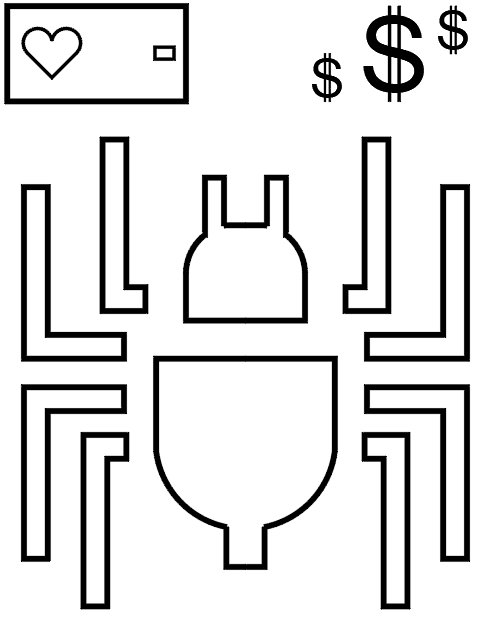
**Loyalty Crawler**

**Architectural Description**

**Version 1.0**



UTD Student Group:

Anthony Spencer (Group Leader)  
Alexander Baselice  
Alexander Lundin  
Jairo Galarza  
Joseph Samonte  
Tony Nhan

RECORD OF CHANGES

\*A - ADDED M - MODIFIED D – DELETED

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| VERSION NUMBER | DATE | NUMBER OF FIGURE, TABLE OR PARAGRAPH | **A\* M D** | TITLE OR BRIEF DESCRIPTION | CHANGE REQUEST NUMBER |
| 1.0 | 03/01/2019 |  |  | Third deliverable |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**TABLE OF CONTENTS**

**Section Page**

[**ABSTRACT**](#_4qtpzccd7j9d) **5**

[**SECTION 1. OVERVIEW**](#_ksk2dqcvgda) **6**

[1.1 Scope](#_lm30927p9bui) 6

[1.2 Purpose](#_mox6dn4z2pb3) 6

[1.3 Intended Users](#_joj6wkqcu9vq) 6

[1.4 Conformance to this recommended Practice](#_dtnlktczq8ln) 7

[1.5 Overview](#_ssfihx9st2pj) 7

[**SECTION 2. REFERENCES**](#_y1g4nvyybxv2) **8**

[**SECTION 3. DEFINITIONS**](#_lv3g1mrwfyyv) **9**

[**SECTION 4. CONCEPTUAL FRAMEWORK**](#_vo5tgogixvy3) **11**

[4.1 ARCHITECTURAL STYLE USED](#_ivtpb5yn579b) 11

[4.2 STAKEHOLDERS AND THEIR ROLES](#_6rw74imscpzy) 11

[4.3 ARCHITECTURAL ACTIVITIES IN THE LIFE CYCLE](#_v40wcqu0vfuv) 12

[4.4 USES OF ARCHITECTURAL DESCRIPTION](#_4r271z2i0sqm) 13

[**SECTION 5. ARCHITECTURAL DESCRIPTION PRACTICES**](#_3tr0iau5f7wm) **14**

[5.1 ARCHITECTURAL DOCUMENTATION](#_a9u0u3bva9lp) 14

[5.2 SELECTION OF ARCHITECTURAL VIEWPOINTS](#_a5d1g867sjhm) 14

[5.3 ARCHITECTURAL MODEL AND VIEWS](#_8fz8y31nw1cu) 18

[5.4 CONSISTENCY AMONG ARCHITECTURAL VIEWS](#_2115pipgdai4) 23

[5.5 ARCHITECTURAL RATIONAL](#_8hwnzeh9y8w6) 24

[5.6 TECHNOLOGY, SOFTWARE AND HARDWARE USED](#_hx36du870osm) 24

**LIST OF FIGURES**

**Figure Page**

Figure 3-1. MVC pattern control flow [1](#_meukdy)0

Figure 5-1. Application-Usage-Viewpoint-1.0 [1](#_meukdy)7

Figure 5-2. Package-Diagram-1.0 [1](#_meukdy)8

Figure 5-3. Architecture-1.0 19

Figure 5-4. Archive-View-1.0 20

Figure 5-5. File-View-1.0 21

Figure 5-6. Session-View-1.0 22

Figure 5-7. Device-Diagram-1.0 23

Figure 6-1. Hardware-View-1.0 26

**LIST OF TABLES**

**Table Page**

[Table 4-1. Architectural Activities 1](#_meukdy)2

Table 5-1. Business Process Viewpoint14

Table 5-2. Product Viewpoint15

Table 5-3. Application Behavior Viewpoint15

Table 5-4. Application Cooperation Viewpoint16

Table 5-5. Application Structure Viewpoint16

Table 5-6. Application Usage Viewpoint17

# ABSTRACT

This document will describe the architecturally significant components the UTD Student Group shall use during the next deliverable. This document will outline the main pillars used for product construction and the finishing touches will be addressed later, during design. The main method for conveying the architecture will be different diagrams and views that represent the foundation for the application.

# SECTION 1. OVERVIEW

## 1.1 Scope

This document outlines the necessary definitions and structures pertaining to the architectural model of the web crawler project. In the subsequent sections, the UTD Student group identifies the relevant stakeholders at hand, the architectural scheme applied to the product, and the architectural activities/workflows supported by this scheme. The UTD student group also provides diagrams of the architectural system with a description of how each piece of the system works together. The MVC pattern control flow the UTD student group is using is discussed. The webcrawler is broken down into multiple parts. The different elements of the GUI are broken down to show how those pieces work together.

## 1.2 Purpose

The purpose of this document is to adequately and articulately define the architectural foundation for the web crawler product. The primary purpose of this document is to satisfy the functioning structure of the loyalty rewards searching capability of the web crawler product. In particularly, the UTD Student Group identifies the architectural scheme applied to the product, provides the rationale for using this scheme for the model, defines a generalized component structure of the model, and illustrates the model through the use of architectural view diagrams. In essence, this document is intended to capture and convey the significant architectural decisions which have been made in designing and building the system. It is a documented method in which the system’s architect and others involved in the project can better define and understand the problems to be solved and how they will be represented in the system.

## 1.3 Intended Users

Alliance Data Sponsors - the sponsor that provide the demand for the project. Alliance wish to create the software that allow them to have easier time finding loyalty rewards programs online. The web crawler will search and collect the data without users having to actively seek the artifacts themselves.

*Sponsors:*

* Jeff Buchmiller
* Felix Tsai

Development and Testing Team - members of the UTD Student Group responsible for the planning, implementation, and testing of the web crawler project. The team is both responsible and a stakeholder in the success of the project.

*Developers and Testers:*

* Anthony Spencer (Team Leader)
* Alex Lundin
* Tony Nhan
* Joseph Samonte
* Jairo Galarza
* Alex Baselice.

## 1.4 Conformance to this recommended Practice

For conformance to this recommended practice, an architecture shall adhere to section 5 of this document. Any deviation from section 5 will break conformance and this architectural document shall no longer apply.

## 1.5 Overview

This specification document is organized as follows:

1. Section 1, Introduction - describes the premise of this document
2. Section 2, References - lists and describes all references in this document
3. Section 3, Definitions - common definitions and terms
4. Section 4, Conceptual Framework - outline of the architecture based on customer roles
5. Section 5, Architectural Description Practices - architectural diagrams, in reference to the framework

# 

# SECTION 2. REFERENCES

List of References

[1] Cabibbo.dia.uniroma3.it. (2019). [online] Available at: http://cabibbo.dia.uniroma3.it/ids/altrui/ieee1471.pdf [Accessed 24 Feb. 2019].

[2] https://code.tutsplus.com/tutorials/mvc-for-noobs--net-10488

Reference Descriptions

[1] This is the PDF standard for IEEE 1471 Software Architecture Documentation

[2] This page explains MCV pattern for noobs

# 

# SECTION 3. DEFINITIONS

1. model-view-controller (MVC)

*This term refers to the architectural pattern that separates three core features from each other. Model to represent data. View for the user to interact with and the controller to facilitate data transfer between the view and the model.*

1. artifacts

*The term artifacts refers to any and all documents retrieved by the web crawler, as well as the documents stored in the crawler cache and artifact file reserve. This includes, but is not limited to, HTML files, image files (.png, .jpeg, etc.), and text files (.txt, .pdf, .docx, etc.).*

1. search session

*The term refers to an instance of activating the web crawler for a duration of time, and the resulting artifacts for which it finds.*

1. crawler cache

*This term refers to the architectural component (GUI & database structures) that shall store artifacts found during a single search session.*

1. artifact file reserve

*This term refers to the architectural component (GUI & database structures) that the user shall use to organize and store artifacts deemed relevant from the crawler cache.*

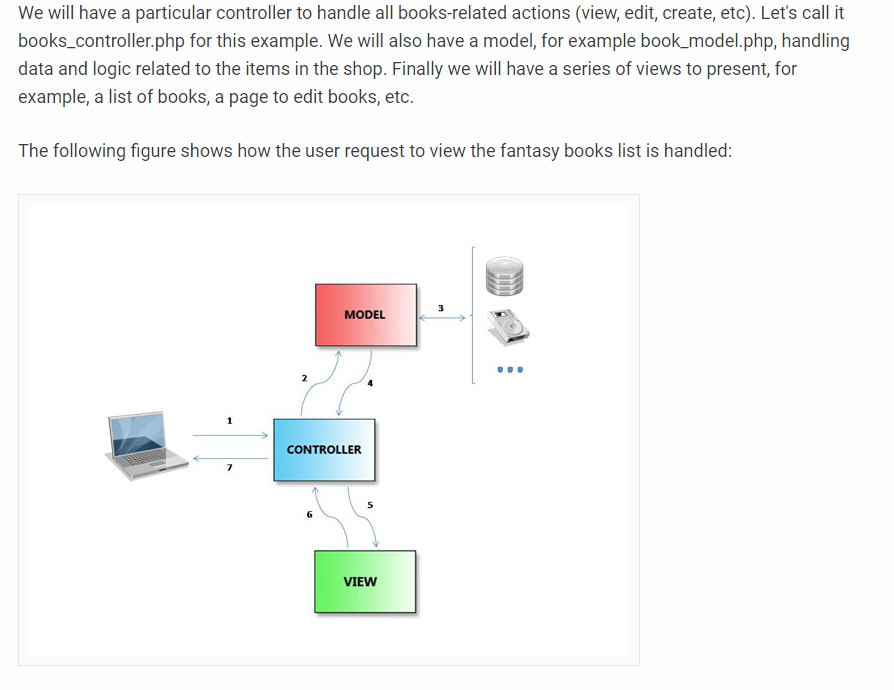
1. validation

*process of checking if specification meets customers needs*

1. verification

*process of checking if software implementation meets specification*

Figure 3-1. MVC pattern control flow [2].



# SECTION 4. CONCEPTUAL FRAMEWORK

## 4.1 ARCHITECTURAL STYLE USED

The UTD Student Group shall use a Model-View-Controller architecture to build the web crawler application. The application is divided into five main feature sets:

1. GUI
2. Web crawler mechanism
3. Main controller
4. AWS database
5. Local storage

The MVC architecture will support the previous four features by separating the application into three modular components. The model will represent the data retrieved from the AWS Database. The View is the GUI that the user will interact with to initiate the crawler and see the returned results. The controller facilitates data transfer between the model and the view.

## 4.2 STAKEHOLDERS AND THEIR ROLES

*Stakeholders*

Sponsors (Jeff and Felix). These are the clients who are responsible for starting the project. They are the acquirer in this case. They will also be the users when the project is over. They will be the ones using the software to look for specific links that helps them do their job more efficiently. They also supervise the project by giving feedback on the deliverables. They are also acting as the user, owner, and customer. At the end of the project, the sponsors want to get a product that will help them do their jobs better.

*Team members*

Each team member contributes to the role of the architect. The team member uses the requirements provided to draft and discuss a possible architecture. This architecture will change based on updates from the sponsor. At the end of the project, each of the team members wants gain experience with devising an architecture for a software project. The team member also wants to learn what to do and not to do when devising an architecture for a software project in the future.

## 4.3 ARCHITECTURAL ACTIVITIES IN THE LIFE CYCLE

The UTD Student group will incorporate architectural activities into the life cycle at various points in the project. The first, and most obvious is this third deliverable, the architectural document. After the initial submission, the group shall do informal reviews during detail design to ensure that it is building a satisfactory system, in accordance with the planned architecture described in this document.

Scenario 4.3.1 from IEEE 1471-2000 describes an architecture for new system construction, where the user and the acquirer are identical, as is the case for our project [1].

1. the architectural documentation can be used throughout life cycle to predict fitness of the system
2. The architectural documentation will evolve with the system
3. The architectural documentation will serve as a reference point for all future changes

Table 4-1. Architectural Activities.

|  |  |  |  |
| --- | --- | --- | --- |
| Stages | Virtual Meetings | In person Meetings | Individual Work |
| Stage 1 | Architectural Pattern Discussion and Review | Whiteboard Client Server Architecture Prototype | Review different architectural patterns to increase discussion quality |
| Stage 2 | Client Server Prototype Review |  | Draw diagrams based on Client Server Architecture |
| Stage 2.1 |  |  | Group consensus to switch to MVC pattern |

## 4.4 USES OF ARCHITECTURAL DESCRIPTION

We have found several uses for the architectural description so far:

* 1. Analysis of alternative architectures

See Table 4-1

* 1. Communications between aquirers and developers

Validation that we are building the right thing for the customer

* 1. Development and maintenance documentation

Verification that we are building the thing right based on this description

* 1. Review, analysis and evaluation of the system across the life cycle

Basis for architectural reviews during design and implementation

# SECTION 5. ARCHITECTURAL DESCRIPTION PRACTICES

## 5.1 ARCHITECTURAL DOCUMENTATION

The scope this documentation is to show how the database, the web crawler system, and the local system (the front-end) all fit together. This documentation also provides all the viewpoints that the UTD Student Group considered for the architecture. For the database, the group is using Amazon RDS Database Management System (Free Tier). The web crawler system is written in Python with Scrapy. This document will show how the front-end has three different parts. The frontend has a session a session view, a file view, and an archive view.

## 5.2 SELECTION OF ARCHITECTURAL VIEWPOINTS

Viewpoints are a means to focus on particular aspects of the architecture. These aspects are determined by the concerns of a stakeholder with whom communication takes place. What should and should not be visible from a specific viewpoint is therefore entirely dependent on the argumentation with respect to a stakeholder concerns.

**5.2.1 Business Process Viewpoint**

The Business Process viewpoint is used to show the high-level structure and composition of one or more business processes. This viewpoint contains other directly related concepts, such as the assignment of business processes to roles and the information used by the business process.

Table 5-1. Business Process Viewpoint.

|  |  |
| --- | --- |
| Business Process Viewpoint | |
| Stakeholders | Process architect |
| Concerns | Structure of business processes, consistency and completeness, responsibilities |
| Purpose | Designing |
| Abstraction Level | Detail |
| Layer | Business layer (to be decided) |
| Aspects | Behavior |

**5.2.2 Product Viewpoint**

The Product viewpoint depicts the value the product offers to the customers or other external parties involved and shows the composition of one or more products in terms of the constituting (business or application) services, and the associated contract(s) or other agreements. A Product viewpoint is typically used in product development to design a product by composing existing services or by identifying which new services have to be created for this product, given the value a customer expects from it.

Table 5-2. Product Viewpoint.

|  |  |
| --- | --- |
| Product Viewpoint | |
| Stakeholders | Product developers |
| Concerns | Product development, value offered by the products of the enterprise. |
| Purpose | Designing and deciding |
| Abstraction Level | Coherence |
| Layer | Business layer, application layer |
| Aspects | Behavior |

**5.2.3 Application Behavior Viewpoint**

The Application Behavior viewpoint describes the internal main behavior of an application. This viewpoint also helps in identifying functional overlap between different applications.

Table 5-3. Application Behavior Viewpoint.

|  |  |
| --- | --- |
| Application Behavior Viewpoint | |
| Stakeholders | Application developers |
| Concerns | Structure, relations, consistency, and dependencies between modules of application. |
| Purpose | Designing and reduction of complexity |
| Abstraction Level | Coherence, details |
| Layer | Application layer (to be determined) |
| Aspects | Information, behavior, structure |

**5.2.4 Application Cooperation Viewpoint**

The Application Co-operation viewpoint describes the relations between applications components in terms of the information flows between them, or in terms of the services they offer and use. This viewpoint is typically used to create an overview of the application landscape of an organization.

Table 5-4. Application Cooperation Viewpoint.

|  |  |
| --- | --- |
| Application Cooperation Viewpoint | |
| Stakeholders | Process, application and product developers |
| Concerns | Relations and dependencies between applications, structure of information flow between services, consistency and completeness |
| Purpose | Designing |
| Abstraction Level | Coherence, details |
| Layer | Application layer (to be determined) |
| Aspects | Information, behavior, structure |

**5.2.5 Application Structure Viewpoint**

The Application Structure viewpoint shows the structure of one or more applications or components. This viewpoint is useful in designing or understanding the main structure of applications or components and the associated data.

Table 5-5. Application Structure Viewpoint.

|  |  |
| --- | --- |
| Application Structure Viewpoint | |
| Stakeholders | Application and product developers |
| Concerns | Application structure, consistency and completeness, reduction of complexity |
| Purpose | Designing |
| Abstraction Level | Details |
| Layer | Application layer (to be determined) |
| Aspects | Information, structure |

**5.2.6 Application Usage Viewpoint**

The Application Usage viewpoint describes how applications are used to support one or more business processes, and how they are used by other applications.

Table 5-6. Application Usage Viewpoint.

|  |  |
| --- | --- |
| Application Usage Viewpoint | |
| Stakeholders | Application and product developers, Jeff and Felix |
| Concerns | Reduction of complexity, consistency and completeness |
| Purpose | Designing, deciding |
| Abstraction Level | Coherence |
| Layer | Business and application layers (to be determined) |
| Aspects | Behavior, structure |

Figure 5-1. Application Usage Viewpoint

## 

Diagram above is subject to change (More detail will be added in regards to application interaction)

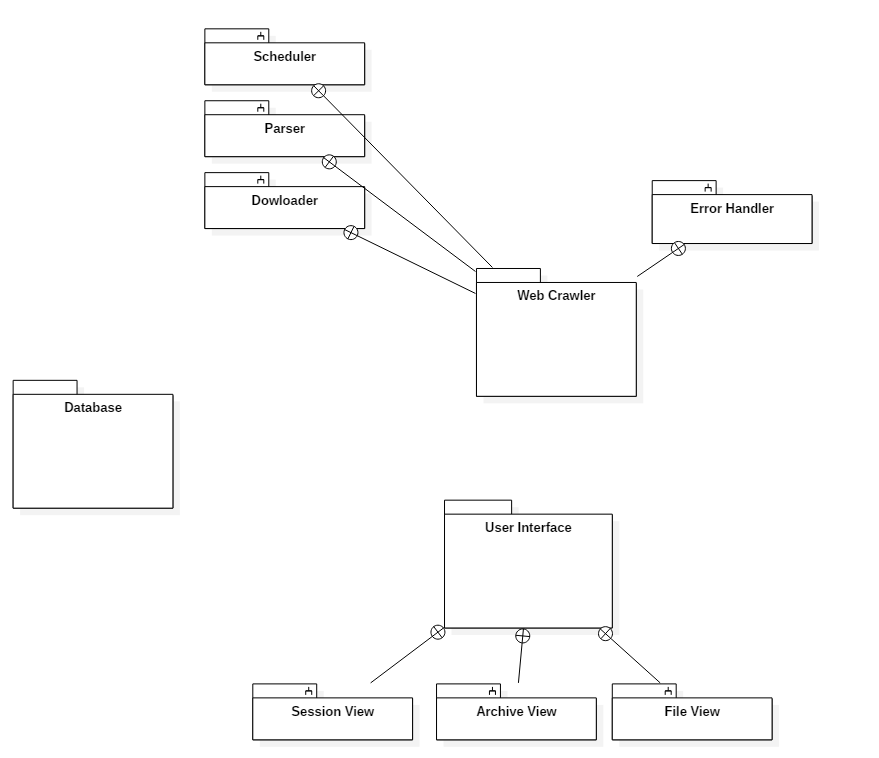
## 

## 5.3 ARCHITECTURAL MODEL AND VIEWS

**5.3.1 Package Diagram**

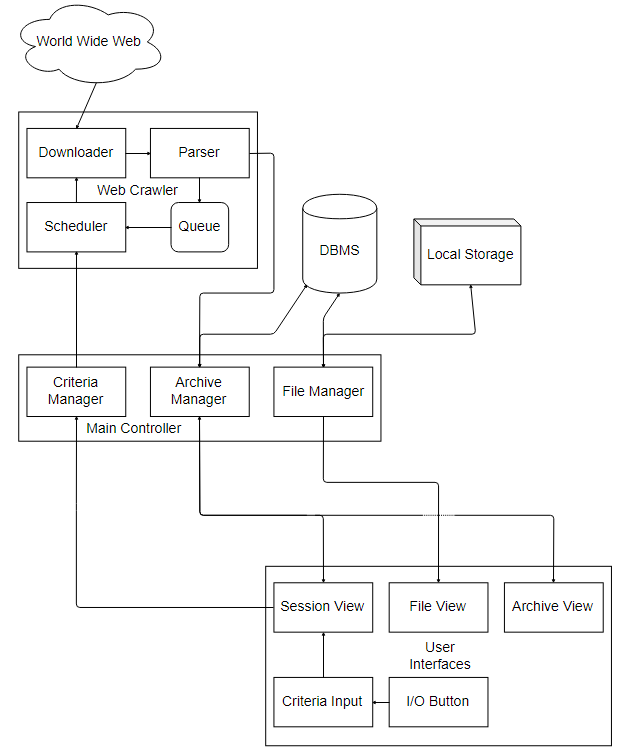
This diagram depicts the core features in terms of packages which are the larger folders and subsystems which are the half size folders with wrenches on the top left tab. Our third deliverable requires that we “use packages stereotyped as subsystems <<subsystem>>” the wrenches are our subsystems.

Figure 5-2. Package-Diagram-1.0



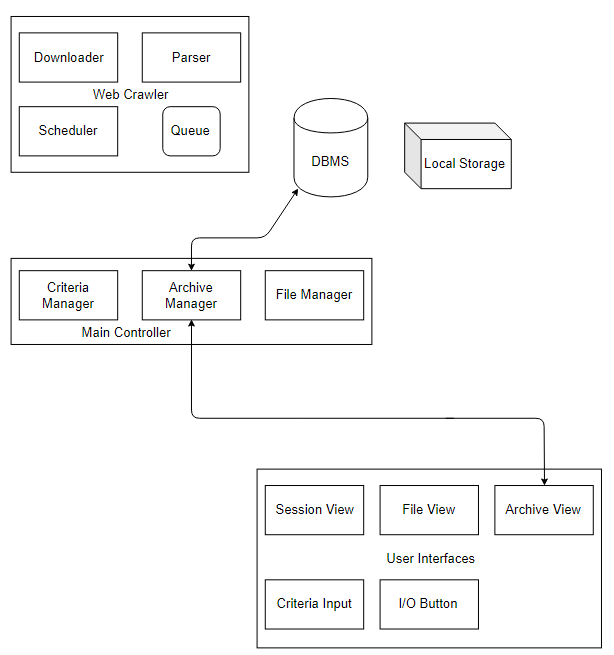
**5.3.2 Architecture Model**

Figure 5-3. Architecture-1.0



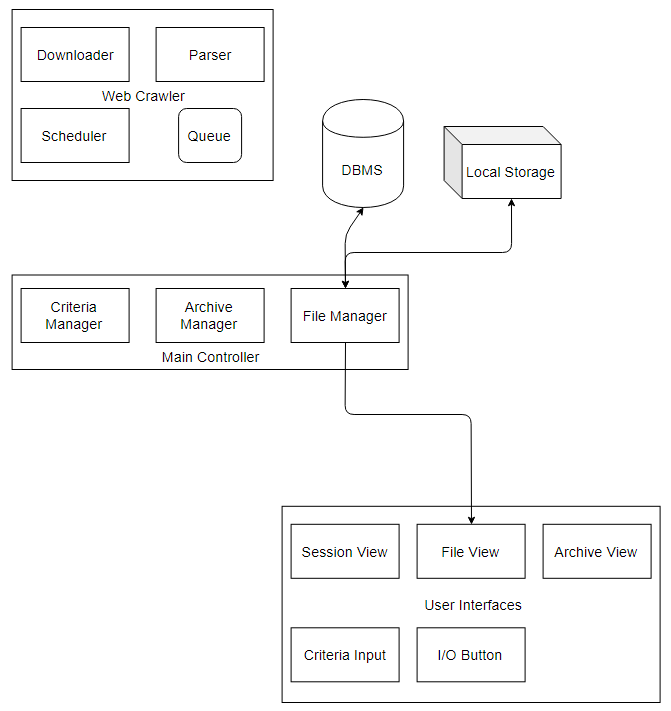
**5.3.3 Archive Retrieval View**

Figure 5-4. Archive-View-1.0



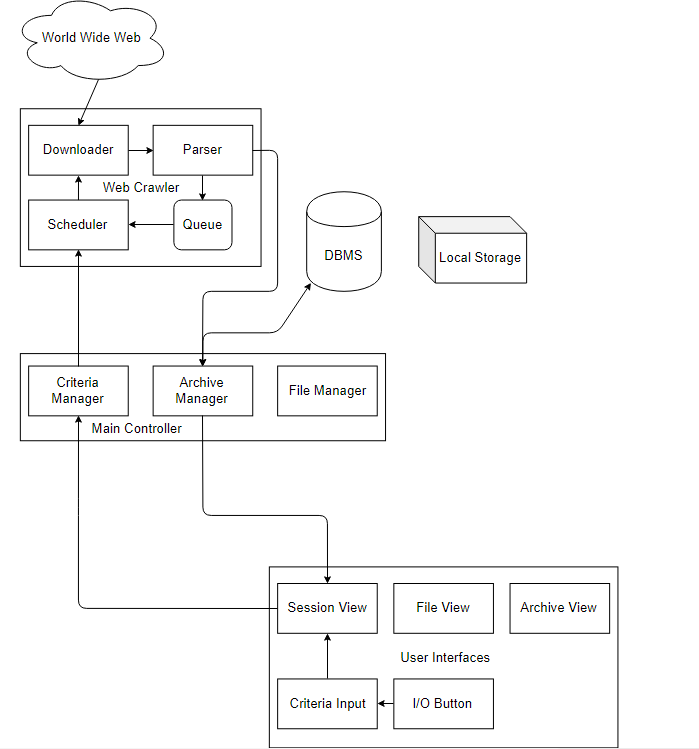
**5.3.4 File Reserve View**

Figure 5-5. File-View-1.0



**5.3.5 Session View**

Figure 5-6. Session-View-1.0



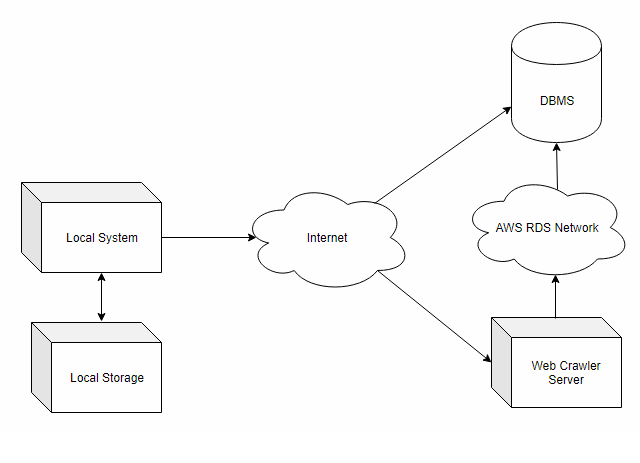
## 

## 

## 

**5.3.6 Device Diagram**

Figure 5-7. Device-Diagram-1.0



## 

## 5.4 CONSISTENCY AMONG ARCHITECTURAL VIEWS

All the architectural views contain the same 5 elements. These common elements are GUI, Web crawler mechanism, Main controller, AWS database and Local storage. The way the elements are named are slightly different in each iteration. the main controller is called the “Web Crawler server” in section 5.7. In section 5.4 the GUI is also known as the user interface.

## 5.5 ARCHITECTURAL RATIONAL

For this product, the UTD student group identified two common architectures to consider: client-server and model-view-controller. Initially, the group chose the client-server architecture as the basis for our model, following the pattern commonly used in vast number of websites today. However, as we continued identifying our relevant entities and the relationships between them, the group realized the number of true “services” incorporated into our model were perhaps too few to make a client-server architecture necessary. Thus, upon reaching this realization, the group decided it would be more practical to model our product after the model-view-controller architecture. As seen in our architectural views in section 5.3 of this document, we designed a master controller, itself with its own constituents, to facilitate these few “service” workflows incorporated into our model; the master controller may be thought of as a “domestic” server to our client entities/classes. Furthermore, our model demonstrates that a user will be exposed to the model through their respective view (i.e. the user interface). When the user engages our product, whether to conduct a search session, retrieve an archived session, or save session artifacts to local storage, the controller will carry out these actions, thus changing and/or affecting the model. When there is indeed a resulting change to the model from one of these actions, the controller will instigate the respective view to reflect the updated model.

## 5.6 TECHNOLOGY, SOFTWARE AND HARDWARE USED

**Technology**

Python high-level programming languages - Python will be use to implement the main controller for the overall system and also the web crawling mechanism. Python was chosen due to it existing web scraping framework, ease of use, and ability to run on all prominent operating system like Windows, MacOS, Linux.

Windows Directories - the local storage will use windows folder and file system to manage file.

Amazon Relational Database Service (RDS) Free Tier - free database system from Amazon Web Services. Come with:

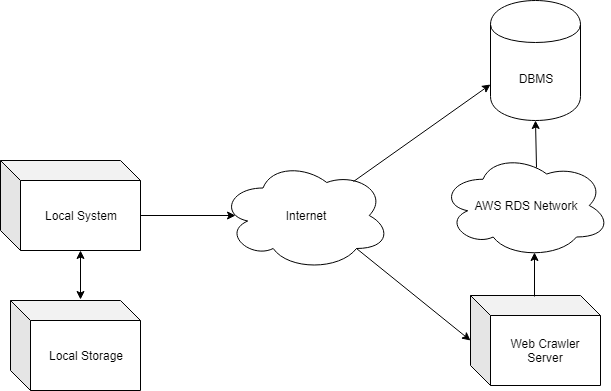
* 750 hours of Amazon RDS Single-AZ db.t2.micro Instance usage running MySQL.
* 20 GB of General Purpose (SSD) DB Storage
* 20 GB of backup storage for your automated database backups and any user-initiated DB Snapshots

**Software**

* Python - programming language to implement main controller.
* Python | Scrapy - open source and collaborative web scraping framework. The framework will handle the web page download and scraping.
* MySQL open source relational database with Amazon RDS - MySQL will be the database management software that will be use for managing the stored artifacts.
* Any web browser (Chrome, Firefox, Explorer, Safari) - web browser is required for accessing the web application.
* Text Editors to handle front end graphical user interface.

**Hardware**

Figure 6-1. Hardware-View-1.0

****

* Local System - User’s device with existing operating system and web browser to access the web application ‘Loyalty Crawler’
* Local Storage - User’s hard disk that can be use store artifact locally so user can access them whenever.
* Web Crawler Server - Server that provide the web based application ‘Loyalty Crawler’
* Amazon RDS Database Management System - Amazon database provided through Amazon RDS Free Tier service. Limited Storage and Access

**APPENDICES**

# 

**DOCUMENT CHANGE REQUEST (DCR)**

|  |  |
| --- | --- |
| Document Title: **Software Requirements Specification** | Tracking Number:  DCR-AD- |
| Name of Submitting Organization:  UTD Student Group | |
| Organization Contact:  amlundin88@gmail.com | Phone: |
| Mailing Address: | |
| DCR Description: | Date: |
| Change Location:  (use section #, figure #, table #, etc.) | |
| Proposed change: | |
| Rationale for Change: | |
| Note: For the ***appropriate authority*** to take appropriate action on a change request, please provide a clear description of the recommended change along with supporting rationale.  Email to:  anthonygaganovspencer@gmail.com  alexbaselice2@gmail.com  amlundin88@gmail.com  jgalarza303@gmail.com  josephisnt@gmail.com  xclearzx@gmail.com  Submit online:  ***Print this sheet and store in this Google Drive folder***  *SE Senior Project/Deliverables/Document Change Request Folder*  DCR Form 1/2009 | |