Team Our Daily Bread (ODB): Content Recommendation Engine

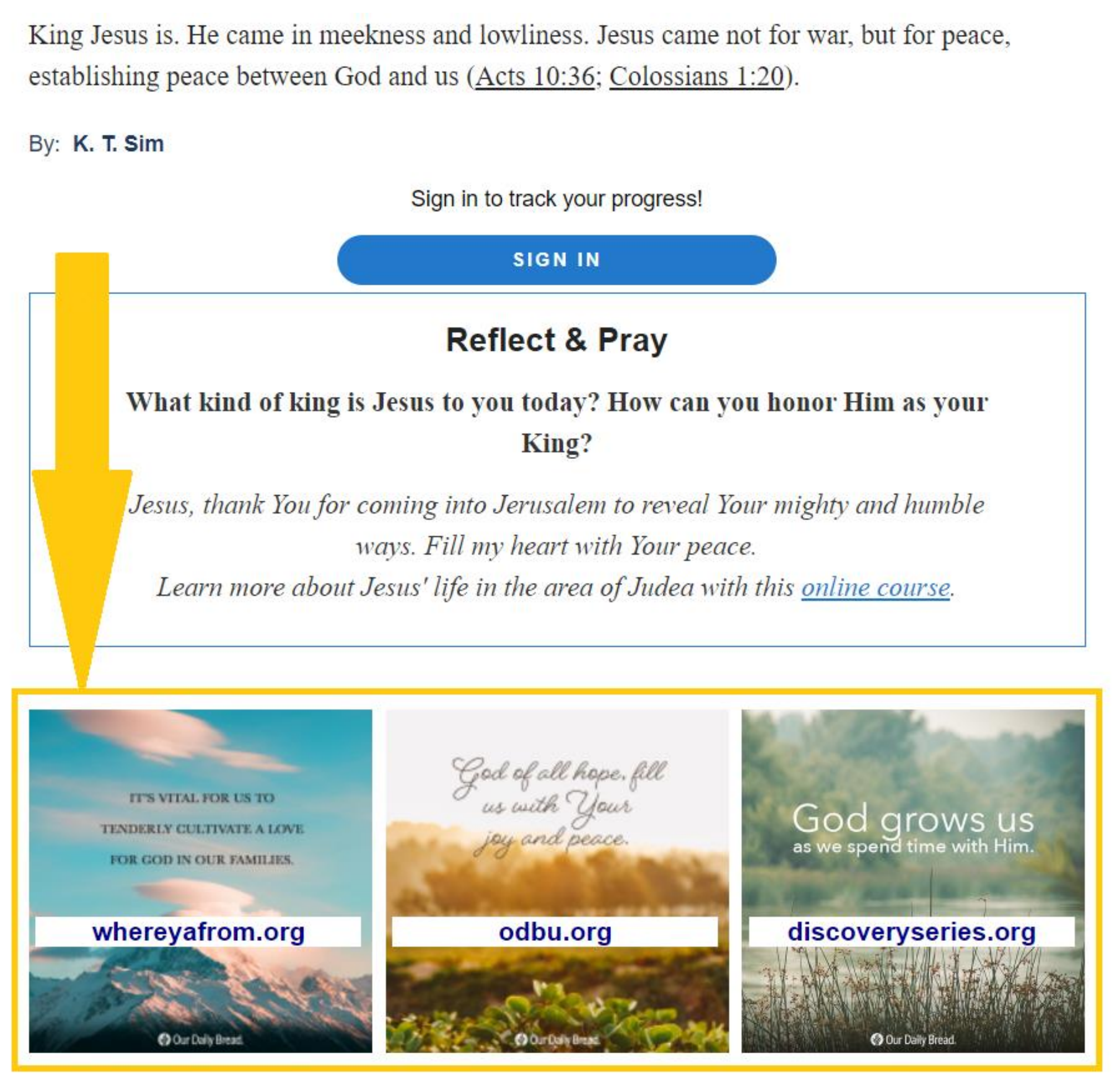
Client: Our Daily Bread Ministries (Joe Fahnestock)

Jacob Johnson, Thomas Lobbestael, Michael Dykema, Emily Linderman

Overview

Goal:

Build a recommendation engine using Amazon Web Services (AWS) that will provide content recommendations to users who visit their daily devotional. Current user and site data will be collected, transformed, and processed into a machine learning model, with results accessible from an API, and monitoring for costs and errors.



King Jesus is. He came in meekness and lowliness. Jesus came not for war, but for peace, establishing peace between God and us (Acts 10:36; Colossians 1:20).

By: **K. T. Sim**

Sign in to track your progress!

SIGN IN

Reflect & Pray

What kind of king is Jesus to you today? How can you honor Him as your King?

Jesus, thank You for coming into Jerusalem to reveal Your mighty and humble ways. Fill my heart with Your peace.

Learn more about Jesus' life in the area of Judea with this online course.

IT'S VITAL FOR US TO TENDERLY CULTIVATE A LOVE FOR GOD IN OUR FAMILIES.
whereyafrom.org
God of all hope, fill us with Your joy and peace.
odbu.org
God grows us as we spend time with Him.
discoveryseries.org
Our Daily Bread

Figure 1: Mock of content cards for recommended content

User Data

Problem:

ODB currently utilizes Google Analytics to track site and page statistics for their different websites. The past and future data needs to be stored and processed for machine learning. Additionally, page content scraped from websites will need to be stored.

Solution:

- A data lake stores the data of various formats and schemas
- Google Analytics data is streamed into the data lake on a regular basis
- Different metrics such as user information and page hits are aggregated for machine learning
- Scraped site content is stored in database and moved into data lake for later use

Recommendation Engine

Problem:

When a user visits ODB's daily devotional, links to other content should appear at the bottom of the page. These recommendations should be based on content popularity, other user behaviors, and content similarity

Solution:

- Use collaborative filtering to find users with similar profiles and behavior to find new content for a user
- When not enough user data is present, find similar content using topic analysis
- Topic Analysis is performed by processing website page text and creating topics by finding similar words
- Pages are grouped by topic and stored for retrieval on site visit
- Recommendations are calculated when user visits page and an API is called

Topic 0 words	Topic 0 weights	Topic 1 words	Topic 1 weights	Topic 2 words	Topic 2 weights
psalm	0.5	dark	1.0	news	0.9
lord	0.3	light	0.7	good	0.7
prais	0.3	isaiah	0.6	joy	0.4
prayer	0.3	great	0.4	heard	0.3
day	0.3	water	0.3	proclaim	0.3

Figure 2: Sample topics from NMF modelling

Monitoring

Problem:

AWS services are billed on a usage basis, so alarms need to set to warn when costs are higher than expected. Additionally, services such as the API need to be monitored for high numbers of errors to ensure the pipeline is working correctly.

Solution:

- Each service has alarms registered for different metrics such as run time, endpoint calls, storage used, etc.
- Emails are sent to individuals at ODB in case alarm is tripped
- A dashboard displays the most important metrics so the health of the system can be monitored at a glance
- Reports are exportable in a JSON format for later analysis

Technical Details

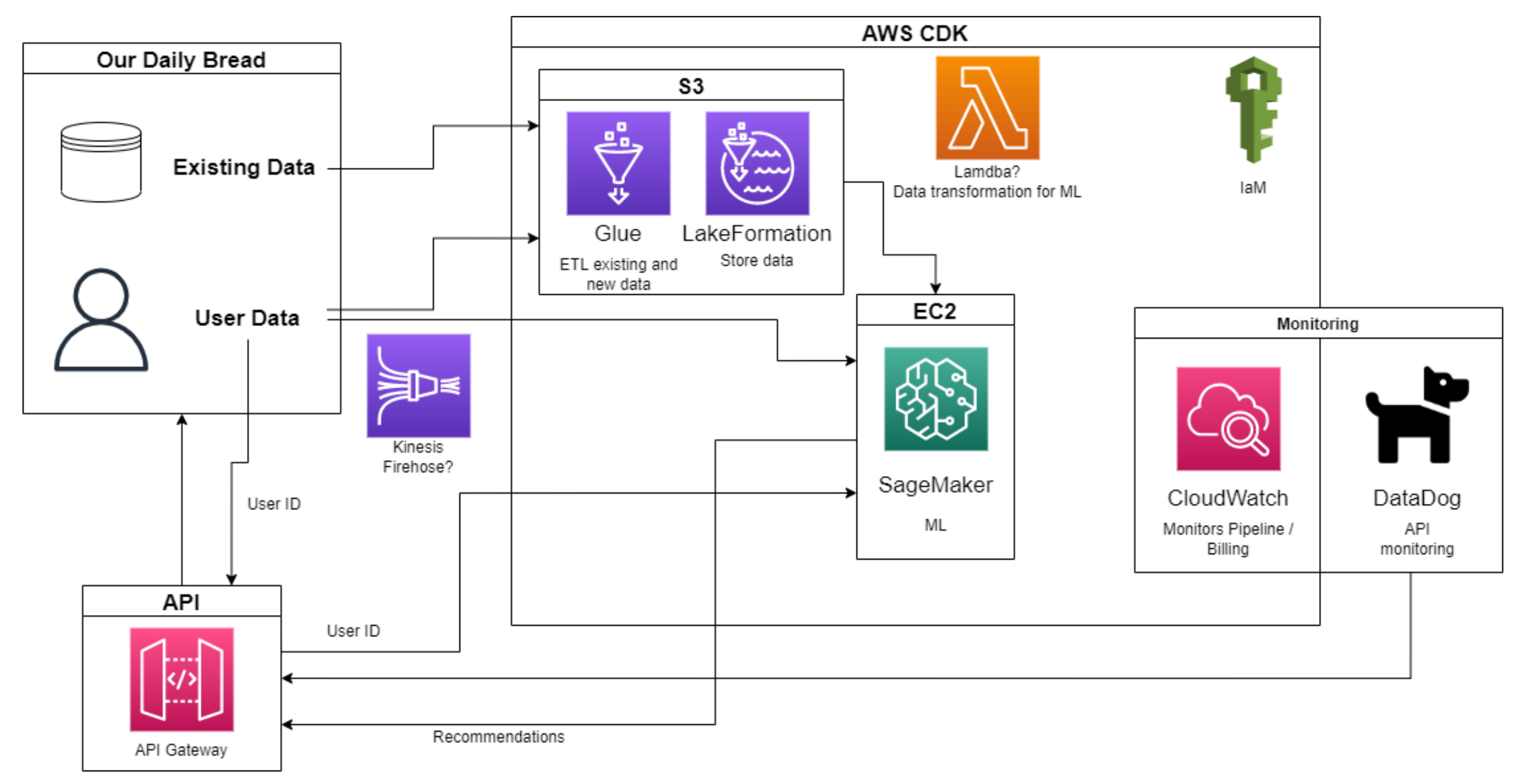


Figure 3: System Architecture

Data

- AWS LakeFormation for data lake creation
- AWS AppFlow streams real-time Google Analytics data into data lake

Recommendation Engine

- AWS SageMaker hosts Docker image of Tensor Flow model and non-negative matrix factorization (NMF) model

API

- AWS ApiGateway hosts a REST API that uses AWS Lambda functions to pull recommended content

Monitoring

- AWS CloudWatch monitors all other AWS services for surpassing billing thresholds and error counting

Web Scraping

- AWS EC2 for hosting and running Python script
- AWS RDS hosts MySQL database for results

Future Work

The web page UI of this project was not the team's responsibility and still needs to added to ODB's devotional page. With more data on specific user behavior the collaborative filtering model's performance would improve substantially.

Acknowledgements

We would like to thank Our Daily Bread Ministries for providing us with this project and their assistance during development. In particular, we would like to thank Joe Fahnestock for his patience, hard work, and help throughout the whole project.