Project Report II – Cloud Computing

Cloud-Based Meal Recipe App

Team number: 38

Student names: Chua Zi Jian (2202172),

Ong Jing Wei (2202024),Terence Tiang Jin Chai (2201920),

Lim Pei Sheng (2200468), Teo Teng Wee Shannon (2201034),

Ryan Lua Jun Xiang (2202819)

05 April 2025, 10:51 am

1. Introduction

This report presents the development of a Cloud-Based Meal Recipe App, which is a platform designed to allow users to create, upload and share meal recipes. The main goal of this project is to provide a user-friendly, accessible, reliable and secure environment to allow users to discover and create recipes anywhere, anytime using a cloud-based infrastructure.

The scope of this project includes the design and implementation of a scalable web application using AWS services to handle data storage and content delivery. The platform focuses on usability, data reliability, and scalability, ensuring that users experience seamless access regardless of growing traffic or data volume.

The application utilizes a **full-stack architecture**, combining:

- **Frontend**: HTML, CSS, and JavaScript.
- **Backend**: PHP connection and server-side functions / applications.
- **Database**: MySQL on AWS RDS for storing recipes and user data.
- **Cloud Infrastructure**: Built on AWS using services such as EC2, Elastic Load Balancer (ELB), Auto Scaling Group, S3, RDS, and VPC.

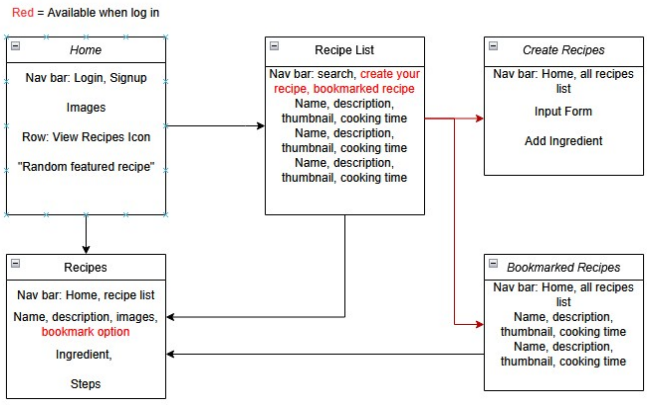
The website allows users to:

- Search and view recipes
- Create accounts
- Upload their own recipes (with images)
- Bookmark recipes for easy access

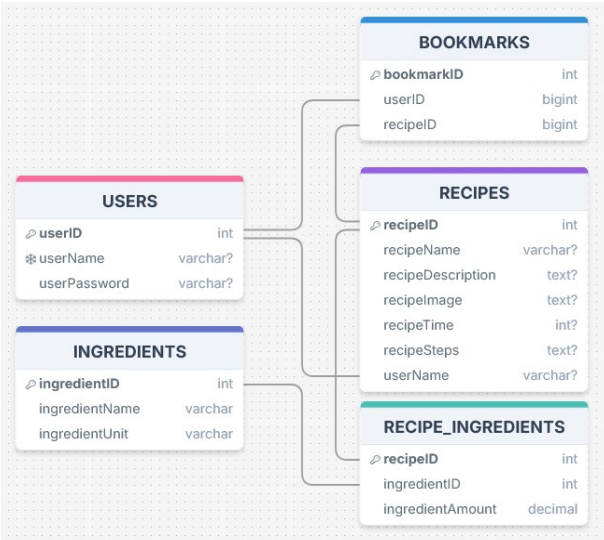
Team Members and Role Breakdown

Team Members	Roles
Chua Zi Jian	Frontend Content Design
Lim Pei Sheng	Frontend Content Design
Terence Tiang Chai Jin	Frontend Content Design
Ong Jing Wei	Cloud Service
Ryan Lua Jun Xiang	Frontend Content Design
Teo Teng Wee Shannon	Database

Content Flow



Database Structure

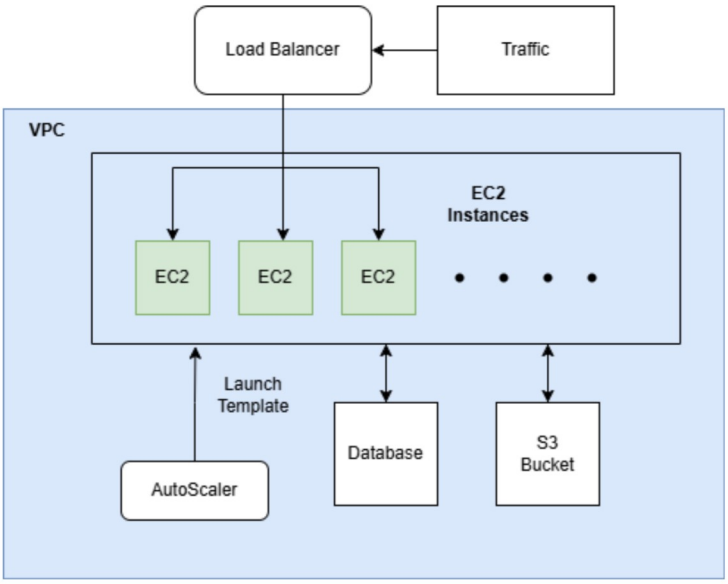


2. Architecture Design and Solutions

2.1 System Architecture

The platform is built on a cloud-based design making use of multiple AWS cloud services.

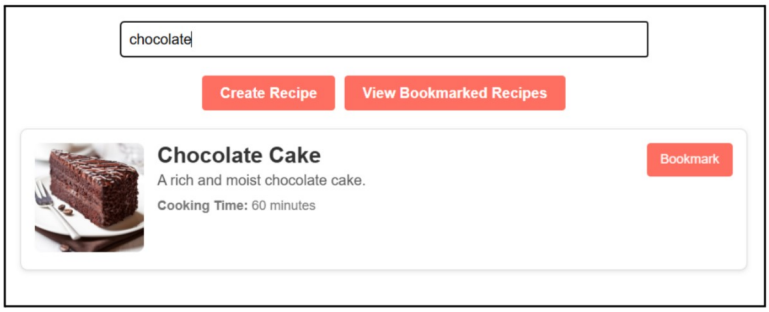
- **EC2**: Host HTML/PHP web pages and backend applications that handles form submissions, user actions and database/storage calls.
- **ELB**: Distributes traffic evenly across all EC2 instances.
- **Auto Scaling Group**: Ensure system responsiveness during demand spikes by scaling the number of EC2 instances. These EC2 instances are also load balanced by the ELB
- **RDS**: Hosts a relational MySQL database for storing user and recipe data.
- **S3**: Stores other data, such as recipe images. Recipe images are stored in a public subfolder, with storage of private data possible.
- **VPC**: Allows the EC2 instances and the RDS/S3 to communicate with each other privately and securely.



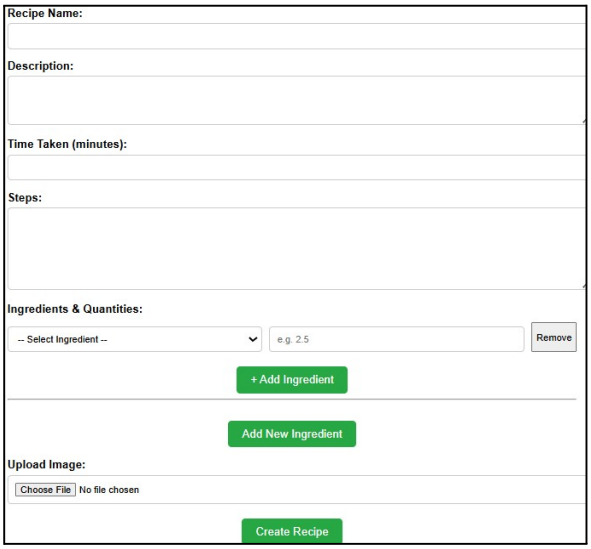
2.2 Workflow scenarios

Here are some possible workflows that a user may take.

1. **User Signup and Login:**
    a. User's sign up input for Username and password(hashed) is stored into database
    b. User's login input for Username and password is verified against database
    c. If the User is verified, a PHP session is started. User ID and username will be set for that session.
2. **Recipe search:**
    a. Application queries RDS and S3 to display all recipes and thumbnails
    b. User searches for recipe via name or description
    c. Application queries RDS to display recipes with matching name or description



3. **Recipe bookmarking:**
    a. Users can click on the bookmark button on the recipes list page or a specific recipes' page.
    b. Bookmarked recipes are linked to their User data in the RDS
    c. Application queries RDS and S3 to display all of User's bookmarked recipes on the Bookmarked recipes page.
4. **Recipe Creation:**
    a. User fills up the recipe creation form.
    b. Recipe data is sent to the backend and stored in RDS, images are uploaded to S3.



2.3 Deployment

On a manually created EC2 instance by the autoscaler, a local folder contains the cloned GitHub repository storing the Web App source files. It is then copied into the apache document root folder deploy. Instances also need to be running a presigned url generator application, which is done using PM2.
An AMI is then created from this instance, which is then used by a launch template for the autoscaler. The launch template has a simple script to make sure the presigned url generator is running using PM2.

3. Discussions

Using AWS features, we have built a scalable and reliable cloud-based web application.

3.1 Scalability

The application has been designed to handle varying levels of user traffic by leveraging AWS services that support dynamic scaling.

- **AWS Auto Scaling Group** automatically adjusts the number of EC2 instances based on real-time demand, ensuring responsiveness during traffic spikes.
- **Elastic Load Balancers (ELB)** distribute traffic evenly across active EC2 instances, preventing overloading of any single instance.
- **AWS S3** provides scalable storage for user-uploaded images, ensuring the application can store images without concerns about storage limitations.

3.2 Reliability

- **Multi-AZ RDS deployment** is utilized to maintain database availability in the event of a failure.
- **Multiple EC2 instances** are deployed to guarantee application availability in case any instance fails.
- **Auto Scaling** automatically replaces unhealthy EC2 instances to maintain the

desired capacity and system performance.

- **Automated RDS backups** and **S3 redundancy** ensure that data and media are safely stored and can be recovered in case of failure.
- **Health checks through the Elastic Load Balancer (ELB)** route traffic only to healthy EC2 instances, ensuring uninterrupted service.

3.3 Security

- **Password hashing** in database for safe storage.
- **VPC** ensures communication between back end services, such as RDS and S3 Buckets, are secure.
- **Security Groups** help manage the connections between services that are interacting with public traffic, and services that are private.

3.4 Limitations

While the current implementation achieves the core goals of being functional, scalable and reliable, there are some areas where the application can be improved by further leveraging other AWS services.

Currently, the user login/signup is managed through a custom implementation of PHP and SQL. Where User passwords are hashed and stored in a RDS database table along with the username. While functional, this approach lacks scalability and may introduce potential security risks.

AWS Cognito could be integrated to handle user login/signup securely. Features like multi-factor authentication, password recovery could upgrade the application's scalability and security.

AWS IAM could also be used to manage access to AWS resources based on user roles. For example, specific permissions can be assigned to different users or groups, such as granting an administrator full access to all AWS services, while restricting a developer's access to only certain resources like EC2 instances or S3 buckets. This ensures that users have only the permissions they need to perform their tasks, following the principle of least privilege.

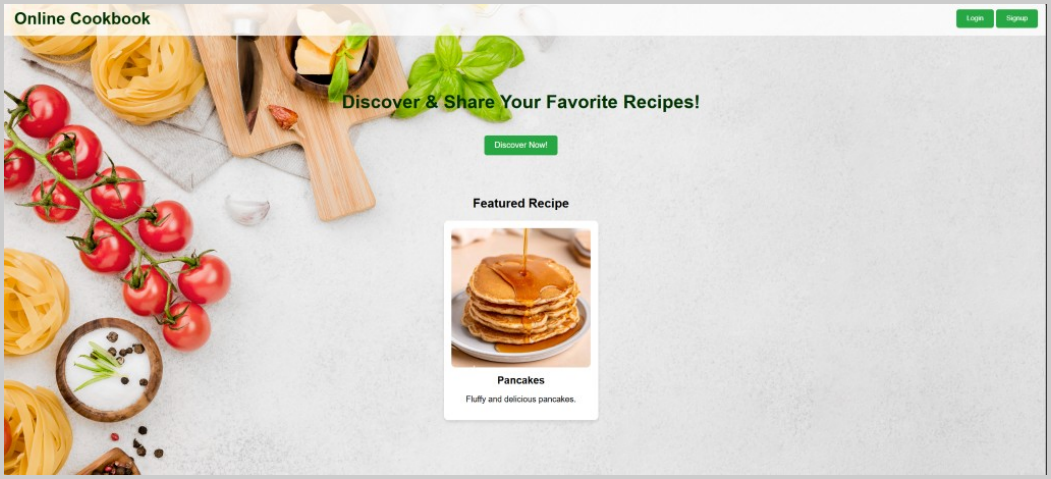
4. Reflection and takeaways

This project offered valuable experience in designing and deploying a full-stack cloud application using AWS services. The group gained hands-on experience with:

- Web development languages and cloud integration
- Designing a scalable and reliable architecture
- Managing backend logic and SQL databases
- Setting up and configuring AWS services like EC2, RDS, and S3

Although some proposed features like AWS Cognito and ElastiCache were not implemented due to time constraints, the deployed website meets its core objectives and serves as a solid foundation for future upgrades.

We are proud of how the system performs under various loads and how smoothly the core features operate, from uploading recipes to searching and sharing them.



5. Contribution from team members

SN	Name of Team Member	Student ID	Responsible Components	Contribution (%)
1	Chua Zi Jian	2202172	Create Recipe	16.7
2	Lim Pei Sheng	2200468	Recipe Page	16.6
3	Terence Tiang Chai Jin	2201920	Home Page	16.7
4	Ong Jing Wei	2202024	Cloud Service	16.7
5	Ryan Lua Jun Xiang	2202819	Recipe List Page	16.6
6	Teo Teng Wee Shannon	2201034	Database	16.7
			Total:	100%

Reference

Appendix I

END OF REPORT