

# Deliverable #2

SE 3A04: Software Design III – Large System Design

**Tutorial Number:** T02

**Group Number:** G02

**Group Members:**

- Alvin Qian (Team Leader)
- Ryan Kumar
- Lucas DeBoer
- Varun Pathak
- Maya Azar
- Moustafa Moustafa

## IMPORTANT NOTES

- Please document any non-standard notations that you may have used
  - *Rule of Thumb*: if you feel there is any doubt surrounding the meaning of your notations, document them
- Some diagrams may be difficult to fit into one page
  - Ensure that the text is readable when printed, or when viewed at 100% on a regular laptop-sized screen.
  - If you need to break a diagram onto multiple pages, please adopt a system of doing so and thoroughly explain how it can be reconnected from one page to the next; if you are unsure about this, please ask about it
- Please submit the latest version of Deliverable 1 with Deliverable 2
  - Indicate any changes you made.
- If you do NOT have a Division of Labour sheet, your deliverable will NOT be marked

# 1 Introduction

This document offers a complete overview of SoleMate, an innovative platform developed to accurately identify sneakers using a photograph. It details the essential components, their interactions, and the data structures utilized. SoleMate uses modern software practices to significantly boost sneaker identification accuracy, enhance user experience, and optimize overall system performance.

## 1.1 Purpose

The primary objective of this document is to clearly identify the architectural framework and specific system components of SoleMate. It aims to provide a solid foundation for software developers, system architects, project managers, and stakeholders involved in the design, development, and deployment phases. By clearly identifying the system's architecture and functionality, this document fosters effective collaboration, informed decision-making, and streamlined maintenance processes.

## 1.2 System Description

SoleMate is a sneaker identification platform built upon a hybrid architectural model that combines Blackboard and Repository architectures. The Blackboard architecture enables specialized modules, such as machine learning classifiers, image recognition algorithms, and database querying components, to collaboratively refine sneaker identification results through iterative information exchange. This cooperative process markedly improves the system's accuracy and reliability. The Repository architecture complements this approach by systematically managing data storage and retrieval, including detailed sneaker databases, historical search records, user accounts, and expert-generated feedback. The integration of these architectures creates an intuitive and efficient user experience, supporting robust functionality, scalability, and responsiveness.

## 1.3 Overview

This document is written to clearly identify SoleMate's architectural design, subsystem roles, and component interactions. Section 2 presents an extensive discussion of the chosen system architecture, elaborating on the reasoning behind the selected architectural frameworks. Section 3 delves deeper into individual subsystems, clarifying their specific responsibilities and detailing how they interact within the overall platform. Section 4 utilizes Class Responsibility Collaboration (CRC) cards to precisely outline each class's role, responsibilities, and interactions. Finally, Section 5 concludes with a concise summary, emphasizing key insights and suggesting areas for future development and refinement.

# 2 Analysis Class Diagram

Note that SME stands for SoleMate Management Engine and SSD stands for SoleMate Shoe Database. These were terms initially introduced in the previous report.

# 3 Architectural Design

This section should provide an overview of the overall architectural design of your application. Your overall architecture should show the division of the system into subsystems with high cohesion and low coupling.

## 3.1 System Architecture

SoleMate utilizes a Blackboard Architecture integrated with a Repository Style Design to enable efficient shoe identification and data management. The Blackboard Architecture allows multiple expert modules to contribute their analyses iteratively, refining identification results until a final decision is reached. When a user submits a shoe image or textual description, the SoleMate Management Engine (SME Controller) acts as the central hub, posting the input to a shared data space where different expert modules—such as machine learning-based classification, image recognition, and database matching—collaborate to analyze the

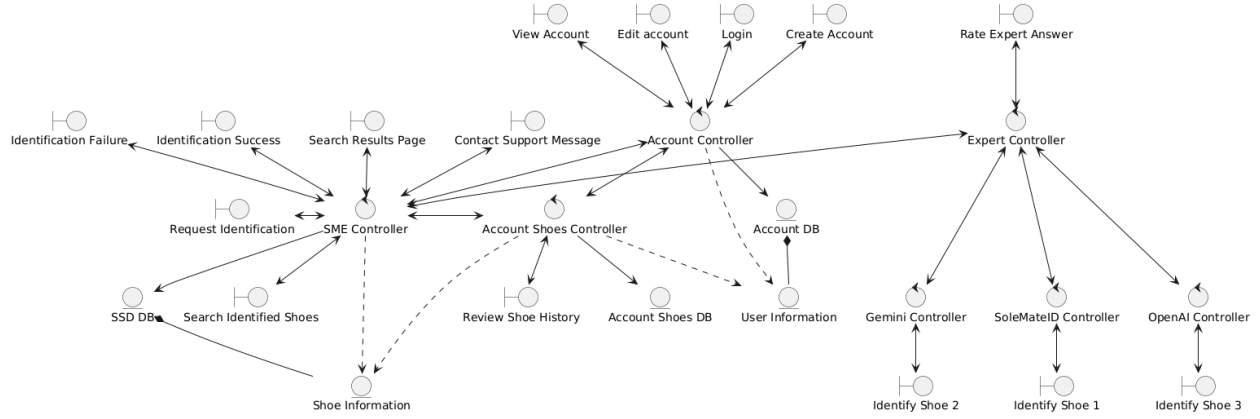


Figure 1: Analysis Class Diagram

data. This approach supports a flexible and scalable identification process, allowing for the addition of new expert modules over time and improving recognition accuracy through iterative refinement.

The Repository Architecture complements the Blackboard system by ensuring structured and persistent data storage. The SSD Database serves as the repository for storing identified shoes, historical searches, and expert ratings, while the Account Shoes Database and Account Database manage user-related data, such as past searches, stored shoe collections, and authentication details. This repository-style approach allows different subsystems to access data concurrently, ensuring efficient retrieval and updates without conflicts. By combining real-time collaborative processing with structured data management, SoleMate ensures both adaptability and long-term reliability in handling sneaker identification and user interaction.

The Blackboard Architecture was chosen because it provides a collaborative problem-solving environment, where different expert modules can iteratively refine identification accuracy. This design is well-suited for SoleMate, as shoe recognition requires integrating various techniques, such as AI models, database lookups, and image recognition. The Repository Architecture was selected to efficiently manage large datasets, ensuring seamless access to stored information while allowing the system to scale as more users and shoe data are added. Together, these architectures provide a flexible, scalable, and data-driven solution for SoleMate’s sneaker identification platform.

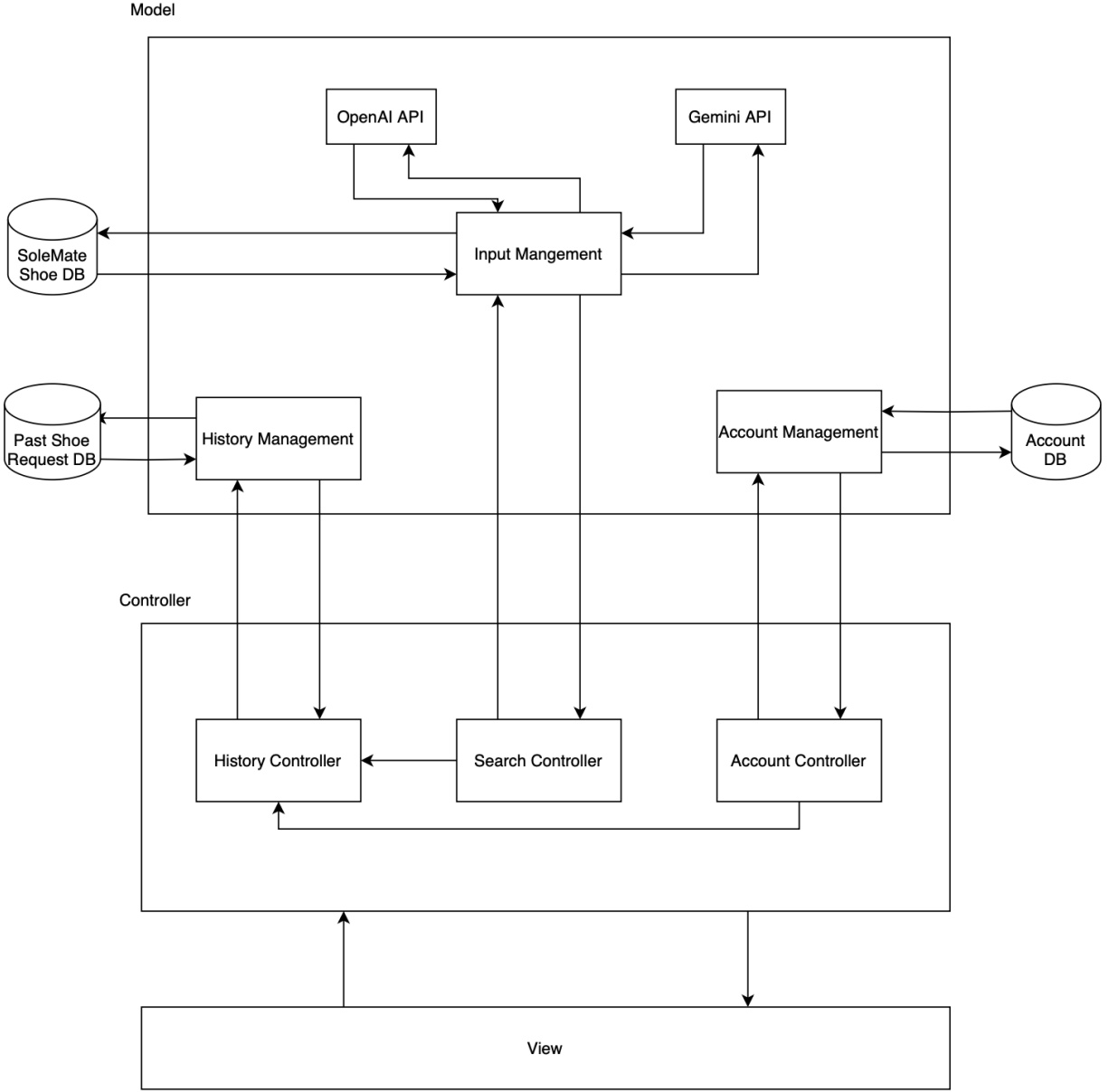


Figure 2: System Architecture Diagram

Several alternative architectural styles were considered but ultimately discarded. The Model-View-Controller (MVC) Architecture was initially evaluated, but it was not ideal for SoleMate, as the system prioritizes data-driven processing and expert collaboration over UI management. The Pipe and Filter Architecture was also considered but was eliminated because it enforces a sequential flow of data, which does not align with SoleMate’s iterative expert-based approach to identification. Lastly, the Batch Sequential Architecture was rejected due to its reliance on processing data in predefined batches, as SoleMate requires real-time responses for shoe identification requests. By combining Blackboard and Repository Architecture, SoleMate successfully integrates adaptive intelligence with structured data storage, creating an optimal solution for sneaker identification and management.

## 3.2 Subsystems

SoleMate utilizes three main subsystems, incorporating both a Blackboard and Repository-like architecture. The SoleMate system integrates a Blackboard Architecture with a repository-style design to streamline shoe identification and user interactions. At the core of the system is the SoleMate Management Engine (SME Controller), which acts as the central coordination unit. When a user submits a shoe identification request through the Request Identification module, the SME Controller processes the request by interacting with different subsystems. The Identification Subsystem plays a key role in this architecture, pulling data from the SSD Database and coordinating with various processing modules to refine identification accuracy. The Account Controller and Account Shoes Controller further support user interactions by managing account-related operations, such as creating, editing, and logging into accounts, as well as storing previously identified shoes for future reference.

The repository-style architecture is evident in how SoleMate manages persistent data across various subsystems. The SSD Database functions as the primary data repository, storing shoe identification results, historical searches, and expert ratings. The Account Shoes Database and Account Database store user-specific data, ensuring seamless retrieval of past interactions. The Support and Feedback System allows users to rate expert answers and interact with customer support in case of identification failures. This feedback is stored and utilized by the SME Controller to enhance the system's future performance. By leveraging a Blackboard approach for expert-driven identification and a repository structure for efficient data storage, SoleMate ensures a robust and scalable platform that enhances user experience and engagement.

## 4 Class Responsibility Collaboration (CRC) Cards

Class Name: Account Controller (Controller)	
Responsibility:	Collaborators:
Knows Edit Account	Edit Account
Knows Login	Login
Knows Create Account	Create Account
Knows View Account	View Account
Knows Account DB	Account DB
Knows User Information	User Information
Knows Account Shoes Controller	Account Shoes Controller
Knows SME Controller	SME Controller

Class Name: Account Shoes Controller (Controller)	
Responsibility:	Collaborators:
Knows Account Controller	Account Controller
Knows User Information	User Information
Knows Shoe Information	Shoe Information
Knows Account Shoes DB	Account Shoes DB
Knows Review Shoe History	Review Shoe History
Knows SME Controller	SME Controller

Class Name: Account DB (Entity)	
Responsibility:	Collaborators:
Knows Account Controller	Account Controller
Knows User Information	User Information

<b>Class Name:</b> Edit Account (Boundary)	
<b>Responsibility:</b>	<b>Collaborators:</b>
Knows Account Controller Handles click-event of "Save" button	Account Controller

<b>Class Name:</b> Login (Boundary)	
<b>Responsibility:</b>	<b>Collaborators:</b>
Knows Account Controller Handles click-event of "login" button	Account Controller

<b>Class Name:</b> Create Account (Boundary)	
<b>Responsibility:</b>	<b>Collaborators:</b>
Knows Account Controller Handles click-event of "Create Account" Button	Account Controller

<b>Class Name:</b> View Account (Boundary)	
<b>Responsibility:</b>	<b>Collaborators:</b>
Knows Account Controller Handles click-event of "View Account" button Displays User Information	Account Controller

<b>Class Name:</b> Account Shoes DB (Entity)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows Account Shoes Controller	Account Shoes Controller

<b>Class Name:</b> User Information (Entity)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows Account DB Knows Account Controller Knows Account Shoes Controller Knows user information like username, password, gender etc.	Account DB Account Controller Account Shoes Controller

<b>Class Name:</b> Review Shoe History (Boundary)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows Account Shoes Controller	Account Shoes Controller

<b>Class Name:</b> Rate Expert Answer (Boundary)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows SME Controller Handles user input for rating experts' answers Displays the rating results	SME Controller

<b>Class Name:</b> Shoe Information (Entity)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows SSD DB Knows SME Controller Knows Account Shoe Controller Displays shoe information for users	SSD DB SME Controller Account Shoe Controller

<b>Class Name:</b> Search Identified Shoes (Boundary)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows SME Controller Displays retrieved and identified shoes Handles sorting options for identified shoes	SME Controller

<b>Class Name:</b> Request Identification (Boundary)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows SME Controller Handles initiation of search for shoe identification	SME Controller

<b>Class Name:</b> Search Results Page (Boundary)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows SME Controller Displays organized search data to users Handles filtering and sorting of search results Displays links to detailed shoe information	SME Controller

<b>Class Name:</b> SME Controller (Controller)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows Identification Success Knows Contact Support Message Knows Search Identified Shoes Knows Search Results Page Knows Account Controller Knows Account Shoe Controller Knows Shoe Information Knows SSD DB Knows Rate Expert Answer Knows Request Identification Knows Identification Failure	Identification Success Contact Support Message Search Identified Shoes Search Results Page Account Controller Account Shoe Controller Shoe Information SSD DB Rate Expert Answer Request Identification Identification Failure

<b>Class Name:</b> Identification Failure (Boundary)	
<b>Responsibility:</b>	<b>Collaborators:</b>
Knows SME Controller Displays identification failure notification to user	SME Controller

<b>Class Name:</b> Identification Success (Boundary)	
<b>Responsibility:</b>	<b>Collaborators:</b>
Knows SME Controller Displays search results after identification	SME Controller

<b>Class Name:</b> Contact Support Message (Boundary)	
<b>Responsibility:</b>	<b>Collaborators:</b>
Knows SME Controller Displays the support contact information	SME Controller

<b>Class Name:</b> SSD DB (Entity)	
<b>Responsibilities:</b>	<b>Collaborators:</b>
Knows SME Controller Knows Shoe Information Knows detailed shoe attributes Knows price trends and availability Handles updates to shoe-related information Handles database indexing and queries	SME Controller Shoe Information

## A Division of Labour

- Introduction (Section 1): Ryan Kumar
- Analysis Class Diagram (Section 2): Lucas DeBoer
- Architectural Design (Section 3): Alvin Qian & Varun Pathak
- Class Responsibility Collaboration (CRC) Cards (Section 4): Moustafa Moustafa & Maya Azar

Signatures:

