dripdrop

Team 25

Zachary Foote, Kaden Wingert, Kolby Kucera, Logan Roe, Elyse Kriegel, Gavin Rich

Faculty Advisor: Simantra Mitra

1. Project Overview

2. Mobile App

3. Infrastructure Design

4. AI Model

5. Backend Design

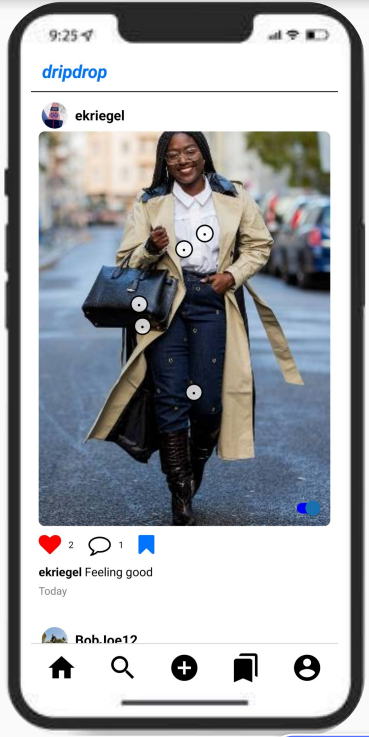
6. Testing & Security

7. Conclusion

Project Overview

What is *dripdrop*?

- *dripdrop* is a social media platform exclusively for outfit sharing
- Bridges the gap between fashion inspiration & budget alternatives



Our Vision

1. **AI** Suggestions & Clothing Identification
    a. **AI-powered** similar outfit recommendations
    b. Clothing identification via **image processing**
2. Outfit Sharing
    a. Curate your style by liking, commenting, saving, and following
3. Powered by AWS
    a. Fast and scalable hosting service to support high traffic

Requirements

Functional Requirements

- User Account Management
- Outfit Sharing
- Interactive Posts
  - Like, comment, & view markers
- Add Product Descriptions to Posts
- Search for Users & Posts
- **AI-Based** Clothing Recommendations

Non-Functional Requirements

Performance	< 500 ms response time
Security	Users must be 13+ API calls only be verified users
Scalability	1,000 concurrent sessions
Usability	Simple & intuitive navigation

1. Project Overview

2. Mobile App

3. Infrastructure Design

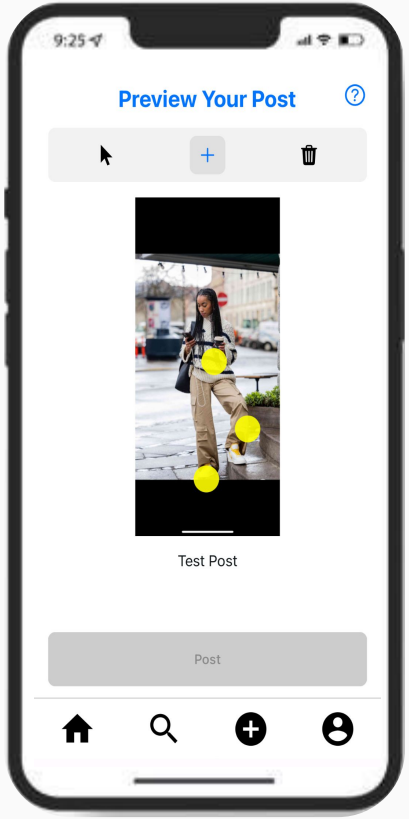
4. AI Model

5. Backend

6. Testing & Security

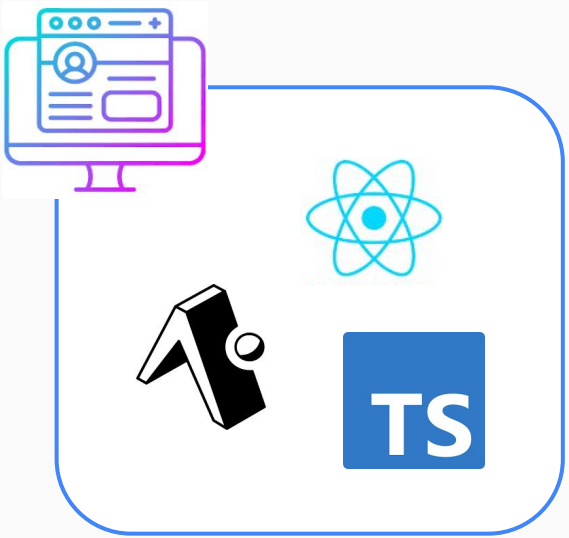
7. Conclusion

Frontend Technologies



Built with React Native and Expo platforms

Written in Typescript



Mobile App Demo

1. Project Overview

2. Mobile App

3. Infrastructure Design

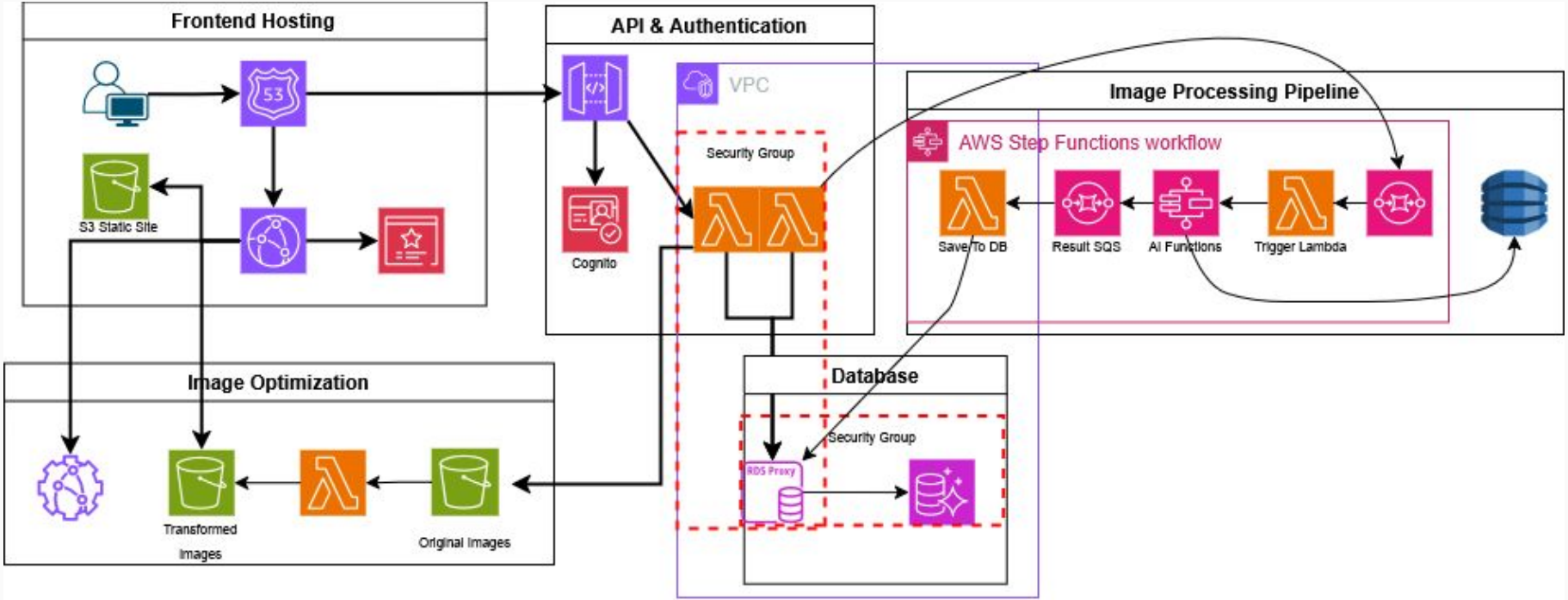
4. AI Model

5. Backend Design

6. Testing & Security

7. Conclusion

Infrastructure



1. Project Overview

2. Mobile App

3. Infrastructure Design

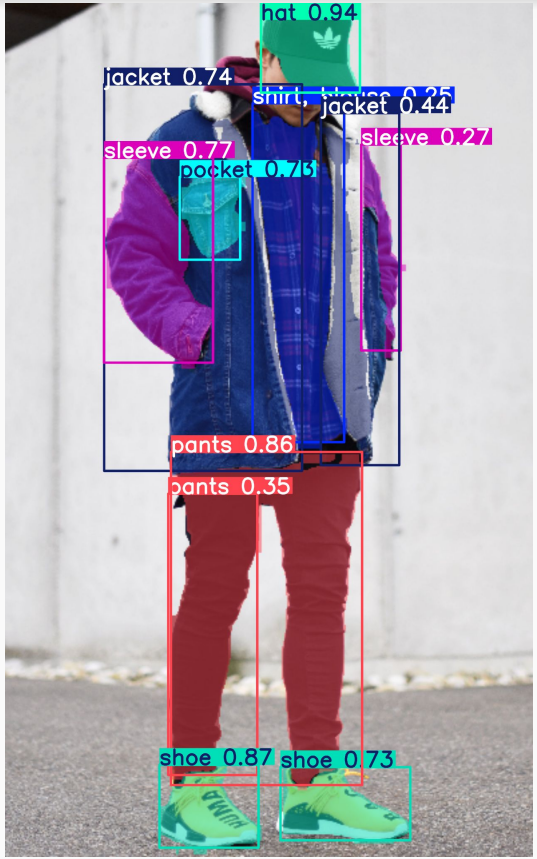
4. AI Model

5. Backend Design

6. Testing & Security

7. Conclusion

AI



Classification Lambdas

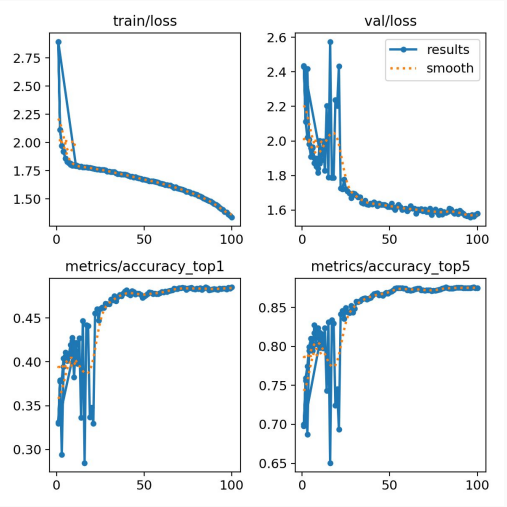
- YOLOv11 classification model
- Adds attributes on segmented clothing items

Segmentation Lambda

- Uses YOLOv11 segmentation model
- Detects clothing items, masks, and bounding boxes

Training

- Trained on the Fashionpedia dataset
- Used Nova computer



1. Project Overview

2. Mobile App

3. Infrastructure Design

4. AI Model

5. Backend Design

6. Testing & Security

7. Conclusion

Database

MySQL Database

Relational Database

Familiarity

AWS compatibility

Amazon Aurora

Performance

Scalability

Security

Lambda Functions

Serverless

Pay-as-you-go

57 Lambda Functions

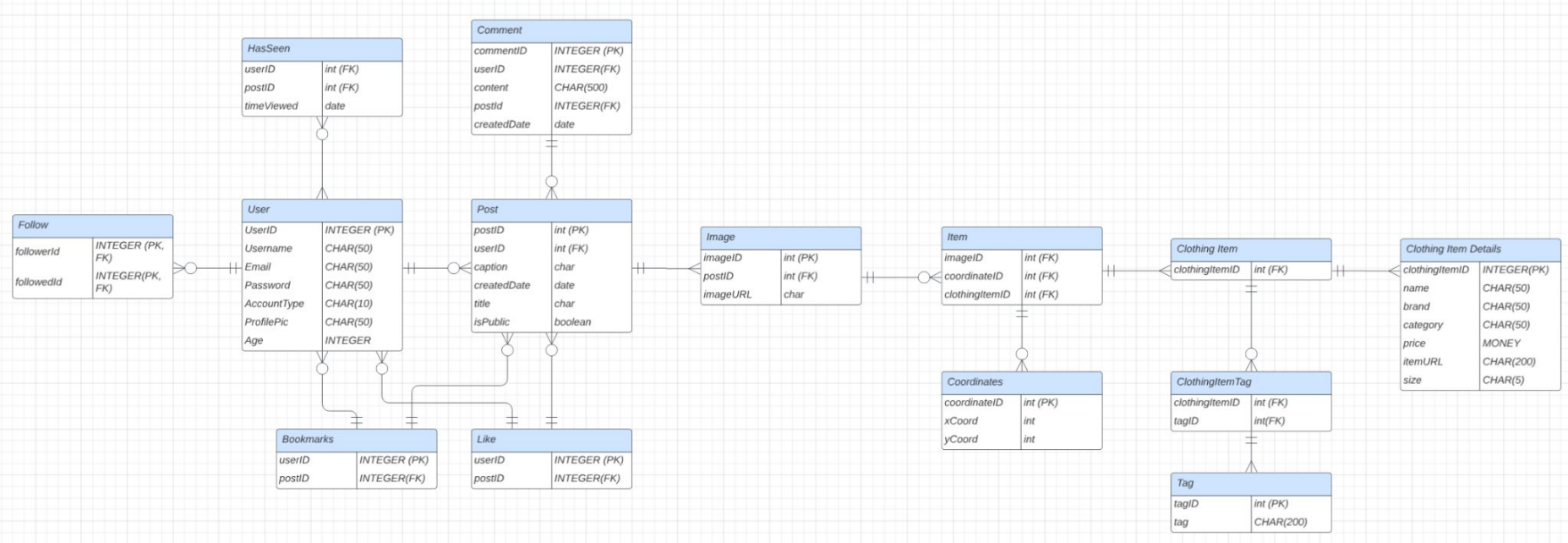
SQLAlchemy

Object Relational Mapping

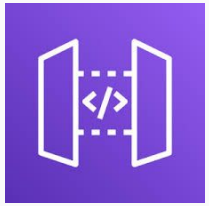
Python Programming

No raw SQL statements

ER Diagram


HasSeen
userID int (FK)
postID int (FK)
timeViewed date
Comment
commentID INTEGER (PK)
userID INTEGER(FK)
content CHAR(500)
postId INTEGER(FK)
createdDate date
Follow
followerId INTEGER (PK, FK)
followedId INTEGER(PK, FK)
User
UserID INTEGER (PK)
Username CHAR(50)
Email CHAR(50)
Password CHAR(50)
AccountType CHAR(10)
ProfilePic CHAR(50)
Age INTEGER
Post
postID int (PK)
userID int (FK)
caption char
createdDate date
title char
isPublic boolean
Image
imageID int (PK)
postID int (FK)
imageURL char
Item
imageID int (FK)
coordinateID int (FK)
clothingItemID int (FK)
Clothing Item
clothingItemID int (FK)
Clothing Item Details
clothingItemID INTEGER(PK)
name CHAR(50)
brand CHAR(50)
category CHAR(50)
price MONEY
itemURL CHAR(200)
size CHAR(5)
Coordinates
coordinateID int (PK)
xCoord int
yCoord int
ClothingItemTag
clothingItemID int (FK)
tagID int(FK)
Tag
tagID int (PK)
tag CHAR(200)
Bookmarks
userID INTEGER (PK)
postID INTEGER(FK)
Like
userID INTEGER (PK)
postID INTEGER(FK)

API



Amazon API Gateway

Fully Managed API Service

Lambda Integration

Pay-as-you-go

Typescript CDK Code

Easily add/remove endpoints

Deploy changes in minutes

API Structure

1. Project Overview

2. Mobile App

3. Infrastructure Design

4. AI Model

5. Backend Design

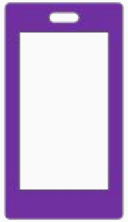
6. Testing & Security

7. Conclusion

Testing

Integration Testing

- ❏ Pytest Library
- ❏ Integrates with Cognito
- ❏ Tests API Health
- ❏ Checks Database
- ❏ Lambda Code



Frontend Testing

- ❏ Jest library (typescript)
- ❏ Mocks backend data
- ❏ Simulates frontend actions
- ❏ Validates frontend flow

User Testing

- ❏ Team members
- ❏ Friends & Roommates
- ❏ Bug Detection
- ❏ Usability

Security

Frontend

- ❏ Age Verification
- ❏ Email Validation
- ❏ Password Hiding

Backend

- ❏ Password Encryption
- ❏ AWS Cognito Authentication
- ❏ User Confirmation
- ❏ IAM & Network Security
- ❏ Data & API Protection

1. Project Overview

2. Mobile App

3. Infrastructure Design

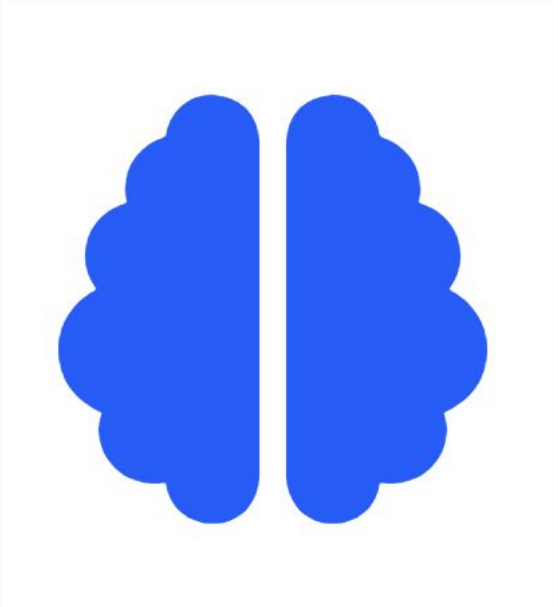
4. AI Model

5. Backend Design

6. Testing & Security

7. Conclusion

Challenges



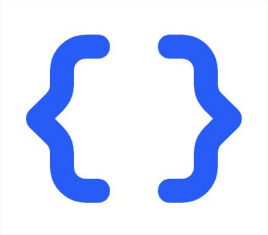
Technical

- Learning AWS technologies
- Using and updating Expo project
- Maintaining dependencies

Team

- Unclear vision from start of project
- Difficulties figuring out feature prioritization and user needs

Learning Takeaways



Technical

- ❏ React
- ❏ React Native
- ❏ TypeScript
- ❏ AWS + Cloud Computing
- ❏ SQLAlchemy
- ❏ AI Model Creation and Training

Team

- ❏ Time Management
- ❏ Assigning tasks by strengths
- ❏ Communication
- ❏ Accountability

Thank you!

Questions?