# CHAPTER 01: INTRODUCTION

# 

This chapter is a part of our Software Requirement Specification and Analysis for the project **“Cistron”**, a **single**-**cell RNA**-sequencing **analysis tool**. In this chapter, I focus on the intended audience for this project.

## **1.1** **Purpose**

This document is the simple outline of the Software Requirement Specification and Analysis of our project for Software Project Lab – 3 titled as “Cistron”. It contains functional, non – functional and supporting requirements and establishes a requirements baseline for developing the system. The Software Requirement Specification holds the requirements that were collected from Research Fellows , Faculty of Biology, Medicine and Health ,The University of Manchesters. The requirements contained in the SRS are independent, uniquely numbered and organized by topic. The SRS serves as an official means of communicating user requirements to the developer and provides a common reference point for both the developer team and the stakeholder community. The SRS will evolve over time as users and developers work together to validate, clarify and expand its contents.

## **1.2** **Intended Audiences**

This SRS is intended for several audiences including the customers as well as the project designers, developers. The customer will use this SRS to verify that the developer team has created a product that the customer finds acceptable. The designers will use this SRS as a basis for creating the system’s design. The designers will continually refer back to this SRS to ensure that the system they are designing will fulfill the customer’s demands. The developers will use this SRS as a basis for developing the system’s functionality. The developers will link the requirements defined in this SRS to the software they create to ensure that they have created a software that will fulfill all of the customer’s documented requirements. When portions of the software are complete, the developer will run their tests on that software to ensure that the software fulfills the requirements documented in this SRS. The testers will again run their tests on the entire system when it is complete and ensure that all requirements documented in this SRS have been fulfilled.

## 

## **1.3** **Conclusion**

## This analysis of the audience helped me to focus on the users who will be using my analysis. This overall document will help each and every person related to this project to have a better idea about the project.

# **CHAPTER 02: BACKGROUND STUDIES**

In this chapter, we succinctly discuss the Inception part of the SRS for our “Cistron”.

## **2.1 INTRODUCTION**

Inception is the beginning phase of requirements engineering. It defines how a software project gets started and what the scope and nature of the problem to be solved are. The goal of the inception phase is to identify concurrent needs and conflicting requirements among the stakeholders of a software project. At project inception, I establish a basic understanding of the problem, the people who want a solution, the nature of the solution that is desired and the effectiveness of preliminary communication and collaborations between the other stakeholders and the software team.

To establish the groundwork, I have worked with the following factors related to the inception phases:

* List of stakeholders
* Recognizing multiple viewpoints
* Working towards collaboration
* Requirements questionnaire

### **2.1.1 LIST OF STAKEHOLDERS**

# **CHAPTER-03: PROJECT DESCRIPTION**

After discussing the inception phase, I need to focus on the Elicitation phase. So, this chapter specifies the Elicitation phase.

## **3.1 INTRODUCTION**

Requirements Elicitation is a part of requirements engineering that is the practice of gathering requirements from the users, customers and other stakeholders. I have faced many difficulties, like understanding the problems, making questions for the stakeholders, problems of scope and volatility. Though it is not easy to gather requirements within a very short time, I have surpassed these problems in an organized and systematic manner.

## **3.2 ELICITING REQUIREMENTS**

Unlike the beginning, Elicitation uses a requirements format that incorporates problem solving, preparation, negotiations and specification components, in which questions were answered. A group of end-users and developers must cooperate in order to generate the demands.

**3.3 COLLABORATIVE REQUIREMENTS GATHERING**

There are many different approaches to collaborative requirements gathering. Each approach makes use of a slightly different scenario. We followed the subsequent steps to do it:

I. Meetings were conducted with the research fellows of The University of Manchester. They were questioned about their requirements and expectations from the tool.

II. They were asked about the comparative analysis of Trajectory Interface.

III. At last we selected our final requirement list from these meetings.

### **3.4 QUALITY FUNCTION DEPLOYMENT**

Quality Function Deployment (QFD) is a technique that translates the needs of the customer into technical requirements for software. It concentrates on maximizing customer satisfaction from the software engineering process. So,I have followed this methodology to identify the requirements for the project. The requirements, which are given below, are identified successfully by the QFD.

#### 

#### **3.4.1 NORMAL REQUIREMENTS**

Normal requirements are generally the objectives and goals that are stated for a product or system during meetings with the stakeholders. The presence of these requirements fulfills stakeholders’ satisfaction. The normal requirements of my project-

* Comparison of the different trajectory inference methods
* Gene Expression
* An easy way to access the data without any bioinformatic expertise.

#### **3.4.2 EXPECTED REQUIREMENTS**

The requirements that are implicit to the system might not be brought up during the meeting because of their fundamental nature. Despite being not explicitly mentioned their presence must be ensured. Otherwise, the product will leave customers dissatisfied. These requirements are called expected requirements and these are stated below.

* Sample and cluster overview panels.
* Tables of most expressed genes and marker genes for samples and clusters.
* Tables of enriched pathways for samples and clusters.
* Different Methods for Trajectory Interface

#### **3.4.3 EXCITING REQUIREMENTS**

* Interactive 2D and 3D dimensional reductions.
* All plots can be exported to PNG. In addition, 2D dimensional reductions can be exported to PDF.
* Tables can be downloaded in CSV or Excel format.

### **3.5 Usage Scenario**

Single cell technologies are becoming increasingly important tools in biological analysis. Complementing average measurements on bulk populations of cells, single-cell measurements provide a finer-grained picture of complex biology and unmask heterogeneity that is present in tissues.

Cistron , a trajectory based single cell analysis tool is a standalone application. This tool will take a normalised biological dataset as input. It will show the gene expression, Sample and Cluster Overview panels, Tables of most expressed

# **CHAPTER-04: SCENARIO BASED MODELING**

For developing our software, we are giving the highest priority to user satisfaction. To identify the requirements to establish meaningful analysis and design model we determine how users want to interact with the system. Thus, our requirements modeling begins with scenario generation in the form of use cases, activity diagrams.

4.1 Use Case

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions that some system or sub-systems can perform in collaboration with one or more external users of the system.

The first step in writing a Use Case is to define that set of “actors” that will be involved in the story. Actors are the different people that use the system or product within the context of the function and behavior that is to be described. Actors represent the roles that people play as the system operators. Every user has one or more goals when using the system.

4.1.1 Primary Actor

Primary actors interact directly to achieve required system function and derive the intended benefit from the system. They work directly and frequently with the software. In our system both users the system both are primary actor.

4.2 Activity diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In this chapter we did try to provide each use case and its corresponding activity diagram together.

**4.3 Use Case and Activity Diagram**

**4.3.1 Level 0 Use Case Diagram of Cistron**

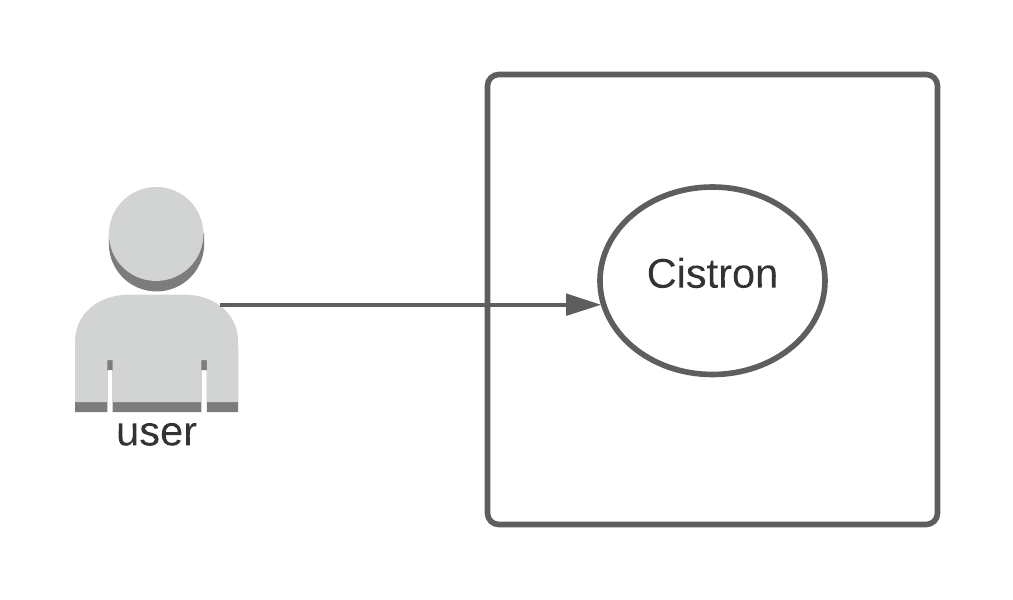


Figure 1: Level 0 use case diagram of Cistron

Table 6: Information about level 0 use case diagram

|  |  |
| --- | --- |
| **Name:** | Cistron |
| **ID:** | L-0 |
| **Primary Actor:** | User |
| **Secondary Actor:** | None |

#### Description of Level 0 Use Case Diagram

After analyzing usage scenario, we found that user interact with our system as primary actor.

### Level 1 Use Case Diagram of Cistron

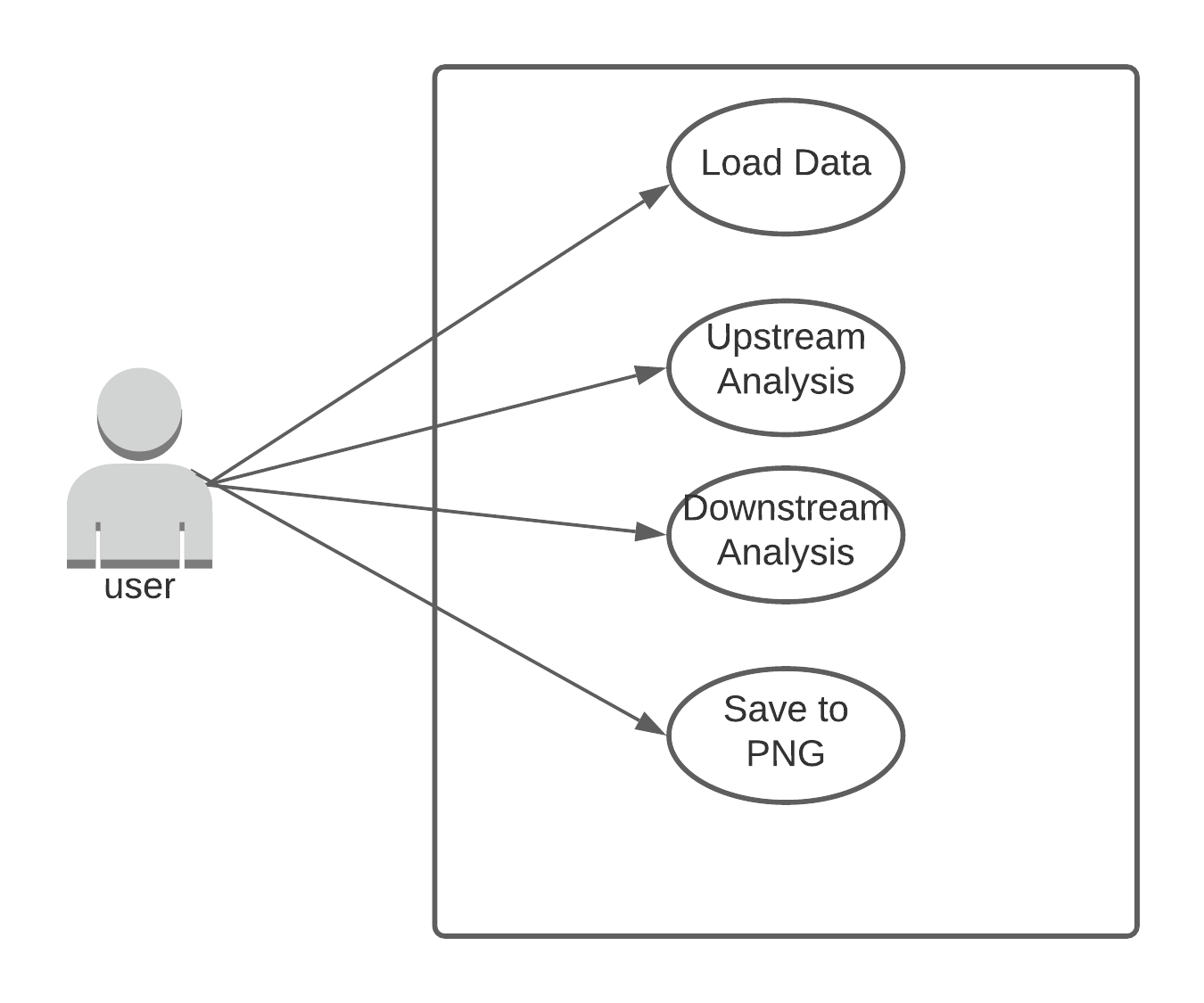


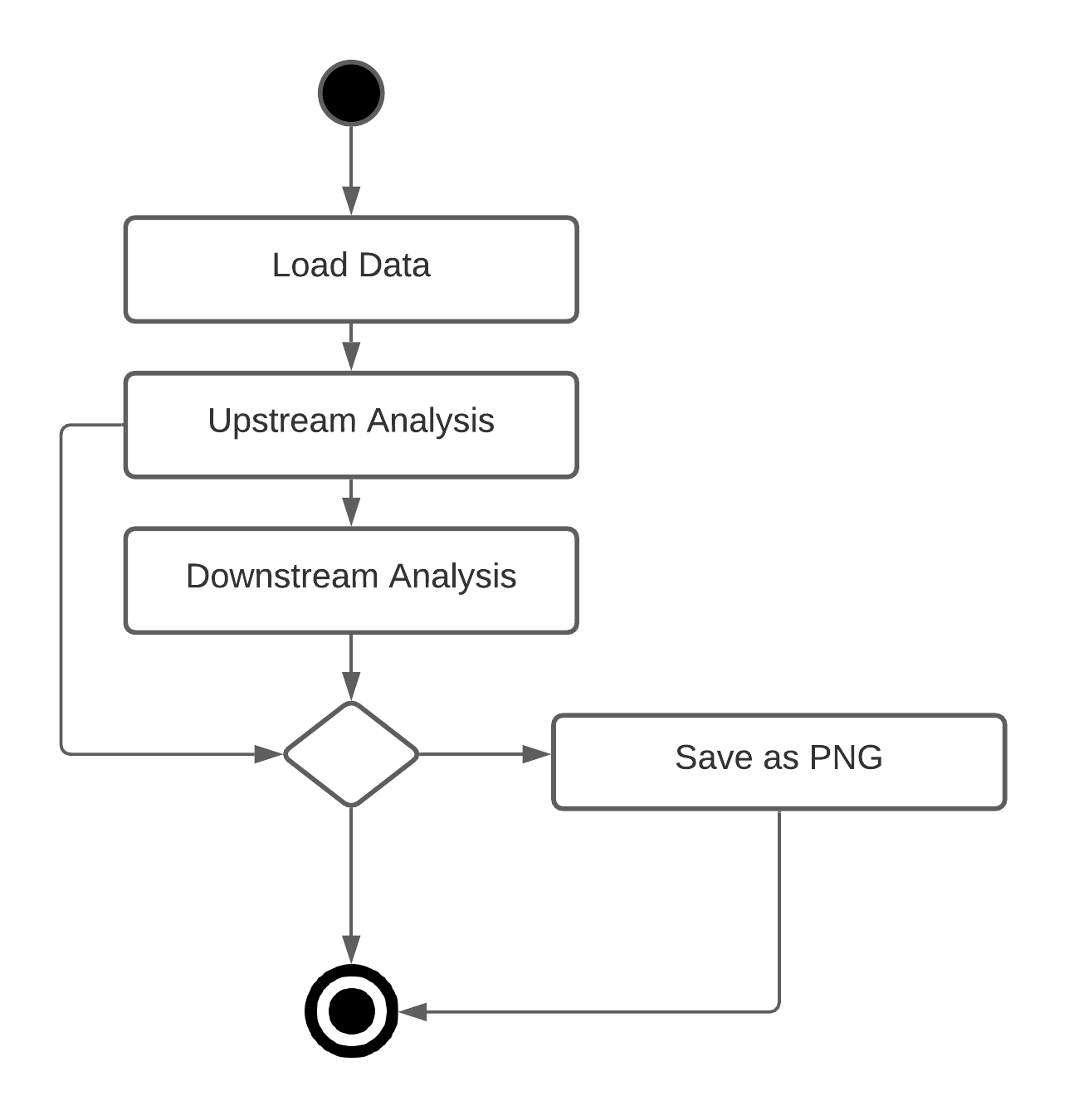
Figure 2: Level 1 use case diagram of Cistron

Table: Information about level 1 use case diagram

|  |  |
| --- | --- |
| **Name:** | Cistron |
| **ID:** | L-1 |
| **Primary Actor:** | User |
| **Secondary Actor:** | None |

#### Description of Level 1 Use Case Diagram

In the usage scenario we separated our System into several modules. Here the user provides scanned image as input and get corresponding Bengali text.



Level 1 Activity Diagram of Cistron

# **CHAPTER-06: CLASS BASED MODELING**

We intended this chapter to describe class-based modeling for our “Cistron”.

In this chapter, our designed class-based model represents the objects that our “Cistron” will manipulate, the operation that will applied to the objects, relationships between and the collaboration that occur between the classes that are defined.

## **5.1 Class Diagram**

In this stage we designed class diagram in the Unified Modeling Language. This is a type of static diagram to describe the structure of our system. Here we also designed two individual design for our two subsystems.

### Class Diagram of Cistron

PreProcessor

convertGrayscale()

reduceNoice()

thresholdImage()

convertBinaryImage()

identifyLine()

segmentCharacterCell()

generateDecimalCode()

inputImage()

generateText()

saveText()

BrailleToextTranslator

ImageTranslator

TextGenerator

convertCharacter()

generateText()

Figure 3: Class diagram of Cistron

# CHAPTER-06: Data Flow Modeling

We intended this chapter to describe data flow modeling for our “Cistron”.

## 6.1Introduction

A data flow diagram is a graphical representation of the flow of data through an information system. We use data flow diagram to diagrammatically represent the flow and exchange of information within our “Cistron”. As previous chapter, we modeled our data flow diagram based on our two main sub system.

## 6.2 Data Flow Diagram

We did try to go initial level to deep level in our system through our data flow diagram.

### Level 0 Data Flow Diagram of Cistron

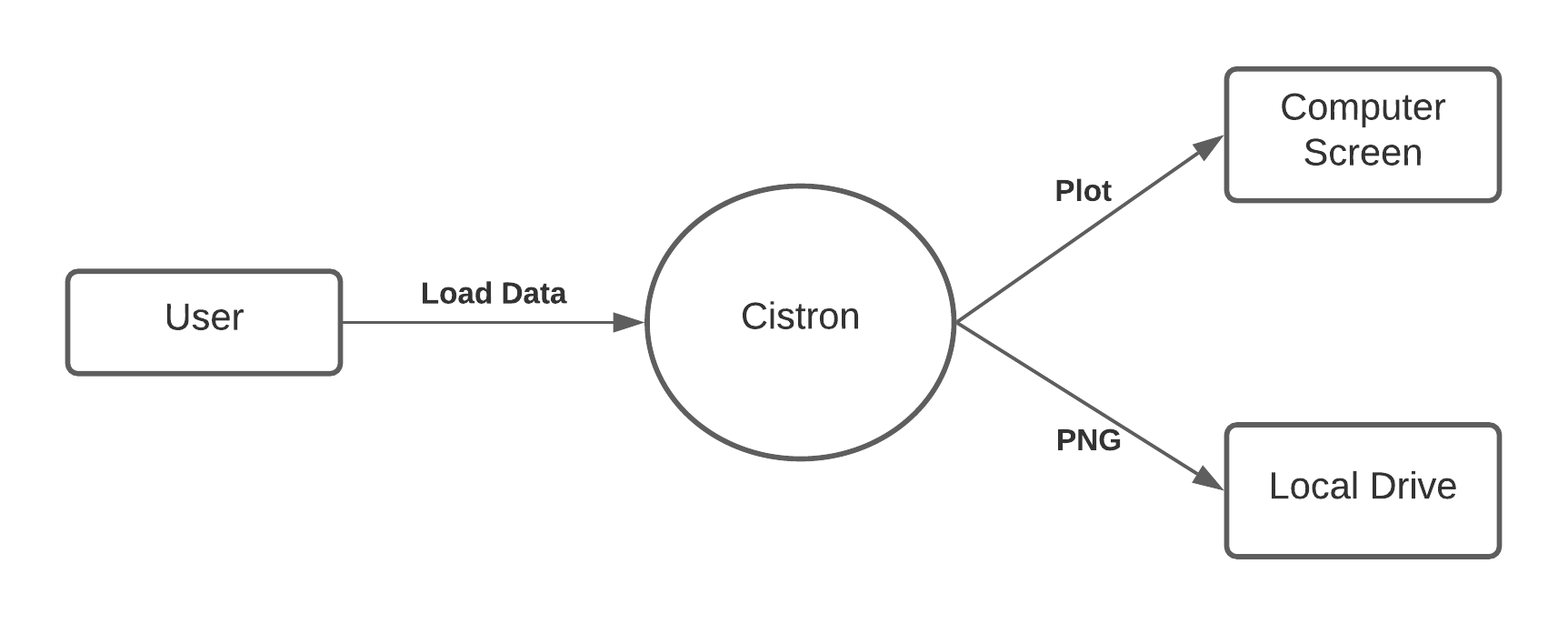


Figure 4: Level 0 data flow diagram of Cistron

### Level 1 Data Flow Diagram of Cistron

Figure 5: Level 1 data flow diagram of Cistron