.. Description générale de l’architecture

L’architecture technique définie pour le projet *Your Car Your Way* repose sur une approche conteneurisée déployée dans le cloud, en utilisant les services managés d’AWS, et en s'appuyant sur les principes de modularité, de sécurité et de scalabilité.

Le système est divisé en deux grandes parties :

- **Le frontend** développé avec **Angular**, déployé dans une tâche ECS distincte, accessible via un **Application Load Balancer (ALB)**.
- **Le backend**, structuré en **modules fonctionnels** (authentification, réservation, support), regroupés dans un service Spring Boot déployé sous forme d’un conteneur Docker dans une tâche ECS dédiée.

L’infrastructure repose sur :

- **Amazon ECS (mode EC2)** pour l’orchestration des conteneurs,
- **Amazon RDS (MySQL)** pour la persistance des données,
- Et un **Load Balancer** configuré avec **certificats SSL** pour sécuriser les accès HTTPS.

Des **services tiers** tels que Stripe (paiement), WebRTC (visio), et un fournisseur SMTP/Email (notifications) sont intégrés à l’architecture.

.. Liste de contrôle de l’architecture terminée

Composants logiciels

- Les modules métier (**authentification**, **réservation**, **support client**) sont clairement séparés dans le code selon le principe de **responsabilité unique**. Chaque module regroupe ses propres entités, services métier, contrôleurs et règles de validation. Cela facilite la lisibilité, la maintenabilité, et la possibilité future d'extraire un module vers un microservice indépendant sans refactorisation majeure.
- L'API REST est conforme à la spécification décrite dans le document d'architecture. Elle respecte les conventions RESTful (utilisation des méthodes HTTP appropriées, noms de ressources au pluriel, statelessness) et expose des endpoints documentés.
- Le frontend Angular est découplé du backend. Elle consomme l'API via HTTP en utilisant des services Angular (HttpClient) et ne dépend d'aucune logique serveur embarquée. Cela permet une séparation claire des responsabilités entre client et serveur, et facilite le déploiement indépendant de l'interface utilisateur.
- La structure du backend suit les standards Spring Boot (contrôleurs, services, repositories).

Cette organisation garantit la **clarté**, la **testabilité** et la **scalabilité** du code, tout en assurant sa conformité aux bonnes pratiques modernes du développement Java/Spring.

Services ou composants tiers logiciels

- L'intégration avec Stripe respecte les bonnes pratiques de sécurité:
    - Appels sécurisé via https.
    - Validation des signatures de webhooks.
    - Informations de paiement ne transitent jamais par nos serveurs ; seul Stripe les traite.
- Le module support encapsule proprement l'intégration avec le serveur WebRTC.
- L'envoi d'emails s'appuie sur un fournisseur externe compatible SMTP ou API (ex. : Mailgun).
- L'application respecte une **politique claire de gestion des dépendances** :
    - Seules les bibliothèques stables et bien maintenues sont utilisées.
    - Aucune dépendance critique n'est utilisée sans audit de sécurité préalable.
    - Les mises à jour de sécurité sont intégrées dans le processus de maintenance régulier.

Gestion des données

- Les données personnelles (nom, adresse, email…) sont stockées dans une base relationnelle MySQL hébergée sur Amazon RDS. Cette base est configurée dans un **VPC privé**, inaccessible publiquement, avec des règles de pare-feu strictes.
- Le schéma de base de données respecte les bonnes pratiques de modélisation relationnelle (norme 3NF).
- Les mots de passe ne sont jamais stockés en clair et sont hashés avec un algorithme sécurisé.
- Les journaux applicatifs (logs) ne contiennent pas de données sensibles ou les masquent.
- Une politique claire de **durée de conservation des données** est prévue dans les modules concernés, permettant de supprimer ou d'anonymiser les informations personnelles conformément à la réglementation.

Infrastructure

- L'application est conteneurisée avec Docker et déployée sur Amazon ECS (mode EC2).
- Le trafic HTTPS est géré via un Application Load Balancer (ALB) configuré avec un certificat SSL via AWS ACM.
- Les tâches ECS sont configurées pour être redémarrables, scalables et monitorées.
- Les scripts ou pipelines de déploiement CI/CD sont documentés, même si simplifiés pour la version initiale.

Sécurité

- Toutes les communications sont chiffrées : HTTPS pour les API, SMTP sécurisé pour les emails, WebSocket sécurisé pour chat + visio.
- L'accès à la base de données est restreint au sein d'un VPC et à un port spécifique.
- Les rôles utilisateurs sont définis (client, support, agent) avec des règles d'accès associées.

- Le système d’authentification repose sur des tokens JWT, avec durée de validité limitée et renouvellement sécurisé