Employing responsive design principles and Node.js automation, I ensured compatibility across various devices and streamlined email sending processes for improved efficiency.

Additionally, I played a pivotal role in addressing production issues promptly by implementing timely bug fixes. Through thorough debugging and testing, I identified root causes of issues and collaborated with cross-functional teams to prioritize and address critical bugs, ensuring system reliability and performance.

I’ll brifely explain some of the tasks that I worked on during my ineternship period, focusing more on Call management feature, because it was the most interesting thing that I worked on during the internship and also because I was responsible for both frontend and backend development of this service.

2.1 CONTRIBUTION TO THE SaaS PLATFORM

During my internship, I had the opportunity to contribute to the improvement of the company's SaaS platform (website). This involved a variety of tasks aimed at enhancing the website's functionality, user experience, and overall performance.

One aspect of the website improvements was the implementation of new features. Working closely with the design team, we identified areas where the website could be enhanced to better meet the needs of users and align with the company's goals. This included adding new sections, implementing chatbots, and integrating third-party services to provide additional functionality.

In addition to adding new features, we focused on refining existing functionalities to improve the overall user experience. Such as optimizing page load times to ensure fast and seamless navigation, enhancing the website's mobile responsiveness to cater to users accessing the site from various devices.

Furthermore, we paid close attention to the visual design and branding elements of the website. This involved updating the website's design to ensure consistency with the company's brand guidelines, refreshing outdated visuals, and creating visually appealing layouts.

Another important aspect of it was performance optimization. We conducted thorough performance audits to identify any bottlenecks or areas for improvement, such as inefficient code, large image files, or excessive use of scripts. By implementing optimizations such as code minification, image compression, and lazy loading techniques, we were able to significantly improve the website's loading speed and overall performance.

Throughout the process of improving the main website, collaboration and communication were key. I worked closely with cross-functional teams, including designers, developers, and product managers, to gather requirements, brainstorm ideas, and ensure alignment with the company's objectives. Regular updates and feedback sessions helped to keep everyone informed and involved in the process, ensuring that the final product met expectations and delivered value to both the company and its users.

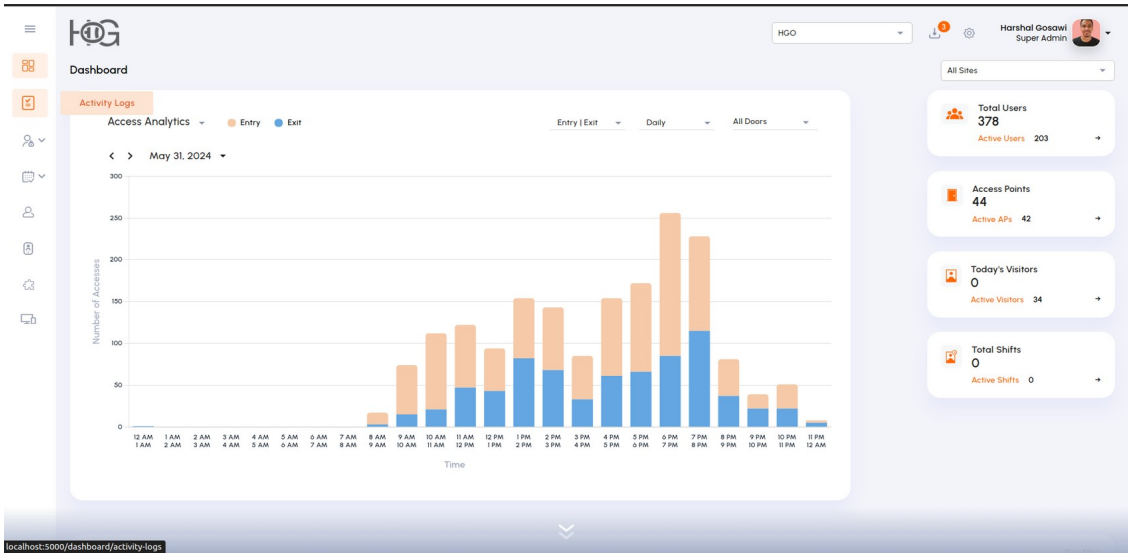


Figure 2.1 Spintly SaaS platform Website

2.2 INTERNAL SUPPORT PORTAL DEVELOPMENT

During my internship I also had the privilege to work on company's Internal Support Portal—a dynamic web application built using React.js for the frontend and Node.js with Express for the backend. Leveraging PostgreSQL and Grafana as backend databases, alongside Kafka for inter-service communication, the portal emerged as a robust platform empowering our customer support and accounts teams with actionable insights and streamlined service management capabilities.

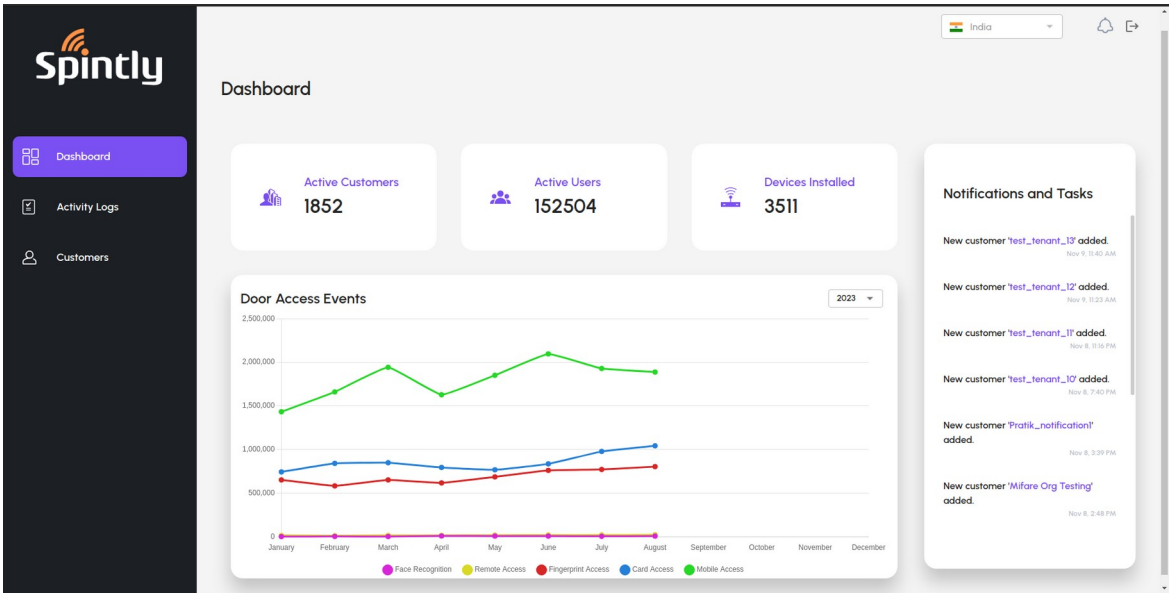


Figure 2.2 Spintly Internal Support Portal

2.2.1 Purpose and Functionality

The primary objective of the Internal Support Portal is to equip support teams with comprehensive insights into our company's customer base. By harnessing data stored in PostgreSQL and Grafana databases, the portal offers real-time visualizations of customer adoption and usage trends over time. This graphical representation enables our teams to make informed decisions and tailor support strategies to better serve our customers' evolving needs.

2.2.2 Features

a. Data Visualization

Utilizing Grafana's powerful visualization capabilities, the portal provides intuitive graphs and charts showcasing key metrics related to customer adoption and usage of our products and features. This allows our teams to identify trends, patterns, and areas for improvement with ease.

b. Service Management

The portal serves as a centralized platform for enabling and disabling various services offered by our company. From attendance management to user, visitor, tenant, and access management, our teams can efficiently manage service configurations based on customer requirements.

c. User-Friendly Interface

Built with React.js, the portal boasts a user-friendly interface designed for seamless navigation and intuitive interaction. This ensures that our teams can access and leverage the portal's functionalities with ease, maximizing productivity and efficiency.

2.2.3 Collaborative Development

Each team member was empowered to take ownership of their respective areas of expertise, whether it was frontend development, backend logic, database management, or UI/UX design. This sense of ownership fostered a greater sense of responsibility and commitment, motivating team members to go above and beyond in delivering high-quality work that exceeded expectations.

2.3 PRODUCTION BUG FIXES

Throughout my internship at Spintly, I was tasked with addressing and resolving various bugs and issues encountered on the SAAMs (Security Access and Attendance Management System) website. As a crucial component of our company's infrastructure, ensuring the stability and reliability of the SAAMs website was paramount to delivering a seamless user experience and meeting the needs of our customers.

2.3.1 Identification and Prioritization

The first step in addressing production bugs was the thorough identification and prioritization of issues. Working closely with the QA team and gathering feedback from end users by analyzing the raised tickets on DevRev (Service we use to raise customer tickets), we categorized bugs based on their severity and impact on user experience. Critical issues affecting core functionalities were prioritized for immediate resolution, while minor bugs were queued for subsequent releases.

2.3.2 Root Cause Analysis

Once bugs were identified and prioritized, the next step involved conducting comprehensive root cause analysis to determine the underlying reasons for their occurrence. This often required diving deep into the codebase, analyzing logs, and tracing the sequence of events leading to the manifestation of the bug. Through meticulous debugging and testing, we were able to pinpoint the root causes and devise effective solutions.

2.3.2 Collaborative Resolution

Addressing production bugs was a collaborative effort involving cross-functional teams, including developers, QA engineers, and product managers. Regular bug examining meetings and communication channels facilitated collaboration and ensured alignment on resolution strategies. By leveraging the diverse expertise of team members, we were able to expedite the resolution process and minimize downtime.

2.3.3 Implementation and Testing

Once solutions were devised, they underwent rigorous testing to ensure their effectiveness and compatibility with existing systems. Unit tests, integration tests, and end-to-end testing were conducted to validate the proposed fixes and prevent regression. Continuous integration and deployment pipelines facilitated seamless deployment of bug fixes, allowing for rapid iteration and rollout of updates.

2.3.4 Monitoring and Feedback

Following the deployment of bug fixes, continuous monitoring and feedback mechanisms were put in place to track the performance of the SAAMS website post-resolution. Monitoring tools and dashboards provided real-time insights into system health and performance metrics, allowing us to proactively identify and address any potential regressions or new issues that may arise.

2.4 EMAIL TEMPLATE DEVELOPMENT

HTML email templates play a crucial role in modern communication strategies, enabling businesses to deliver visually appealing and engaging messages directly to their audience's inbox. During my internship, I had the opportunity to delve into the intricacies of HTML email template development, leveraging AWS SES (Simple Email Service) to handle the actual email sending functionality, .ejs files for templating, and AWS S3 bucket for hosting and accessing images. Below, I'll provide a comprehensive overview of the process, highlighting its advantages, limitations, and important considerations, but before that let’s take a look at an email with and without html email design.

Figure 2.3 Plain Email

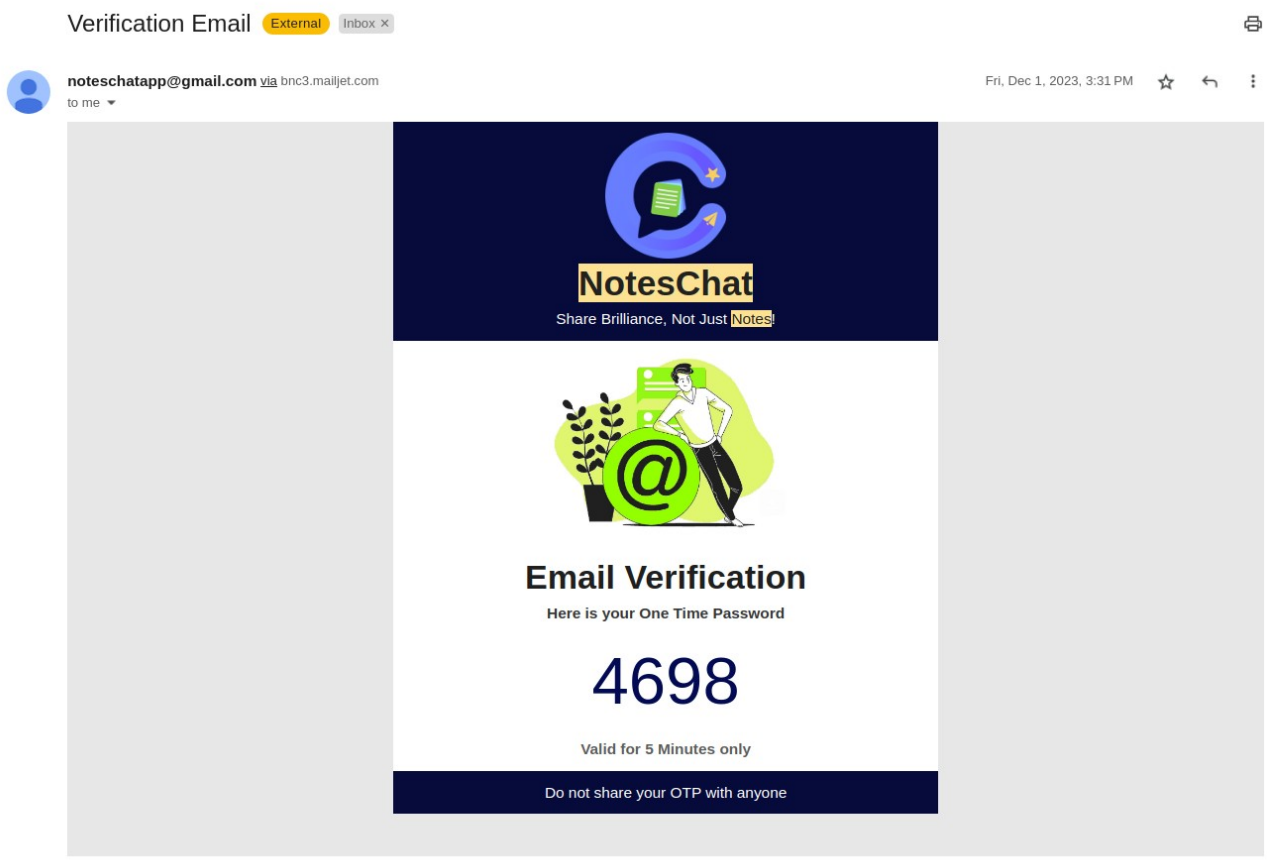


Figure 2.4 HTML Email Template

2.4.1 Advantages of HTML Email Templates

1. Visual Appeal
HTML email templates offer the flexibility to incorporate images, colors, and fonts, enhancing the visual appeal of the email and capturing the recipient's attention. This visual richness can significantly impact engagement rates and drive desired actions, such as clicks or conversions.

2. Brand Consistency
Customizing the design elements of HTML email templates ensures brand consistency across all communication channels. By aligning the templates with the company's branding guidelines, businesses can reinforce brand identity and build trust with their audience.

3. Personalization
Leveraging dynamic content insertion using .ejs files, we were able to personalize email content based on recipient data such as name. This personalization increases relevance and engagement, driving higher open and click-through rates.

4. Tracking and Analytics
HTML emails enable tracking of user interactions such as opens, clicks, and conversions. By integrating tracking pixels and campaign tags, businesses can gain valuable insights into campaign performance, audience behavior, and ROI, allowing for data-driven decision-making and optimization.

5. Automation and Scalability
Utilizing AWS SES and reusable functions, we streamlined the process of creating and sending email templates. This automation not only saves time and resources but also enables businesses to scale their email marketing efforts and reach a large audience efficiently.

2.4.2 Limitations of HTML Email Templates

1. Cross-Platform Compatibility
HTML email rendering can vary across different email clients and devices, leading to inconsistent display and user experience. Designing templates that render correctly across a wide range of email clients, including desktop, web, and mobile, can be challenging and may require extensive testing and optimization.

2. Limited Design Flexibility
Due to the constraints imposed by email client rendering engines and support for CSS styles, achieving complex design layouts can be challenging. Designers must work within the limitations of email client compatibility and prioritize simplicity and functionality over elaborate design elements.

3. Spam Filters
Emails with excessive use of images, links, or certain keywords may trigger spam filters, impacting deliverability and campaign effectiveness. Balancing visual appeal with content relevance and spam filter compliance is essential to ensure that emails reach the intended recipients' inbox.

4. Accessibility
Ensuring accessibility for users with disabilities, such as screen readers, can be challenging when designing HTML email templates. Designers must consider accessibility standards and best practices to ensure that all recipients can access and understand the email content.

2.4.3 Key Considerations for Effective HTML Email Template Development

1. Responsive Design
Prioritizing responsive design principles ensures that email templates are optimized for viewing across various screen sizes and devices. Utilizing media queries and fluid layouts, designers can create templates that adapt seamlessly to different viewport sizes, improving user experience and engagement.

2. Content Optimization
Keeping email content concise, engaging, and relevant increases the likelihood of recipients taking desired actions. Designers should focus on delivering value-driven content that addresses recipients' needs and interests, driving higher click-through and conversion rates.

3. Testing and Iteration
Regular testing of email templates across different email clients and devices is crucial to identify and address rendering issues. Utilizing testing tools and services, designers can simulate email client environments and preview template rendering to ensure a consistent user experience across all platforms.

4. Compliance and Privacy
Designers must ensure that email templates include necessary compliance elements, such as unsubscribe links and clear opt-in/opt-out mechanisms, to comply with regulatory requirements and respect recipient privacy preferences.

2.5 CALL MANAGEMENT SYSTEM DEVELOPMENT

Ever since I was a child, the mystery of automated phone calls has intrigued me. Whenever my parent’s phone rang with a robotic voice on the other end, I couldn't help but wonder: Who was behind that voice? How did they teach a machine to speak, especially in my local language and others I couldn't even understand?

This childhood curiosity ignited a spark within me, leading to my most captivating project during my internship: the development of a Call Management System. Powered by Twilio's functionality and services, this project aimed to unravel the secrets behind automated calls and call handling through features provided by twilio.

2.5.1 Advantages of Call Masking

Call masking, a pivotal feature of the Call Management System, offers numerous advantages in today's digital landscape. It serves as a powerful tool for privacy protection, enabling businesses to connect customers with service providers while safeguarding personal information. By assigning temporary virtual numbers to both parties, call masking ensures anonymity and security, fostering trust and confidentiality in sensitive transactions such as ride-hailing, real estate, and healthcare.

2.5.2 How call masking is achieved using twilio

Call masking in Twilio involves setting up a communication flow where two parties can communicate without revealing their personal phone numbers to each other. This is commonly used in marketplaces, ride-sharing apps, and similar platforms where privacy is essential. I’ll try to explain how call masking works using Twilio's TwiML (Twilio Markup Language) apps and webhooks:

1. Setup Twilio Numbers
You start by acquiring at least two phone numbers from Twilio. One number will be used by the caller (party A), and the other will be used by the callee (party B).

2. TwiML App Configuration
You create a TwiML application in your Twilio account. A TwiML application is essentially a configuration that tells Twilio what to do when it receives a call or message on one of your Twilio numbers. In this case, you'll configure it to use webhooks to control call flow.

3. Webhook Configuration
Within your TwiML application, you define webhook URLs where Twilio will send HTTP requests when certain events occur during a call. These events include incoming calls, call status changes, and user input (DTMF tones).

4. Call Flow Setup
When a call comes in, Twilio sends an HTTP request to your webhook endpoint. Your server responds with TwiML instructions, which can include instructions to dial another number (party B) or to send an automated message to the caller. Let us see at some of the twiml instructions

Atomated message

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
    <Say voice="alice">
        Hello, this is an automated message
    </Say>
</Response>
```

Figure 2.5 Automated Messge Twiml

Whenever our webhook returns the twiml shown in figure 2.4, the caller who triggered the webhook will get to hear an automatted saying “Hello this is an automated message”

Call Forwarding

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
    <Dial callerId="+15555555555">
        <Number>+1234567890</Number>
    </Dial>
</Response>
```

Figure 2.6 Call forwarding/Masking Twiml

Whenever aur webhook returns the twiml shown in figure 2.6, the number ‘+1234567890’ will be dialed by twilio. This is used to achieve call masking in a way that the user will trigger the webhook when he calls on the number ‘+15555555555’, so his dialer app will show this number and not the receivers actual number, when the twilio receives call on ‘+15555555555’ number it will consult our webhook for the twiml instruction. On receipt of the instruction in figure 2.6 twilio will dial the number ‘+1234567890’ using the number ‘+15555555555’, therby connecting the caller and the receiver both seeing the twilio number ‘+15555555555’ on their dialer app therby achieving call masking.

5. Masking Numbers
Instead of directly connecting party A with party B, Twilio acts as an intermediary. It uses a masked phone number to connect the two parties. This masked number serves as a bridge between party A and party B, ensuring that neither party sees each other's actual phone numbers.

6. Dynamic Call Control
During the call, your server can dynamically control the call flow based on various factors. For example, you can add features like call recording, call transcription, call transfer, or automated prompts using TwiML instructions sent via webhooks.

7. Call Completion and Cleanup
Once the call is complete, you can handle call termination and any necessary cleanup tasks in your webhook endpoint. This might involve releasing resources, logging call details, or triggering follow-up actions in your application.

2.5.3 Development: Frontend and Backend Implementation

The development of the Call Management System involved the creation of both frontend and backend components, each playing a crucial role in delivering a seamless user experience and robust functionality.

1. Frontend Development with React + TypeScript

On the frontend, we leveraged the power of React, a popular JavaScript library for building user interfaces to create a dynamic and responsive web interface. The decision to use TypeScript, a statically typed superset of JavaScript enhanced the development process by providing type safety and improved code quality.

Utilizing React's component-based architecture, we modularized the user interface into reusable components, promoting code reusability and maintainability. This

approach facilitated the creation of a mobile-responsive web interface tailored specifically for smartphones, aligning with the inherent nature of call management activities.

2. Backend Development with Node.js + Express

The backend of the Call Management System was developed using Node.js, a runtime environment for executing JavaScript code outside of a web browser and Express.js, a minimalist web application framework for Node.js. This powerful combination provided a solid foundation for building robust and scalable server-side applications.

For data storage and retrieval, we employed PostgreSQL, a powerful open-source relational database management system. The integration of Sequelize, an Object-Relational Mapping (ORM) library for Node.js simplified database interactions by abstracting away the complexities of SQL queries and providing a seamless interface for CRUD operations.

2.5.4 Understanding the call mangement service flow

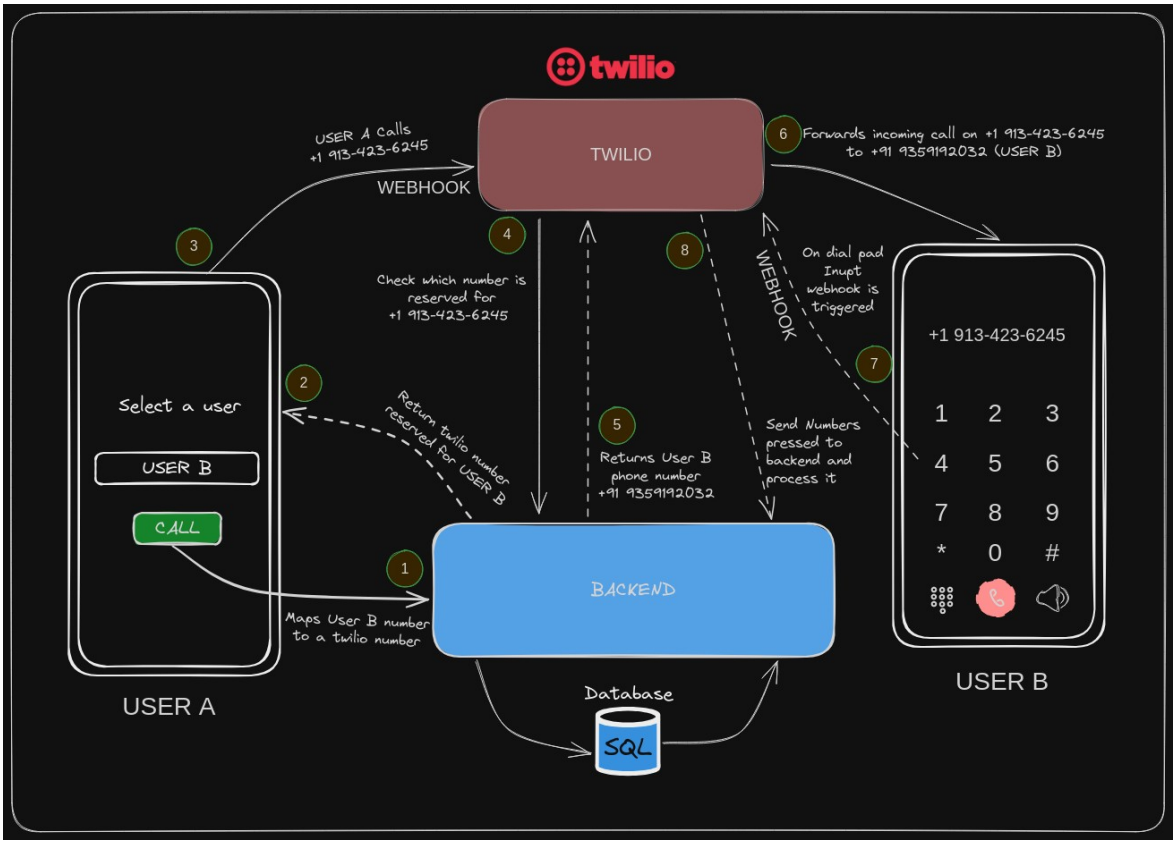


Figure 2.7 Call Management system flow

STEP 01:
User A selects user B from our UI and presses the Call button, on doing so user A is mapped to user B in the database, indicating that user A will be calling user B on assigned twilio number. This assignment or mapping is valid for a configurable amout of time

STEP 02:
Once the users are mapped the assigned twilio number is returned to the user A, after which an script is executed in the web app to redirect the user A to his dialer app with the returned twilio number prefilled for calling.

STEP 03:
The user A then calls the twilio number which will trigger a webhook connected to that twilio phone number. This webhook will send the callers data such as callers phone number, the twilio number which triggered the call etc. to our backend.

STEP 04:
In our backend we will check who triggered the webhook (user A) and for which twilio number, since we had maintained the users mapping we can easily retrive user B phone number.

STEP 05:
We then send the twilio phone number and the number which should be dialed (user B's phone number) in the twiml format (like the one we saw in figure 2.6) back to the twilio.

STEP 06:
Twilio, after receiveing user B's phone number will call user B using the same twilio number on which user A has called (and is still connected with), therby forwarding/masking the call connection. Now user A and user B are connected with each other over a phone call via the twilio number, the twilio number here acts as an middle man. User A on his dialer app see the twilio number which he as dialed on (eg +15555555555) and user B on his dialer app will see the same twilio number (+15555555555) therby hiding or masking each others actual phone numbers.

STEP 07:
While on the call the user A or user B can dial some numbers on their dial pad, this DTMF tone are then sent to the twilio where another webhook will be triggered to capture this DTMF tones or dial pad inputs.

STEP 08:
The captured DTMF tones is sent to the backend where further required processing is done.

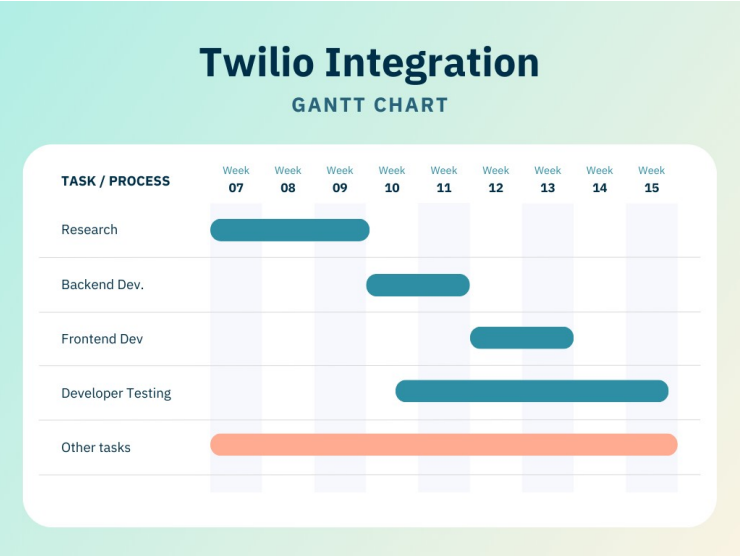


Figure 2.8 Call Management Gantt Chart

CHAPTER 03 : LEARNING

During my internship, I gained invaluable experience and deepened my knowledge in several key areas that have significantly contributed to my development as a software developer. These learnings spanned advanced technical concepts, practical tool usage, and important non-technical skills crucial for professional growth in a tech company.

On the technical front, I delved into advanced concepts of React, which enhanced my ability to build dynamic and responsive user interfaces. I explored the architecture and implementation of microservices, which provided me with insights into designing scalable and maintainable systems. My exposure to Amazon AWS equipped me with the skills to deploy and manage cloud-based applications.

I also tackled the complexities of handling date and time, especially dealing with time zones and daylight saving changes, using Moment.js. Effective version control using Git became second nature, improving my collaboration and code management capabilities. Additionally, I learned about server-sent events and their applications, which expanded my understanding of real-time web functionalities.

Exploring Kafka, a distributed event streaming platform, was another highlight, as it introduced me to handling large-scale data processing and real-time analytics. The principles and implementation of A/B testing allowed me to understand how to optimize features and user experiences through controlled experiments.

Beyond technical skills, I acquired non-technical insights essential for thriving in a tech environment. These included strategies for effective communication, teamwork, and professional development, which are all critical for long-term success in the industry.

3.1 REACT ADVANCED CONCEPTS

At Spintly, we majorly use React for frontend development so I had the opportunity to dive deep into advanced concepts of React, which significantly enhanced my proficiency in this powerful JavaScript library. Building upon my foundational knowledge, I explored several key areas that are critical for developing complex, high-performance web applications. I have listed down some of this key concepts that I had to learn in details during my internship period.

3.1.1 Component Lifecycle

Understanding the component lifecycle in React was crucial for optimizing the performance and behavior of our application. I delved into lifecycle methods such as componentDidMount, componentDidUpdate, and componentWillUnmount, learning how to manage state and side effects effectively. This knowledge allowed me to implement efficient data fetching, resource cleanup, and performance tuning within my components.

3.1.2 State Management

Managing state in large applications can be challenging, and I explored various strategies to handle this complexity. I gained hands-on experience with the Context API, which helped in passing data through the component tree without having to prop-drill at every level. Additionally, I worked with popular state management libraries like Redux, understanding how to maintain a global state and ensure consistency across the application.

3.1.2 Hooks

I learned to use built-in hooks like useState, useEffect, useContext, and useReducer to manage state and side effects in a more declarative manner. Custom hooks also became an essential part of my toolkit, enabling me to encapsulate reusable logic and keep my components clean and focused.

3.1.2 Performance Optimization

Performance is a key consideration in web development, and I delved into techniques for optimizing React applications. I learned about code splitting using React.lazy and Suspense to load components on demand, reducing the initial load time. Memoization techniques with React.memo and useMemo helped in preventing unnecessary re-renders and improving the efficiency of the application. Additionally, I explored the use of tools like the React Developer Tools and profiling features to identify and

address performance bottlenecks.

3.2 MICROSERVICES

During my internship, I had the opportunity to immerse myself in the world of microservices, a modern architectural style that structures an application as a collection of loosely coupled services. This approach offers numerous advantages, including enhanced scalability, flexibility, and resilience, which are crucial for building robust and maintainable systems.

3.2.1 Understanding Microservices Architecture

Microservices architecture involves breaking down a monolithic application into smaller, independent services that communicate with each other through well-defined APIs. Each microservice focuses on a specific business function, such as user management, payment processing, or inventory management, and operates as a standalone application. This modularity allows teams to develop, deploy, and scale services independently, leading to faster development cycles and more efficient use of resources.

3.2.2 Key Components and Concepts

1. **Service Independence:** Each microservice is an autonomous unit with its own database and deployment pipeline. This independence minimizes dependencies between services, reducing the risk of cascading failures and making the system more resilient.

2. **APIs and Inter-Service Communication:** Microservices communicate through APIs, typically using RESTful endpoints. For asynchronous communication, message brokers like Apache Kafka are often employed. This decoupling of services ensures that changes in one service do not directly impact others.

3. **Data Management:** Unlike monolithic architectures, where a single database is shared across the entire application, microservices advocate for decentralized data management. Each service manages its own database, promoting data encapsulation and reducing the likelihood of cross-service data corruption.

4. **Scalability and Flexibility:** Microservices can be individually scaled based on their specific load and performance requirements. This granular scalability is more efficient compared to scaling an entire monolithic application, allowing better resource utilization and cost management.

3.2.3 Challenges and Solutions

1. **Complexity:** Managing multiple services can be complex, especially in terms of monitoring, logging, and debugging. Solutions include using centralized logging systems (e.g., ELK Stack) and distributed tracing tools (e.g., Jaeger, Zipkin) to gain insights into the interactions between services.

2. **Inter-Service Communication:** Ensuring reliable communication between services can be challenging, particularly in handling network latency and failures. Implementing patterns like Circuit Breaker (e.g., Hystrix) and service mesh technologies (e.g., Istio) can enhance the reliability and resilience of inter-service communication.

3. **Data Consistency:** Maintaining data consistency across services requires careful design, particularly when dealing with transactions. Techniques such as eventual consistency and the Saga pattern help manage distributed transactions effectively.

3.2.3 Benefits of Microservices

1. **Scalability:** Microservices allow for targeted scaling of individual services, optimizing resource usage and improving application performance.

2. **Flexibility:** Different services can be developed using different technologies and languages, allowing teams to choose the best tools for each task.

3. **Resilience:** The decoupled nature of microservices enhances fault isolation, ensuring that failures in one service do not cascade and bring down the entire system.

4. **Faster Development:** Independent development and deployment of services enable quicker iterations and faster delivery of new features and updates.

3.3 AMAZON AWS

During my internship, I gained comprehensive experience with several Amazon Web Services (AWS) offerings that are critical for modern web application development. These services include Amazon S3, CloudFront, SQS, Cognito, EC2, AWS RDS, and AWS SNS. Each of these services provides unique capabilities that contribute to building scalable, secure, and high-performing applications.

3.3.1 Amazon S3 (Simple Storage Service)

Amazon S3 is a highly scalable and durable object storage service designed to store and retrieve any amount of data from anywhere on the web. S3 allows you to organize data into buckets, which act as containers for objects (files). S3 ensures the safety and availability of your data. It supports various storage classes to optimize costs based on access patterns, from frequently accessed data (Standard) to long-term archival storage (Glacier). S3 also provides robust security features, including bucket policies, Access Control Lists (ACLs), and integration with AWS Identity and Access Management (IAM) for fine-grained access control.

3.3.2 Amazon CloudFront

Amazon CloudFront is a Content Delivery Network (CDN) service that speeds up the distribution of static and dynamic web content to users globally. CloudFront caches content at edge locations worldwide, reducing latency and improving load times by delivering content from the location closest to the end-user. It integrates seamlessly with other AWS services like S3, allowing for efficient content delivery of assets stored in S3 buckets. CloudFront also provides advanced security features such as SSL/TLS encryption, AWS Shield for DDoS protection, and AWS WAF (Web Application Firewall) for protecting web applications from common web exploits.

3.3.3 Amazon SQS (Simple Queue Service)

Amazon SQS is a fully managed message queuing service that enables decoupling and scaling of microservices, distributed systems, and serverless applications. SQS allows you to send, store, and receive messages between software components without losing messages or requiring other services to be available. It supports two types of message queues: Standard Queues, which offer high throughput and at-least-once delivery, and FIFO Queues, which ensure messages are processed exactly once, in the exact order they are sent. SQS helps in building reliable and scalable applications by enabling asynchronous processing and communication.

3.3.4 Amazon Cognito

Amazon Cognito provides user authentication, authorization, and user management for web and mobile applications. It supports sign-in with social identity providers like Facebook, Google, and Amazon, as well as enterprise identity providers via SAML 2.0 and OpenID Connect. Cognito User Pools handle user registration, authentication, and account recovery, while Cognito Identity Pools enable you to grant users access to AWS resources. With built-in security features such as multi-factor authentication (MFA) and encryption of data at rest and in transit, Cognito ensures that user data and interactions are secure.

3.3.5 Amazon EC2 (Elastic Compute Cloud)

Amazon EC2 offers resizable compute capacity in the cloud, allowing you to run virtual servers on-demand. EC2 provides a wide range of instance types optimized for different use cases, including general-purpose, compute-optimized, memory-optimized, and storage-optimized instances. EC2 supports various operating systems and offers features like auto-scaling, which adjusts compute capacity based on demand, and Elastic Load Balancing, which distributes incoming application traffic across multiple instances. EC2's flexibility and scalability make it ideal for a wide range of applications, from small web applications to large-scale enterprise systems.

3.3.6 Amazon RDS (Relational Database Service)

Amazon RDS is a managed relational database service that simplifies the setup, operation, and scaling of databases in the cloud. It supports several database engines, including MySQL, PostgreSQL, MariaDB, Oracle, and Microsoft SQL Server. RDS automates time-consuming administrative tasks such as hardware provisioning, database setup, patching, and backups. It also provides high availability and durability through Multi-AZ deployments and automated backups. RDS allows you to focus on your application's functionality while AWS manages the underlying database infrastructure.

3.3.7 Amazon SNS (Simple Notification Service)

Amazon SNS is a fully managed messaging service for sending notifications to subscribers. It supports both push and pull messaging, making it versatile for various use cases. With SNS, you can send messages to multiple subscribers, including applications, end-users, and other AWS services. It supports different protocols for delivering messages, such as HTTP/HTTPS, email, SMS, and AWS Lambda. SNS can be used to send notifications for application events, alerting systems, and distributed systems, facilitating real-time communication and ensuring timely

delivery of critical messages.

3.4 HANDLING DATE & TIME

During my internship, one of the most challenging tasks I faced was handling date and time accurately, especially when dealing with time zones and daylight saving time (DST). Initially, understanding and implementing these concepts was difficult, but with persistence, I succeeded.

3.4.1 The Complexity of Time Zones

Time zones are geographic regions with a standard time offset from Coordinated Universal Time (UTC). Handling time zones required converting local times to UTC and vice versa to ensure accuracy across various regions. Each region has different rules, which can change, adding to the complexity.

3.4.2 Daylight Saving Time (DST)

DST further complicates things by shifting clocks forward in warmer months. Not all regions observe DST, and those that do start and end it on different dates. This meant ensuring our system could handle repeated or skipped local times correctly.

3.4.3 Tools and Techniques

To manage these complexities, I used Moment.js and its companion library, Moment Timezone. These tools simplified parsing, formatting, and converting dates across time zones and handled DST adjustments automatically.

3.4.4 Challenges and Learning Outcomes

Initially, I faced issues like incorrect event times and scheduling conflicts. By converting all times to UTC for storage and back to local time for display, I ensured consistency. Rigorous testing across various time zones and DST transitions helped identify and resolve edge cases.

3.5 USING GIT EFFECTIVELY

During my internship, mastering Git for version control was crucial to maintaining a smooth workflow and collaboration within our development team. Git allowed us to track changes, manage code versions, and collaborate efficiently.

In our company, we followed a specific branch naming convention to ensure consistency and clarity: DEVELOPER_FEATURE_DATE. This convention helped in easily identifying who worked on which feature and when it was created. Every new feature branch was checked out from the main branch. For example, if a developer named Harshal was working on a login feature on May 27th 2024, the branch would be named Harshal_LOGIN_27th_May_2024.

We managed multiple environments (development, testing, staging, production), each with its respective base branch. Here's how our workflow operated:

1. **Feature Development in Dev Environment:**

   Check Out a New Branch:

   ```
   git checkout main
   git pull origin main
   git checkout -b Harshal_LOGIN_27th_May_2024
   ```

   Develop and Commit Changes:
   Regular commits are made to the feature branch, documenting the progress and changes:

   ```
   git add .
   git commit -m "Implemented login feature"
   ```

   Push the Branch:
   After development and local testing, the branch is pushed to the remote repository:

   ```
   git push
   ```

   Merge to Development Base Branch:
   Once basic testing is completed by the developers, the feature branch is merged with the dev branch:

```
git checkout dev
git merge Harshal_LOGIN_27th_May_2024
git push
```

2. **Testing in Test Environment:**

   Merge to Testing Base Branch:
   After merging with the dev branch, the feature is moved to the testing environment by merging the feature branch with the test branch:

   ```
   git checkout test
   git merge Harshal_LOGIN_27th_May_2024
   git push
   ```

   QA Testing:
   QA conducts thorough testing on the test branch. There could be several new features in the testing environment which are not yet deployed to production.

3. **Preparation in Staging Environment:**

   Create Staging Branch:
   A single feature approved by QA is moved to the staging environment. A staging branch is created from the main branch and is merged with the feature branch:

   ```
   git checkout main
   git pull origin main
   git checkout -b staging_LOGIN_27_05_2024
   git merge Harshal_LOGIN_27th_May_2024
   git push
   ```

   QA Testing on Staging:
   QA conducts another round of testing on the staging branch. This branch represents the main branch plus the QA-approved feature.

4. **Release to Production:**

   Merge Staging to Main
   Once the staging branch is approved, it is merged into the main branch by raising a pull request (PR). The PR is then approved and merged with main, so now the feature is released to production.

3.6 SERVER-SENT EVENTS (SSE)

During my internship, one of the most challenging tasks I faced was handling date and time accurately, especially when dealing with time zones and daylight saving time (DST). Initially, understanding and implementing these concepts was difficult, but with persistence, I succeeded.

3.6.1 How SSE Works

SSE utilizes a simple protocol where the server sends messages over an HTTP connection. The client, typically a web browser, listens to these messages and processes them as they arrive.

3.6.2 Advantages of SSE

1. **Simplified Communication:** SSE provides a straightforward way for the server to send real-time updates to the client without complex setups.

2. **Efficient Resource Usage:** SSE uses a single HTTP connection, reducing the overhead compared to continuous polling.

3. **Automatic Reconnection:** The EventSource API automatically handles reconnections, providing robustness to network issues.

4. **Compatibility:** SSE works seamlessly with existing web technologies and is supported by most modern browsers.

3.6.3 Limitations of SSE

1. **One-Way Communication:** SSE is designed for one-way communication from the server to the client. For two-way communication, other technologies like WebSockets might be more appropriate.

2. **Limited Browser Support:** While most modern browsers support SSE, some older browsers and Internet Explorer do not.

3. **Connection Limits:** Browsers impose limits on the number of concurrent SSE connections, which might be a consideration for applications requiring many simultaneous connections.

3.7 KAFKA

During my internship, I had the opportunity to work extensively with Apache Kafka, a distributed event streaming platform, as part of our microservice architecture. Kafka played a crucial role in ensuring seamless communication and data exchange between various services in our system.

3.7.1 Kafka in Our Microservice Architecture

In our company, we used Kafka to enable event-driven communication among microservices. This approach allowed services to publish and subscribe to event streams, facilitating independent operation and interaction solely through Kafka topics. For instance, when a new user registered, the user service would publish an event to a Kafka topic. Other services, like email notifications or analytics, would subscribe to this topic to perform their respective tasks, eliminating direct dependencies between services.

Kafka acted as a central hub for data integration within our microservices. It collected and distributed data streams from various sources, ensuring that all services had access to up-to-date information. This integration was essential for maintaining consistency across our system.

3.7.2 Kafka in Our Microservice Architecture

In our architecture, microservices functioned as both producers and consumers within Kafka. Producers published events to Kafka topics, and consumers subscribed to these topics to receive and process the events. The topics were divided into partitions, enabling parallel processing and horizontal scaling of our microservices. This partitioning allowed us to handle increased traffic and maintain efficient processing.

Our Kafka setup ran on a cluster of servers known as brokers, which stored data and served client requests. Kafka’s architecture ensured fault tolerance by replicating data across multiple brokers. This replication meant that if one broker failed, another could take over, preventing data loss and maintaining service continuity.

A significant feature of Kafka that we leveraged was message retention. Kafka stored messages for a configurable period, allowing microservices to consume data at their own pace and replay messages if needed. This feature was particularly useful for event sourcing and auditing, ensuring that all events were processed correctly even if some services experienced downtime or delays.

3.7.3 Limitations of Kafka

Despite its many advantages, using Kafka in our microservice architecture also presented some challenges and limitations. Setting up and maintaining a Kafka cluster can be complex, requiring careful planning and configuration to ensure optimal performance and fault tolerance. Managing Kafka's infrastructure, including brokers, topics, and partitions, demands significant effort and expertise.

While Kafka guarantees message ordering within a partition, achieving global ordering across all partitions can be challenging. This limitation can complicate scenarios where strict ordering of messages across the entire system is required. Additionally, Kafka can be resource-intensive, especially in large-scale deployments.

Kafka also has a steep learning curve, particularly for developers and administrators new to the platform. Understanding its concepts, configurations, and best practices requires time and training, which can slow down initial adoption and implementation.

3.8 A/B TESTING

During my internship, I was involved in implementing A/B testing to optimize our web application's user interface (UI) and improve user engagement. A/B testing, also known as split testing, is a method of comparing two or more versions of a webpage or app against each other to determine which one performs better. This process involves displaying different versions to different segments of users and analyzing their interactions to make informed decisions.

3.8.1 Implementation of A/B Testing

In our A/B testing setup, we used Server-Sent Events (SSE) to dynamically inform the client which version of the UI to display. SSE allowed us to push real-time updates from the server to the client's browser, ensuring that users received the appropriate UI version as soon as they accessed the application. This approach provided a seamless experience without requiring the user to refresh the page or take any additional action.

To implement this, we created multiple versions of the UI, each with slight variations in design or functionality. When a user accessed the application, the server would randomly assign them to one of the versions and send this information to the client using SSE. This random assignment ensured an unbiased distribution of users across different versions, which is crucial for the validity of the A/B test.

3.8.2 Tracking and Analyzing Performance with Mixpanel

To track user interactions and determine which version of the UI performed better, we integrated Mixpanel, a powerful analytics platform. Mixpanel allowed us to collect detailed data on user behavior, such as clicks, page views, and conversion rates. By tagging specific actions and events within the application, we could analyze how users interacted with each version of the UI.

Mixpanel provided valuable insights into user engagement, highlighting which version led to higher user satisfaction and better performance metrics. For instance, we could compare the conversion rates of different versions to see which one performs better.

3.8.3 Advantages of A/B Testing

A/B testing offers several advantages for optimizing web applications and improving user experience. One of the primary benefits is data-driven decision-making. Rather than relying on intuition or assumptions, A/B testing provides concrete evidence of which UI version performs better based on real user interactions.

This method also allows for incremental improvements. By testing small changes one at a time, we could identify what specific elements contributed to better performance. This iterative approach ensures continuous optimization and refinement of the UI.

Additionally, A/B testing helps mitigate risks associated with major design changes. Instead of overhauling the entire application at once, we could test changes on a smaller scale and roll out successful versions gradually. This reduces the likelihood of negatively impacting the user experience.

3.8.4 Advantages of A/B Testing

While A/B testing is a powerful tool, it also comes with challenges and considerations. One of the primary challenges is ensuring that the test results are statistically significant. This requires a sufficiently large sample size and adequate testing duration to draw reliable conclusions. Without these, the results might be inconclusive or misleading.

Another consideration is the potential for user segmentation to affect results. Different user groups may respond differently to the same UI change, so it's important to account for variations in user demographics, behavior, and preferences.

Lastly, maintaining consistency and avoiding biases in the testing process is crucial. Random assignment of users to different versions and consistent tracking of performance metrics are essential to ensure the validity of the test.

3.9 NON TECH LEARNINGS

My internship at the Spintly was not only an opportunity to deepen my technical skills but also a crucial period for personal and professional growth. The non-technical learnings I gained have proven to be equally valuable, shaping my approach to work and collaboration in a tech-driven environment.

3.9.1 Effective Communication

One of the most significant non-technical skills I developed was effective communication. In a collaborative environment, clear and concise communication is essential. I learned how to articulate my ideas, present my work, and provide updates to team members and stakeholders. Whether it was through daily stand-ups, meetings, or written reports, I honed my ability to convey complex information in an understandable manner. This skill has been particularly beneficial in ensuring that projects stayed on track and that everyone involved had a clear understanding of their roles and responsibilities.

3.9.2 Team Collaboration

Working in a team setting emphasized the importance of collaboration. I learned how to work effectively with cross-functional teams, including developers, designers, and QA testers. This experience taught me the value of diverse perspectives and how leveraging the strengths of each team member can lead to better problem-solving and innovative solutions. Collaborative tools and practices, such as code reviews and pair programming, also enhanced our productivity and ensured high-quality deliverables.

3.9.3 Time Management

Managing multiple tasks and deadlines was a critical aspect of my internship. Being a full stack developer I developed strong time management skills, learning how to prioritize tasks, set realistic deadlines, and manage my workload efficiently. This involved balancing long-term projects with immediate responsibilities and learning how to break down complex tasks into manageable steps. Effective time management not only helped me meet deadlines but also reduced stress and increased productivity.

3.9.4 Adaptability

The fast-paced nature of the tech industry requires a high degree of adaptability. During my internship, I faced various challenges, such as changing project requirements, new technologies, and unforeseen obstacles. I learned to embrace these changes and adapt quickly. This flexibility allowed me to stay resilient and continue progressing despite uncertainties, a crucial skill in any dynamic work environment.

3.9.5 Problem-Solving and Critical Thinking

Every project presented unique challenges that required creative problem-solving and critical thinking. I learned how to approach problems systematically, analyze different solutions, and choose the most effective course of action. This process often involved seeking input from colleagues, researching best practices, and testing multiple solutions. Developing these skills has been invaluable in tackling complex issues and contributing to successful project outcomes.

3.9.6 Networking and Relationship Building

Building professional relationships and networking within the company was another important learning aspect. I learned how to connect with colleagues, seek mentorship, and participate in company events. These interactions provided valuable insights into different career paths and helped me build a supportive professional network. Networking also opened up opportunities for collaboration and knowledge sharing, which enriched my learning experience.

CHAPTER 04 : CHALLENGES

During my internship as a software developer, I faced several challenges that tested my abilities and fostered significant growth. These challenges were crucial in shaping my professional development and enhancing my problem-solving skills. Among the various difficulties encountered, two areas stood out: task management and research-oriented learning.

Task management presented a considerable challenge as I had to juggle multiple responsibilities simultaneously. Balancing coding tasks, bug fixes, feature development, and meetings required efficient prioritization and time management. Meeting deadlines while ensuring the quality of my work demanded a structured approach and adaptability to shifting priorities. Learning to manage my time effectively was essential to staying on top of my workload and maintaining productivity.

Research-oriented learning was another major hurdle. As part of my role, I frequently encountered new technologies, frameworks, and concepts that I needed to understand and apply quickly. The vast amount of available information often felt overwhelming, and distinguishing between reliable sources and less useful content was challenging. Additionally, integrating theoretical knowledge into practical applications required a deep understanding of the material and often involved troubleshooting unexpected issues.

Overcoming these challenges was not easy, but it provided invaluable lessons and skills that will serve me well in my career. Through task management, I learned the importance of organization, prioritization, and efficient time use. Research-oriented learning taught me to be resourceful, persistent, and proactive in seeking solutions and expanding my knowledge base. These experiences collectively contributed to my growth as a software developer, equipping me with the tools needed to tackle future challenges.

4.1 TASK MANAGEMENT

One of the most significant challenges I faced during my internship was task management. As a full stack developer, I was responsible for juggling multiple tasks, ranging from coding new features to fixing bugs and assisting with various projects. Managing these diverse responsibilities effectively was crucial to ensure timely and quality delivery. At some times it was quiet frustrating as well.

4.1.1 Initial Struggles

At the beginning of my internship, I struggled with prioritizing tasks. Every task seemed urgent and important, making it difficult to decide which ones to tackle first. The absence of a structured task management approach led to occasional missed deadlines and a sense of being overwhelmed. Additionally, I found it challenging to estimate the time required for each task accurately. This often resulted in underestimating the complexity of certain assignments, leading to last-minute rushes and added stress.

4.1.2 Learning to Prioritize

To overcome these challenges, I had to develop a better system for prioritizing tasks. I started by categorizing tasks based on their urgency and importance. This approach helped me focus on high-priority tasks that required immediate attention and allowed me to defer or delegate less critical ones. By breaking down larger tasks into smaller, manageable chunks, I was able to create a more organized and realistic schedule.

4.1.3 Time Management Techniques

Improving my time management skills was another crucial aspect. I began using time management techniques, which involves working in focused intervals of 25 minutes followed by short breaks. This method helped me maintain concentration and productivity throughout the day. Additionally, I set specific goals for each work session, which provided a sense of accomplishment and kept me motivated. This technique was suggested to me by one of my collegue.

4.1.4 Learning to Communicate and Collaborate

Effective communication and collaboration with team members also played a vital role in managing tasks efficiently. I learned to communicate proactively about my workload and seek clarification on priorities. Regular check-ins with my supervisor and team allowed me to align my tasks with the overall project goals and receive timely feedback. This collaborative approach not only improved task management

but also fostered a supportive team environment.

4.1 RESEARCH ORIENTED LEARNING

During my internship, one of the toughest challenges I faced was research-oriented learning. As a software developer intern, I often had to learn about new technologies, frameworks, and concepts quickly to apply them to my projects. This process was demanding and required a lot of dedication and patience.

At first, the sheer amount of information available online was overwhelming. When I needed to learn something new, I found countless tutorials, articles, and documentation. It was hard to figure out which resources were useful and which ones weren’t.

Time management added to the difficulty. I had to balance researching and learning new things while also working on my assigned tasks and meeting deadlines. Often, I had to quickly understand and apply new concepts, which sometimes led to a shallow understanding. This lack of deep knowledge could make it challenging to produce high-quality work.

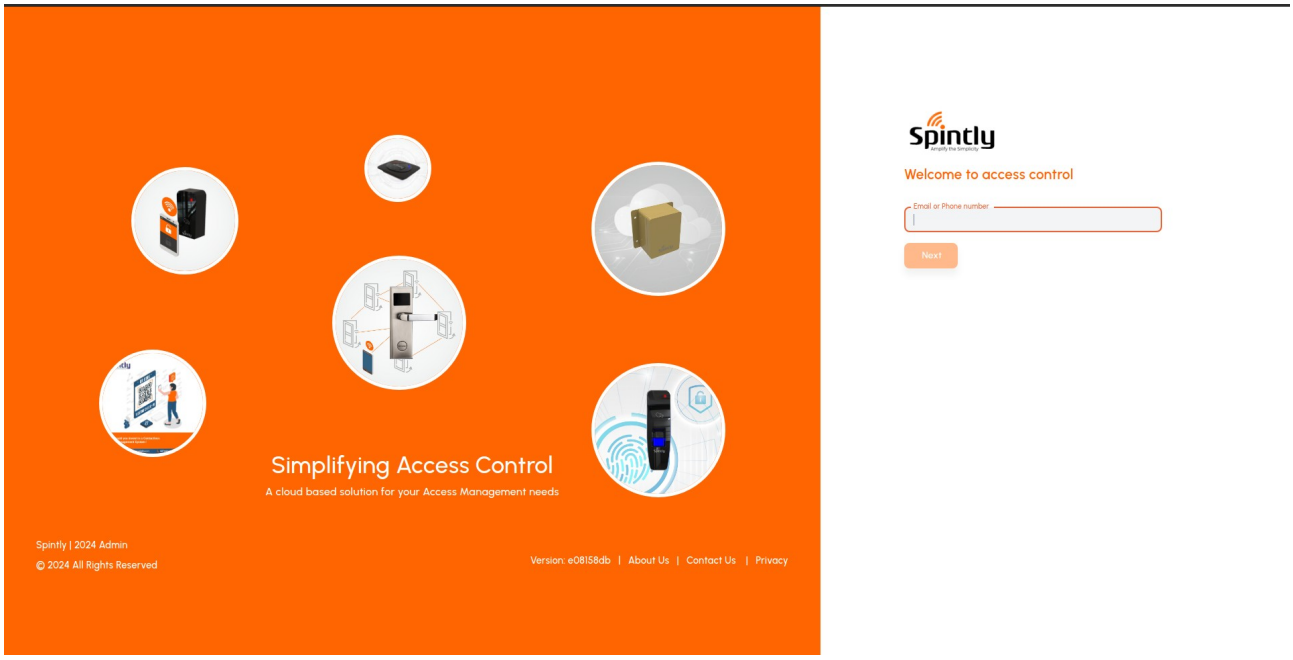
Another major challenge was putting theoretical knowledge into practice. Reading about a new technology is one thing, but using it in a real project is another. I often ran into issues that weren’t covered in the resources I studied. Finding solutions required digging deeper into documentation and forums, which was time-consuming and sometimes frustrating.

To manage these challenges, I developed a structured approach to my research. I started by identifying reliable sources of information, such as official documentation and reputable blogs. This helped me find accurate and up-to-date information more quickly. I also broke down complex topics into smaller, manageable parts, focusing on one aspect at a time to build a solid foundation before moving on to more advanced concepts.

Collaboration with colleagues was also incredibly helpful. Discussing my challenges and solutions with more experienced team members provided valuable insights and speed up my learning process.

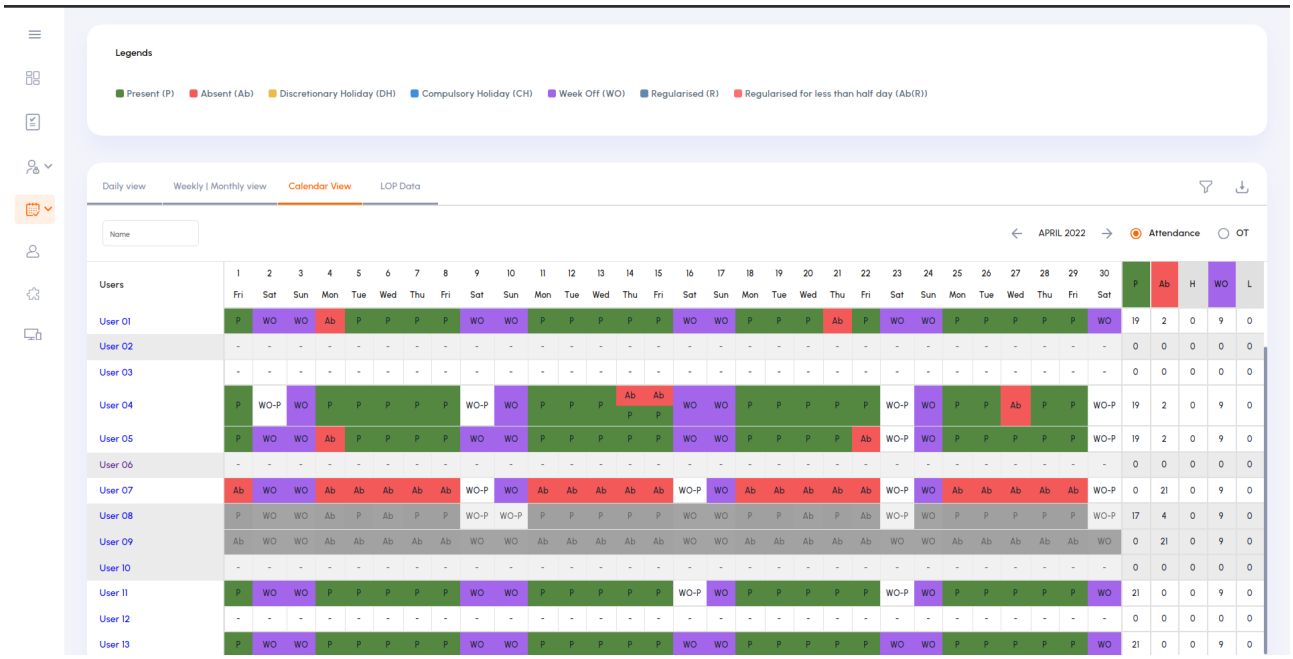
Despite the challenges, the experience of research-oriented learning during my internship was extremely rewarding. It taught me the importance of persistence, critical thinking, and resourcefulness. I learned how to quickly adapt to new technologies and solve complex problems on my own. These skills are crucial in the tech industry, where continuous learning and innovation are key.

Appendix I: Samples of the work done

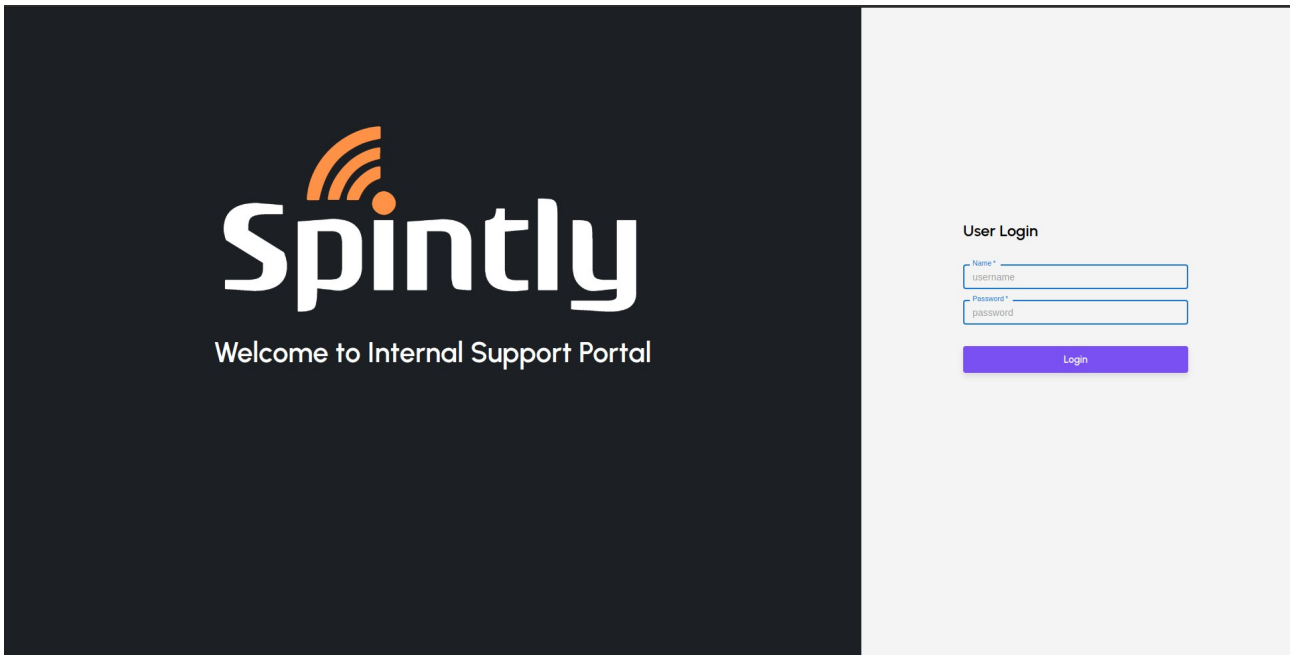


Spintly SAAMS portal Login Page

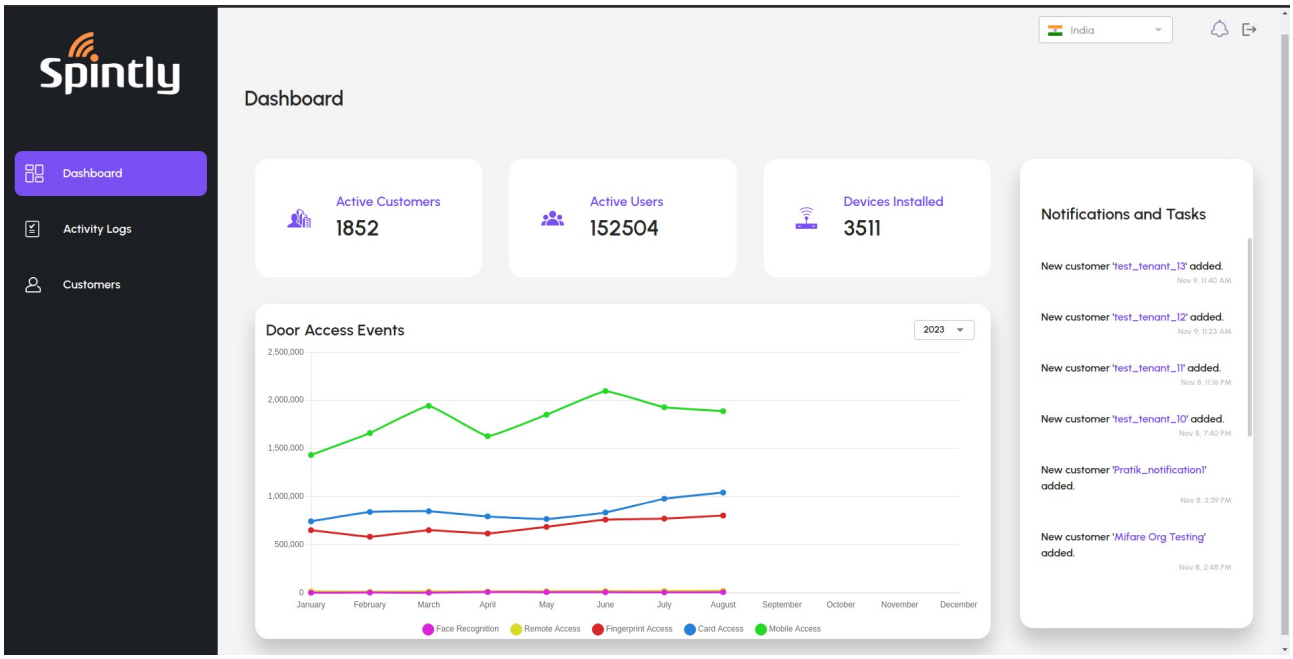
Spintly SAAMS portal Dashboard Page



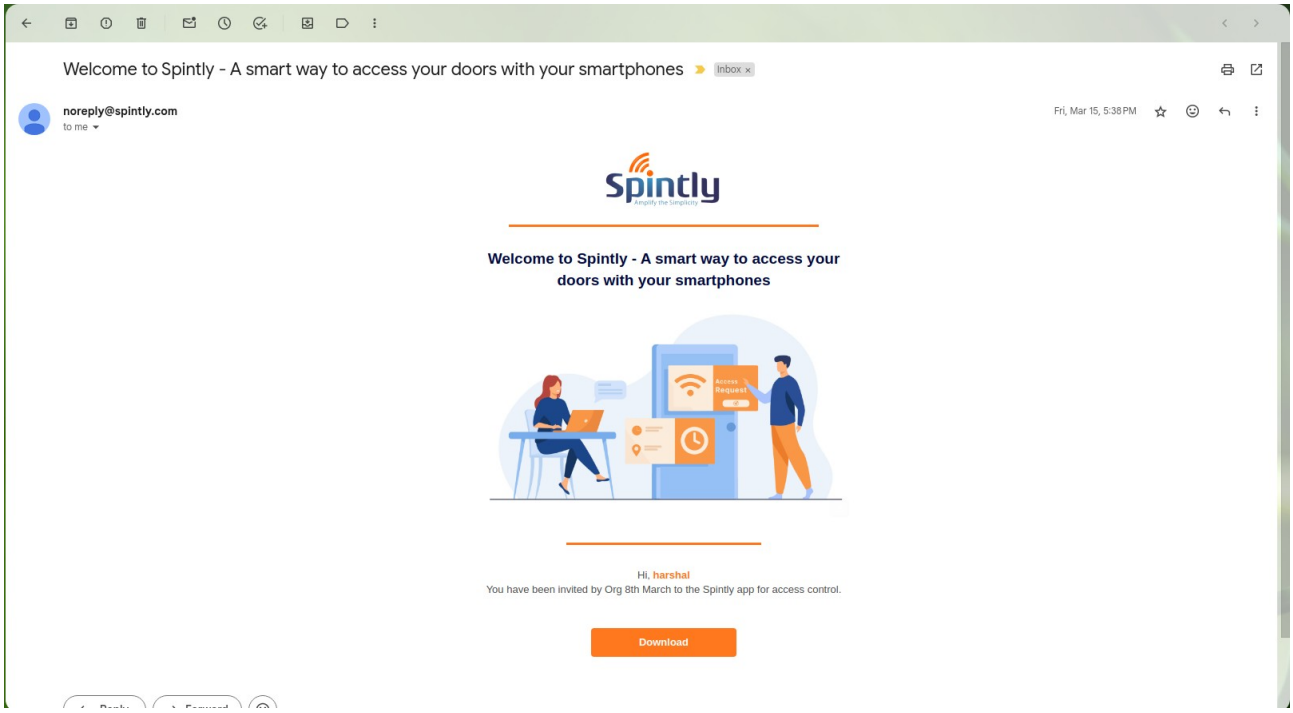
Spintly SAAMS portal Attendance Page



Spintly Internal Support Portal Login Page

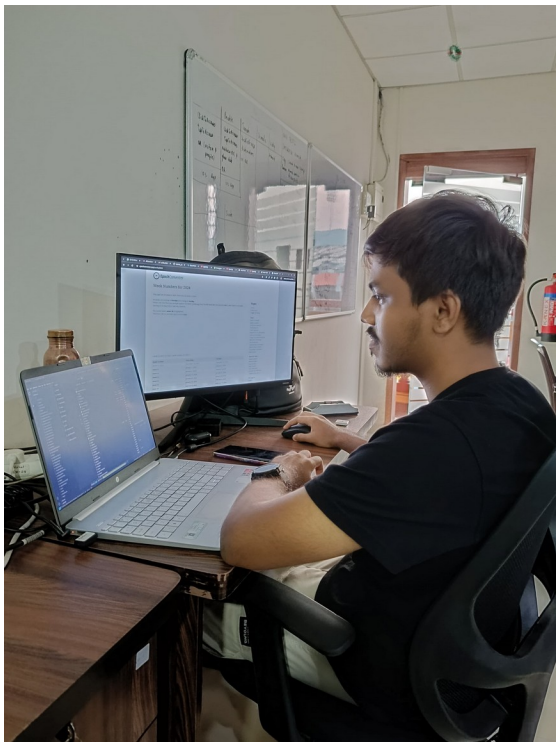


Spintly Internal Support Portal Dashboard Page



HTML Email Sample

Appendix II: Photos while I was at work



Me while working

My cubicle / setup at Spintly Goa Office

Spintly Goa Office Wroking area.

Spintly Goa Office Conference room.

REFERENCES

- https://spintly.com/hardware-products/
- https://legacy.reactjs.org/docs/getting-started.html
- https://kafka.apache.org/documentation/
- https://momentjs.com/docs/
- https://developer.mozilla.org/en-US/docs/Web/API/Server-sent_events/Using_server-sent_events
- https://docs.aws.amazon.com/
- https://www.geeksforgeeks.org/microservices/