TAG-B
PARKING

About TagB

Tag B Group is well-known transportation, parking, garage management, and valet service provider based in Washington, D.C. It wanted to create a scalable solution to maintain a full parking and valet management system for multiple users.

Challenges

- It was difficult for Tag B group to manage parking and valet services without a proper system. There was no way to see real-time data from diverse users, such as customers, parking lot owners, etc.
- Managing a proper record of locations and parking criteria, manually scheduling parking, and allocating valet services on time was difficult.
- Users have to either manage without reservations or suffer from lags in parking lot bookings.
- There was no method to acquire useful insights into user behavior. Therefore, it was impossible to make data-driven decisions for business growth.

Solution

Intuitive and engaging applications

- Mobile and web applications were developed to build a consolidated platform for multiple users (super admin, admins, clients, business users, etc.) We created three applications for three different platforms: Android, an iOS mobile application, and a web application.
- Using cloud-based tools, we developed seamless user experiences across all applications.
- Examples of users include i. customers who want to save time by pre-reserving parking places for their automobiles. ii. Parking lot owners that wish to keep track of what's going on in their lots. iii. Super admins oversee the centralized system, which can be accessible via the web and mobile apps. Other users include enforcement workers, guest users, admins, and more.

Secure and robust architecture

- A system with microservices-based architecture was developed for efficiently handling various third-party as well as custom developed integrations. It includes payment integrations and multiple user modules. The database architecture was designed using the cloud-based service Amazon RDS.

User modules & business dashboards

- Various user modules were created, including super admin, admin, customer, clients, enforcement staff, etc.
- We created a strong set of dashboards, reports, and visualizations that provide precise information for day-to-day activities across the platform. Super admins, for example, will be able to see activities undertaken by clients, admins, customers, and others across the platform. Clients who owns parking lots would be able to see daily activities through the dashboard as well.

Payment & refund management

- Created a safe and reliable payment management system that allows users to add several payment methods. The refunds will be made in the same manner as the original payment. Payment statuses can be updated by super admins or clients as pending, received, refunded, or advanced.

Integration with LPR camera

- We developed a system that can be integrated with a License Plate Recognition(LPR) camera. The camera captures the plate numbers of the vehicles entering and exiting the parking lot. The system either stores it for later review or runs the image through video analytics software. Further, it compares it with license plate numbers database records for verification.

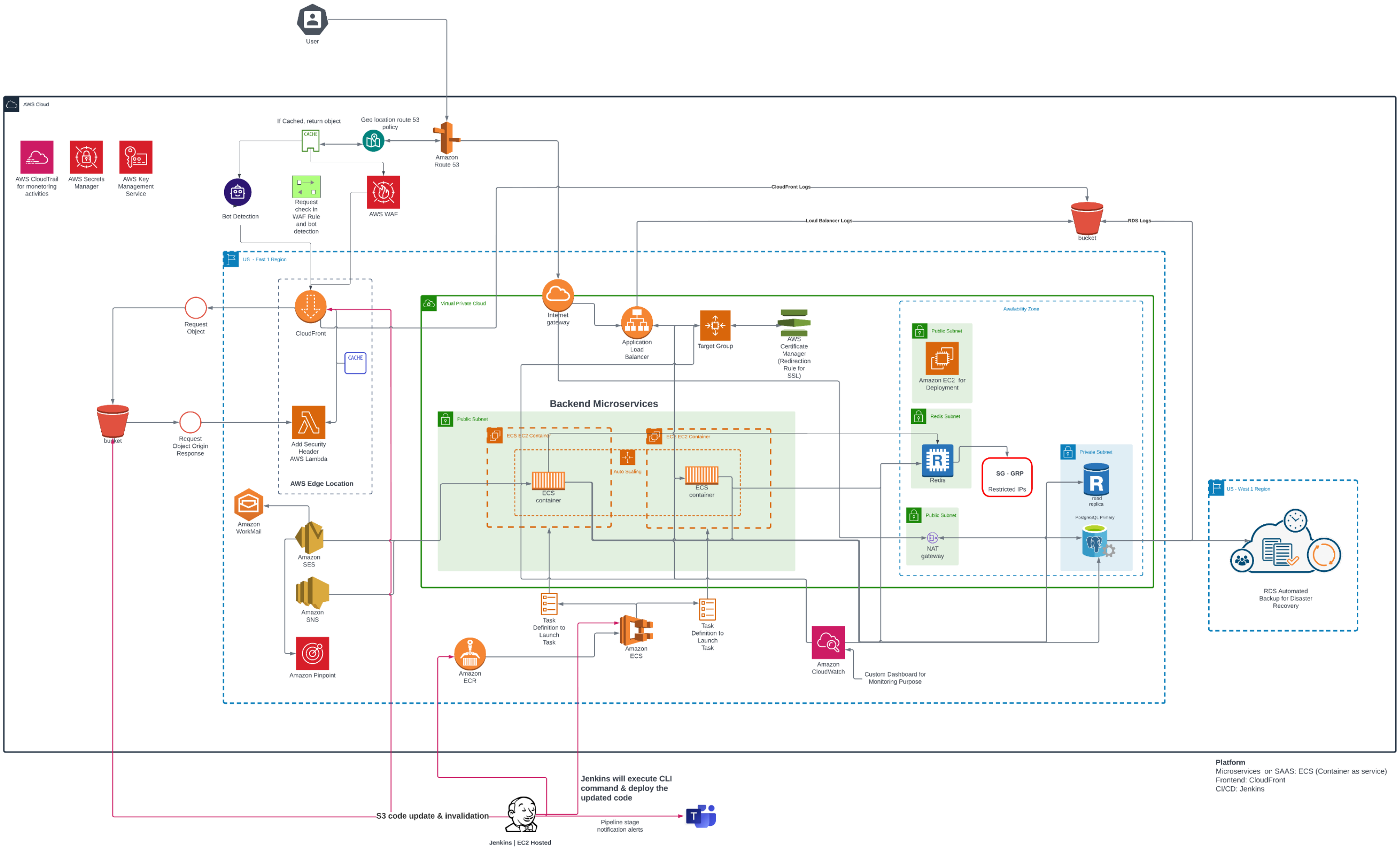
Email service & chat

- Super admins can use chat conversations to communicate with other admins. Email services have been developed to allow Super admins to send mass emails to other users. Emails are also sent to users for various purposes, including the generation of invoices, login credentials, refund statuses, e-pass, and new offers.

Key Results

- Successfully developed a centralized platform for booking parking lots on a global scale.
- Increased visibility into reservation counts and parking spaces in real-time
- Able to make data-driven decisions based on the meaningful insights received from user activities

Architecture Overview



Platform
Microservices on SAAS: ECS (Container as service)
Frontend: CloudFront
CI/CD: Jenkins

Amazon RDS:

- Amazon RDS was employed to store application and user data, including user account information, parking lot information, license plate number recording, and so on.
- In Amazon RDS, restricted access has been kept for specific IP addresses for security purposes.

Amazon ECS:

- Containerized APIs were developed and hosted in Amazon ECS, which can be used by front-end applications.
- For delivering new modules and updates, Amazon Elastic Container Service (ECS) manages the application's microservices backend. Examples include various modules and functionalities such as advanced parking booking, payment and refund administration, parking lot creation, etc.

AWS Task Definition:

- Task definition gives commands to ECS; for example, each task will have certain configurations such as data volumes, memory utilization required, and a number of containers required.

AWS Lambda & Lambda Edge Security Headers:

- The main function of AWS Lambda function is to compress users' profile pictures.
- Lambda edge security headers add a security layer when content is displayed using CloudFront from an S3 bucket. For example, several users will book parking spots from various places using the app, posing a significant security concern.

S3 Bucket:

- Documents belonging to various users, such as administrators, customers, and clients, were stored in an S3 bucket. Clients of Tag B, for example, will have access to data on revenue tracking, parking spots, and booking slots. These individuals will register as customers on the site and will need to upload documents, which will be stored in S3.

Amazon CloudFront:

- CloudFront was used to distribute static and dynamic content across the application frontends. It helped create customized user experiences and deliver content with high speed using its edge computing capabilities across multiple channels.
- CloudFront distribution, for example, pulls files from S3 and displays them in the frontend, as indicated in the diagram. Also, it chooses and restricts access to particular information for specific people. For example, admin users can not access the company's financial information.

AWS SES & SNS:

- AWS SES is a service that sends emails to users. An email will be sent to the user after the a successfull registration with the application, for example. Alternatively, bills will be delivered through email after consumers make a payment.
- Users receive notifications from AWS SNS. Notifications for communications about offers, payment refunds, pass expiration, and other topics will be sent.

Amazon ECR:

- Docker images were stored in Amazon Elastic Registry(ECR) for deployments.

Application Load Balancer:

- In the case of multiple requests from different users worldwide, the application load balancer distributes traffic across various targets, such as EC2 instances of ECS containers in multiple availability zones.

Monitoring:

- CloudWatch is used to keep track of infrastructure-related logs, metrics, and data, while CloudTrail is used to keep track of operational actions in AWS accounts.