A white rectangular frame with black border

AI-generated content may be incorrect.

**Project Proposal:**

**Webchat Realtime**

**Serverless**

**Tác giả: Ngô Thành Vinh**

**HO CHI MINH CITY – 16th August, 2025**

MỤC LỤC

[1. Executive Summary 4](#_Toc206427232)

[1.1.Bối cảnh 4](#_Toc206427233)

[1.2.Mục tiêu dự án 4](#_Toc206427234)

[1.3.Phạm vi dự án 4](#_Toc206427235)

[1.4.Lợi ích kinh doanh 5](#_Toc206427236)

[2. Problem Statement 5](#_Toc206427237)

[2.1.Vấn đề hiện tại 5](#_Toc206427238)

[2.2.Những vấn đề có thể gặp 5](#_Toc206427239)

[2.3.Các bên bị ảnh hưởng 6](#_Toc206427240)

[2.4.Hệ quả 6](#_Toc206427241)

[2.5.Cơ hội thị trường 6](#_Toc206427242)

[3. Solution Architecture 7](#_Toc206427243)

[3.1.Sơ đồ kiến trúc AWS 7](#_Toc206427244)

[3.2.Dịch vụ AWS 7](#_Toc206427245)

[3.3.Luồng xử lý 8](#_Toc206427246)

[3.4.Bảo mật 8](#_Toc206427247)

[3.5. Khả năng mở rộng và hiệu năng 8](#_Toc206427248)

[4. Technical Implementation 8](#_Toc206427249)

[4.1. Các giai đoạn triển khai 8](#_Toc206427250)

[4.2. Yêu cầu kỹ thuật 10](#_Toc206427251)

[4.3. Phương pháp phát triển 10](#_Toc206427252)

[4.4. Rollback 11](#_Toc206427253)

[5. Project Timeline & Milestones 11](#_Toc206427254)

[5.1.Timeline 11](#_Toc206427255)

[5.2.Cột mốc quan trọng 11](#_Toc206427256)

[5.3.Phụ thuộc 12](#_Toc206427257)

[5.4.Thời gian dự phòng 12](#_Toc206427258)

[6. Budget Estimation 12](#_Toc206427259)

[6.1.Chi phí AWS cho 1 nhóm nhỏ 12](#_Toc206427260)

[6.2.Chi phí AWS cho nhóm từ 100-500 người 13](#_Toc206427261)

[6.3.ROI (Return on Investment) 14](#_Toc206427262)

[7. Risk Assessment 14](#_Toc206427263)

[8. Expected Outcomes 14](#_Toc206427264)

[8.1. Kết quả mong đợi 14](#_Toc206427265)

[8.2. Giá trị mang lại 15](#_Toc206427266)

# Executive Summary

## 1.1.Bối cảnh

Trong thời đại công nghệ hiện nay nhu cầu trao đổi thông tin nhanh chóng. Các website nhắn tin truyền thống thường yêu cầu duy trì server vật lý hoặc server ảo liên tục chẳng hạn như Messenger hay Whatsapp đẫn đến những vấn đề như sau như chi phí vận hành cao khi người dùng ngày càng tăng. Khó khăn trong việc mở rộng hệ thống khi có nhiều kết nối đồng thời. Nguy cơ downtime do lỗi server làm ảnh hưởng trải nghiệm người dùng. Khó quản lý và đảm bảo chất lượng hình ảnh khi số lượng lớn. Có thể mất hoặc bị lộ dữ liệu người dùng lẫn thông tin đoạn chat do quản lý không tốt.

## 1.2.Mục tiêu dự án

Dự án AWS Serverless Realtime WebChat nhằm xây dựng một website nhắn tin thời gian thực, có khả năng cập nhật linh hoạt và nhanh chóng, bảo mật cao, chi phí tối ưu và triển khai trên hạ tầng serverless của AWS. Là một dự án với những giải pháp mang tính thực tiễn có thể áp dụng vào nhiều trường hợp trong thực tế như hệ thống hỗ trợ khách hàng, nhắn tin nội bộ trong doanh nghiệp và trường học

## 1.3.Phạm vi dự án

Dự án AWS Serverless Realtime WebChat nhằm xây dựng một website nhắn tin thời gian thực được triển khai hoàn toàn trên AWS với các thành phần sau trên AWS: Frontend sẽ được phát triển bằng React và Vite được lưu trữ trên S3 và phân phối tới người dùng thông qua CloudFront . Backend sẽ được xây dựng bằng các Lambda function và giao tiếp thông API Gateway REST API và Websocket. Thông tin người dùng sẽ được quản lý bằng Cognito và đồng bộ thông tin với bảng User trong DynamoDB khi người dùng lần đầu đăng nhập. Tin nhắn sẽ được lưu trữ trong bảng Messages của DynamoDB và có hỗ trợ chức năng gửi, nhận tin nhắn trong thời gian thực. Hình ảnh do người dùng gửi khi nhắn tin hoặc khi cập nhật avatar sẽ được tải lên S3 để lưu trữ và giữ đúng chất lượng hình ảnh, sau đó sẽ được lưu vào bảng dữ liệu tương ứng trong DynamoDB để hiển thị cho người dùng. Quản lý trạng thái online và offline của người dùng thông qua Websocket connect và disconnect. Tích hợp CloudWatch để kiểm tra log khi có lỗi xảy ra.

## 1.4.Lợi ích kinh doanh

Nhờ việc sử dụng mô hình serverless trên AWS nên sẽ giúp tối ưu chí phí vì chỉ phát sinh chi phí khi request nhờ nguyên tác tính phí của AWS là Pay Per Use mà không cần phải duy trì server 24/7 đồng thời cũng hạn chế việc thuê server nhưng không tận dụng hết dẫn tới việc lãng phí tài nguyên. Ngoài ra hệ thống AWS còn có khả năng tự mở rộng khi số lượng người dùng tăng đột biến. Bên cạnh đó trang web còn có tính sẵn sàng cao nhờ CloudFront, API Gateway và DynamoDB giúp đảm bảo website gần như luôn uptime. Cuối cùng là tính bảo mật luôn được đảm bảo nhờ Cognito giúp đảm bảo phải xác thực người dùng theo tiêu chuẩn bảo mật của AWS

# Problem Statement

## 2.1.Vấn đề hiện tại

Ngày nay Các website nhắn tin truyền thống thường được triển khai trên server vật lý hoặc server ảo, ví dụ như Messenger hay WhatsApp. Việc này buộc doanh nghiệp phải duy trì hạ tầng 24/7, dẫn đến nhiều thách thức trong quá trình vận hành.

## 2.2.Những vấn đề có thể gặp

|  |  |
| --- | --- |
| Vấn đề | Nguyên do |
| Chi phí vận hành cao | Do phải duy trì server liên tục 24/7 kể cả khi lượng người truy cập thấp |
| Khó khăn trong việc mở rộng | Hệ thống truyền thống khó đáp ứng khi có nhiều kết nối đồng thời, đặc biệt trong giờ cao điểm có thể dẫn đến ảnh hưởng đến trải nghiệm người dùng |
| Nguy cơ downtime | Một sự cố ở máy chủ chính có thể làm toàn bộ dịch vụ ngừng hoạt động |
| Bảo mật dữ liệu | Việc tự quản lý thông tin người dùng, tin nhắn và hình ảnh dễ dẫn đến nguy cơ rò rỉ hoặc mất mát dữ liệu. |
| Khó kiểm soát chất lượng hình ảnh | Khi có nhiều hình ảnh được lưu trong hệ thống thường hình ảnh sẽ bị giảm chất lượng để tối ưu cho việc tải và lưu trữ |

## 2.3.Các bên bị ảnh hưởng

|  |  |
| --- | --- |
| Đối tượng | Ảnh hưởng |
| Người dùng | Trải nghiệm kém nếu ứng dụng bị gián đoạn, tin nhắn không đến nơi hoặc hình ảnh mờ, tải chậm |
| Doanh nghiệp | Phải chi trả chi phí hạ tầng lớn, đồng thời uy tín bị ảnh hưởng khi dịch vụ không ổn định |
| Đội ngũ kĩ thuật | Tốn nhiều thời gian vào việc quản lý, vận hành máy chủ thay vì tập trung phát triển tính năng mới |

## 2.4.Hệ quả

Nếu tiếp tục vận hành theo mô hình cũ, doanh nghiệp sẽ phải đối mặt với chi phí vận hành tăng theo thời gian, khó tối ưu khi nhu cầu thay đổi. Mất người dùng do dịch vụ không ổn định hoặc không có nhiều điểm cải tiến so với những trang web khác. Rủi ro bảo mật và gián đoạn dịch vụ ảnh hưởng nghiêm trọng đến uy tín của doanh nghiệp

## 2.5.Cơ hội thị trường

Việc áp dụng mô hình serverless của AWS mang lại giải pháp tối ưu:

+Tự động mở rộng khi lượng người dùng tăng, giảm chi phí khi ít người dùng

+Chi phí theo mức sử dụng (pay-per-use), không cần duy trì máy chủ 24/7

+Bảo mật cao nhờ tích hợp sẵn các dịch vụ như Cognito, IAM

+Khả năng ứng dụng rộng rãi: hệ thống chat có thể dùng trong hỗ trợ khách hàng, giáo dục trực tuyến, cộng đồng doanh nghiệp hoặc nhắn tin nội bộ công ty, trường học

# Solution Architecture

## 3.1.Sơ đồ kiến trúc AWS

A screenshot of a video game

AI-generated content may be incorrect.

## 3.2.Dịch vụ AWS

-Amazon S3: Lưu trữ frontend và hình ảnh chat,avatar người dùng

-CloudFront: Giúp phân phối website toàn cầu, giảm độ trễ

-Amazon API Gateway (REST + WebSocket):

+ REST API xử lý request như gửi tin nhắn, đổi avatar,lấy và thông tin người dùng.

+ WebSocket API đảm bảo realtime messaging và trạng thái online/offline.

-AWS Lambda: Chạy code xử lý tin nhắn, cập nhật avatar, quản lý user mà không cần server vật lý. Tự động mở rộng theo số request.

-Amazon DynamoDB: Cơ sở dữ liệu NoSQL, hiệu năng cao, phù hợp với chat real-time.

-Amazon Cognito: Quản lý đăng nhập/đăng ký, xác thực người dùng, tích hợp dễ dàng với frontend.

-Amazon CloudWatch: Theo dõi log, giám sát hiệu năng, phát hiện lỗi.

## 3.3.Luồng xử lý

-Người dùng truy cập ứng dụng webchat qua CloudFront → tải frontend từ S3.

-Khi đăng nhập, Cognito xác thực thông tin và gọi lambda lưu dữ liệu vào bảng User

-Khi gửi tin nhắn:

+ Frontend gọi API Gateway REST API → Lambda xử lý → ghi vào bảng Message

+ Lambda thông báo qua WebSocket API → các client kết nối nhận tin nhắn ngay lập tức.

-Khi gửi hình ảnh hoặc cập nhật avatar:

+ Ảnh được upload vào S3.

+ Lambda cập nhật vào bảng Message hoặc User trong DynamoDB.

-CloudWatch ghi lại toàn bộ log để hỗ trợ giám sát, debugging và theo dõi lỗi.

## 3.4.Bảo mật

-Cognito bắt buộc người dùng phải xác thực sau khi đăng ký mới được sử dụng.

-IAM Roles & Policies phân quyền chi tiết cho Lambda, S3, DynamoDB, tránh lạm quyền.

-CloudFront và S3 bucket policy đảm bảo truy cập hình ảnh công khai có kiểm soát

## 3.5. Khả năng mở rộng và hiệu năng

Lambda: Tự động mở rộng khi số lượng request tăng đột biến.

DynamoDB: Hỗ trợ auto-scaling và đọc/ghi tốc độ cao cho chat realtime.

CloudFront: Giảm tải cho S3, phân phối frontend và hình ảnh nhanh hơn.

WebSocket API: Duy trì kết nối realtime ổn định với hàng nghìn người dùng cùng lúc

# Technical Implementation

## 4.1. Các giai đoạn triển khai

-Giai đoạn 1: Chuẩn bị:

+Phân tích yêu cầu chi tiết và thiết kế kiến trúc hệ thống.

+Cấu hình S3 bucket (1 bucket để lưu ảnh, 1 bucket để host frontend).

+Cấu hình CloudFront để phân phối website toàn cầu.

+Tạo bảng DynamoDB: User và Message.

+Cấu hình Cognito User Pool để quản lý đăng ký/đăng nhập.

-Giai đoạn 2: Phát triển Backend

+Xây dựng Lambda function:

-sendMessage (REST API) – gửi tin nhắn, lưu vào DynamoDB, đồng thời phát qua WebSocket.

-getMessage (REST API) – lấy tin nhắn từ DynamoDB và hiển thị cho người dùng

-getUser (REST API) – lấy thông tin người dùng đã có trong DynamoDB và hiển thị

-me (REST API) – lấy thông tin chủ tài khoản

- createUser (REST API) – lưu thông tin người vào DynamoDB nếu chưa có

-upAvatar (REST API) – tạo preURL S3 để có thể tải avatar lên nhanh chóng

-updateAvatar (REST API) – tải ảnh lên S3, cập nhật avatar vào DynamoDB.

-connect / disconnect – quản lý trạng thái online/offline qua WebSocket.

-Cấu hình API Gateway (REST + WebSocket).

+Tích hợp CloudWatch log cho từng Lambda để theo dõi lỗi

-Giai đoạn 3: Xây dựng Frontend

+Xây dựng giao diện chat: đăng nhập, danh sách bạn chat, khung nhắn tin, update avatar.

+Tích hợp Cognito để xác thực người dùng.

+Gọi API Gateway REST API để gửi/nhận tin nhắn.

+Sử dụng WebSocket để hiển thị tin nhắn thời gian thực và trạng thái online/offline-Giai đoạn 4: Kiểm thử

+Test các chức năng chính: đăng nhập, gửi tin nhắn, nhận realtime, cập nhật avatar, upload ảnh.

+Kiểm tra hoạt động của WebSocket với nhiều client đồng thời

+Kiểm thử bảo mật: chỉ user đã xác thực mới đăng nhập được

-Giai đoạn 5: Triển khai và phân phối

+Deploy frontend lên S3 và cấu hình phân phối qua CloudFront

+Deploy backend

## 4.2. Yêu cầu kỹ thuật

Compute: AWS Lambda (Node.js runtime) dùng để xử lý toàn bộ logic backend.

Storage:

+DynamoDB với hai bảng: User (lưu thông tin người dùng sau khi đăng nhập lần đầu), Messages (lưu trữ nội dung tin nhắn).

+S3: 1 bucket để lưu trữ ảnh/avatar, 1 bucket để lưu và chạy frontend (React + Vite).

Network & API:

+API Gateway REST API (sendMessage, getMessage, getUser, me, createUser, upAvatar, updateAvatar).

+API Gateway WebSocket (connect/disconnect để quản lý trạng thái online/offline và realtime chat).

+CloudFront để phân phối frontend toàn cầu với độ trễ thấp.

Authentication: AWS Cognito User Pool để quản lý đăng nhập/đăng ký và xác thực người dùng.

Monitoring & Logging: CloudWatch để ghi log và phát hiện lỗi của Lambda.

## 4.3. Phương pháp phát triển

-Phương pháp phát triển:

+Áp dụng mô hình Agile với các sprint ngắn (1–2 tuần), giúp bổ sung và điều chỉnh tính năng kịp thời.

+Quản lý mã nguồn bằng Github

+Sử dụng công cụ Postman để test API trong quá trình phát triển.

-Nguyên tắc phát triển:

+Mỗi chức năng backend được xây dựng thành một Lambda function độc lập để dễ bảo trì, dễ mở rộng.

+Frontend giao tiếp với backend hoàn toàn qua API Gateway REST API và WebSocket.

## 4.4. Rollback

+Lambda functions: sử dụng Lambda versioning và alias để quay lại phiên bản ổn định khi phát hiện lỗi sau khi triển khai.

+Frontend (S3 + CloudFront): lưu giữ bản build frontend trước đó trong S3, có thể phục hồi nhanh bằng cách thay thế file lỗi bằng build cũ.

+DynamoDB: thực hiện backup định kỳ bằng AWS Backup; nếu có lỗi khi thay đổi cấu trúc bảng, có thể restore dữ liệu từ bản backup gần nhất.

# Project Timeline & Milestones

## 5.1.Timeline

|  |  |
| --- | --- |
| Tuần | Công việc chính |
| Tuần 1 | - Phân tích yêu cầu chi tiết, thiết kế kiến trúc tổng quan. - Cấu hình AWS cơ bản: S3 (2 bucket: ảnh + frontend), CloudFront, DynamoDB (User, Messages), Cognito  - Xây dựng Lambda functions cơ bản: createUser, me, getUser, sendMessage, getMessage,upAvatar, updateAvatar |
| Tuần 2 | -Phát triển frontend (React + Vite): giao diện login/signup/confirm, danh sách bạn chat, khung nhắn tin, trang thông tin  -Tích hợp Cognito và kết nối REST API |
| Tuần 3 | -Hoàn thiện backend: upAvatar, updateAvatar, phát tin nhắn realtime qua WebSocket  -Kết nối frontend với Websocket  -Deploy frontend lên S3 + CloudFront  -Testing: test Lambda, test realtime với nhiều client  -Chuẩn bị tài liệu, demo, báo cáo. |

## 5.2.Cột mốc quan trọng

-Cuối tuần 1: Hoàn thành hạ tầng AWS và backend cơ bản

-Cuối tuần 2: Hoàn thành frontend với đầy đủ chức năng

-Cuối tuần 3: Hoàn thành dự án và báo cáo

## 5.3.Phụ thuộc

-Backend (Lambda, API Gateway, DynamoDB) phải được triển khai ổn định trước khi frontend có thể tích hợp API.

-Frontend phải được hoàn thành trước khi tích hợp cognito

-CloudFront phụ thuộc vào việc upload frontend hoàn chỉnh lên S3

## 5.4.Thời gian dự phòng

-Dự phòng khoảng 2 ngày trong tuần 3 để xử lý lỗi phát sinh khi kiểm thử hoặc deploy.

-Nếu không phát sinh lỗi, thời gian này sẽ dùng để tối ưu UI/UX hoặc viết báo cáo chi tiết hơn

# Budget Estimation

## 6.1.Chi phí AWS cho 1 nhóm nhỏ

-S3:

+Lưu trữ frontend và ảnh người dùng.

+Với dung lượng nhỏ (dưới 5GB) gần như miễn phí trong Free Tier.

-CloudFront:

+Dùng để phân phối frontend.

+Free Tier có 1TB data transfer, đủ cho giai đoạn thử nghiệm.

-DynamoDB:

+Bảng User và Message. Với ≤ 25 read/write capacity units gần như miễn phí với Free Tier

-Lambda:

+Free Tier 1 triệu request/tháng, 400.000 GB-Seconds

-API Gateway:

+1 triệu request/tháng miễn phí.

-Cognito:

+Free Tier 50.000 MAU (monthly active user).

-CloudWatch:

+Log cơ bản miễn phí (dưới 5GB).

Tổng chi phí dự kiến cho: khoảng 0 – 10 USD, nằm trong Free Tier.

## 6.2.Chi phí AWS cho nhóm từ 100-500 người

-S3 (lưu ảnh + frontend):

+Giả sử trung bình mỗi người upload 10 ảnh (100 KB/ảnh).

+Dung lượng: 100 × 10 × 100 KB = 100 MB (cho 100 user) đến 500 MB (cho 500 user).

+Chi phí S3: ~0.023 USD/GB/tháng gần như không đáng kể (< 1 USD).

-CloudFront (phân phối website và ảnh):

+100 – 500 người, mỗi người 10 MB băng thông/tháng.

+Tổng 1 – 5 GB → nằm trong Free Tier (1 TB).

+Chi phí: khoảng 0 - 1 USD.

-DynamoDB (User + Message):

+User: 500 record.

+Message: giả sử trung bình 100 tin nhắn/ngày/người → 500 × 100 × 30 = 1.5 triệu tin nhắn/tháng.

+Với DynamoDB on-demand:

-1.5 triệu write request ≈ 1.5M × 1 WCU = ~1.5M WCU.

-1.5 triệu read request ≈ 1.5M × 1 RCU = ~1.5M RCU.

+Tính chi phí: khoảng 1.25 USD/million WCU/RCU đến khoảng 3.75 USD/tháng.

-Lambda:

+Giả sử 500 user hoạt động, mỗi user 100 tin nhắn/ngày.

+Tổng request ~ 50,000/ngày = 1.5 triệu/tháng.

+Free Tier: 1 triệu request, vượt 0.5 triệu thì chi phí thêm rất nhỏ (~0.2 USD)

+Tính GB-Seconds: với function nhẹ (128 MB, 100 ms) gần như miễn phí.

-API Gateway:

+REST + WebSocket.

+1.5 triệu request/tháng.

+Free Tier 1 triệu request.

+0.5 triệu request × 3.5 USD/million = ~1.75 USD.

-Cognito:

+Free Tier: 50,000 MAU thì 500 user vẫn miễn phí.

-CloudWatch:

+500 MB log/tháng khoảng 0.5 USD.

Tổng chi phí ước tính: khoảng 6 – 8 USD/tháng cho 100 – 500 user

## 6.3.ROI (Return on Investment)

Ngắn hạn: chi phí rất thấp, dễ dàng mở rộng cho nhóm nhỏ và vừa.

Dài hạn: nếu user tăng lên hàng nghìn người, chi phí sẽ tăng nhưng vẫn thấp hơn nhiều so với việc duy trì server truyền thống 24/7.

# Risk Assessment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rủi ro | Loại | Mức độ | Kế hoạch giảm thiểu | Phương án dự phòng |
| Downtime do cấu hình AWS sai | Kỹ thuật | Trung bình | |  | | --- | |  |   Test từng thành phần trước khi tích hợp | Rollback Lambda version, phục hồi cấu hình S3/CloudFront trước đó |
| Quá tải khi nhiều user cùng online | Vận hành | Thấp | |  | | --- | |  |   Dùng DynamoDB on-demand và Lambda auto-scale | Cấu hình cảnh báo qua CloudWatch Alarm khi vượt ngưỡng thì auto-trigger AWS Lambda scale-out hoặc bật thêm region dự phòng. |
| Bảo mật (truy cập trái phép) | Kỹ thuật | Cao | Dùng Cognito + IAM phân quyền | Reset mật khẩu, thu hồi token, backup dữ liệu |
| Lỗi frontend khi build/deploy | Kỹ thuật | Trung bình | |  | | --- | |  |   Giữ bản build ổn định | |  | | --- | |  |   Re-deploy bản cũ từ S3 |

|  |
| --- |
|  |

# Expected Outcomes

## 8.1. Kết quả mong đợi

+Hệ thống webchat realtime chạy ổn định trên AWS, bảo mật và mở rộng dễ dàng.

+Đáp ứng đầy đủ chức năng: đăng nhập, gửi/nhận tin nhắn realtime, upload avatar, hiển thị trạng thái online/offline.

+Uptime 99% trong thời gian hoạt động

## 8.2. Giá trị mang lại

+Ngắn hạn: một hệ thống chat phục vụ cho một nhóm nhỏ có thể vận hành được tốt

+Dài hạn: có thể mở rộng để trở thành nền tảng chat cho doanh nghiệp, hỗ trợ khách hàng hoặc giáo dục trực tuyến với chi phí thấp và khả năng mở rộng cao.