Savvy

**Software Architecture Document**

**Version 1.0**

**02/11/2020**

**By**

**Pragati Gupta,**

**Tejaswi Kukalakunta,**

**Vishali Gummadi**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Description of Versions / Changes** | **Responsible Party** | **Date** |
| 1.0 | Initial version | Pragati Gupta | 10/17/2020 |
|  | Added Entity Data Model diagram | Tejaswi | 10/20/2020 |
|  | Wireframes | Vishali | 10/20/2020 |
|  | Prototype | Pragati | 10/25/2020 |
|  | Coding | Pragati , Tejaswi ,Vishali | 10/26/2020 |
|  | Testing | Pragati , Tejaswi ,Vishali | 11/02/2020 |

Table of Contents

[1. Introduction 1](#_Toc353470033)

[1.1. Purpose 1](#_Toc353470034)

[1.2. Scope 1](#_Toc353470035)

[1.3. Definitions, Acronyms, and Abbreviations 2](#_Toc353470036)

[1.4. References 2](#_Toc353470037)

[1.5. Overview 2](#_Toc353470038)

[2. Architectural Representation 3](#_Toc353470039)

[3. Architectural Goals and Constraints 4](#_Toc353470040)

[4. Use-Case View 4](#_Toc353470041)

[4.1. Actors 4](#_Toc353470042)

[4.2. Use-Case Realizations 5](#_Toc353470043)

[4.2.1. Login 5](#_Toc353470044)

[4.2.2. Request Analysis (Get Report) 7](#_Toc353470045)

[4.2.3. Retrieve Last Report 9](#_Toc353470046)

[4.2.4. Print Report 10](#_Toc353470047)

[4.2.5. Email Report 11](#_Toc353470048)

[5. Logical View 12](#_Toc353470049)

[5.1. Overview 12](#_Toc353470050)

[5.2. Interface Definitions 14](#_Toc353470051)

[6. Data View 23](#_Toc353470052)

[7. Deployment View 25](#_Toc353470053)

Software Architecture Document

# Introduction

This document provides a high level overview and explains the architecture of the Savvy.

The document defines goals of the architecture, the use cases supported by the system, architectural styles and components that have been selected. The document provides a rationale for the architecture and design decisions made from the conceptual idea to its implementation.

## Purpose



Figure: 1.1 Use Case View

The Software Architecture Document (SAD) provides a comprehensive architectural overview of the Savvy.

## Scope

The scope of this SAD is to explain the architecture of the Savvy.

This document describes the various aspects of the system design that are considered to be architecturally significant. These elements and behaviors are fundamental for guiding the construction of the system and for understanding this project as a whole. Who requires a technical understanding of the Savvy system are encouraged to start by reading the Project Proposal, Concept of Operations and Software Requirements Specification documents developed for this system.

## Definitions, Acronyms, and Abbreviations

* **Apache** – Web Server
* **CSS** – Cascading Style Sheets
* **HTTP** – Hypertext Transfer Protocol
* **SQL**– Structured Query Language
* **WWW** – World Wide Web
* **SAD** – Software Architecture Document
* **UML** – Unified Modeling Language
* **User –** Registered Professor
* **XAMMP** – Cross –platform to build WordPress site offline
* **PHP** – Personal Home Page
* **VSC**– Visual Studio Code (Text Editor)

# Architectural Representation

**Use Case view**

**Audience**: Professors of an institute.

**Area**: Processor can create modules and divide the syllabus in to chapter (modules) for the students and upload, pdf, video, ppt, world documents in to the portal. Central functionality of the system is to create questions online (Type of questions can be Fill\_in\_the\_blanks, Match the followings, Matching questions, True and false, Short answers questions, Essay questions) as well as they can upload a text format question and answers, application will detect the questions and answer and create the paper for the students. Professor will get the grades after exam is done.

**Related Artifacts**: Use- Manual

.

# Architectural Goals and Constraints

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

1. The system is meant as a proof of concept for a more complete project prediction system to be built in the future. As a result, one goal of this document is to be useful to future architects and designers.
2. The system will be written using Visual Studio Code (Text Editor). MySQL for database and will be deployed to Apache webserver. These special deployment requirements require additional consideration in the development of the architecture.
3. The Software Requirements Specification outlines a number of anticipated changes that the application could face over time. One of the primary goals of the system architecture is to minimize the impact of these changes by minimizing the amount of code that would need to be modified to implement them. The architecture seeks to do this through the use of modularization and information hiding to isolate components that are likely to change from the rest of the system.

# Use-Case View

The purpose of the use-case view is to give additional context surrounding the usage of the system and the interactions between its components.

## Actors

|  |  |
| --- | --- |
| **User** | |
|  | The user will drive all operation of the software. No distinction is made in regards to type of user. The user interacts with all available interfaces to initiate and monitor all application operations. |
| **Web Portal** | |
|  | The web portal is the main user interface for the system. |

## Use-Case Realizations

### Login

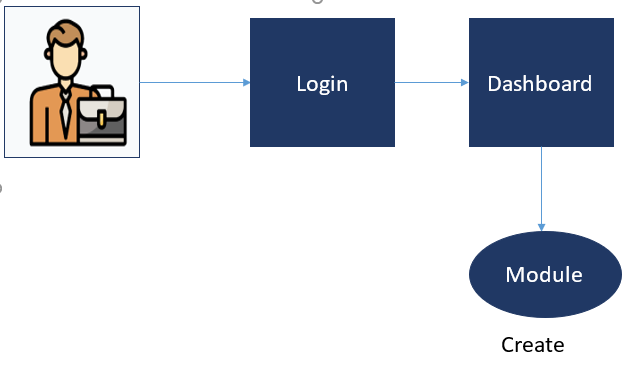
User credentials are authenticated and user is redirected to application home page.



**Figure 4.1** Login Use Case Diagram

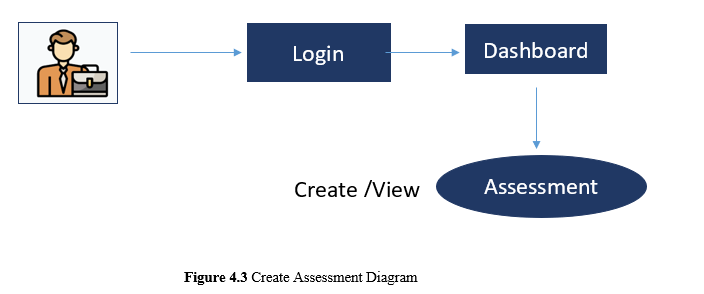
### Module Creation

User requests a report for a user-specified project and report is displayed.



**Figure 4.2** Module Creation Diagram

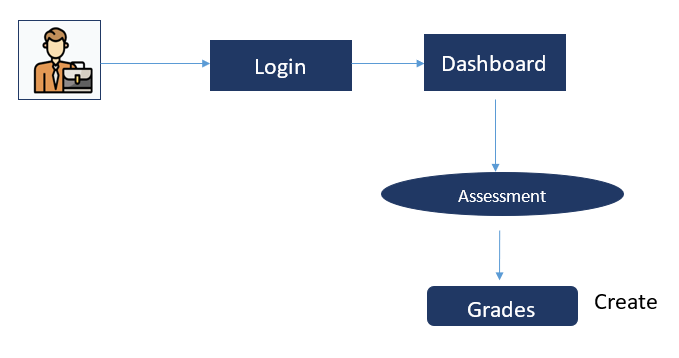
### Create Assessment



**Figure 4.3** Create Assessment Diagram

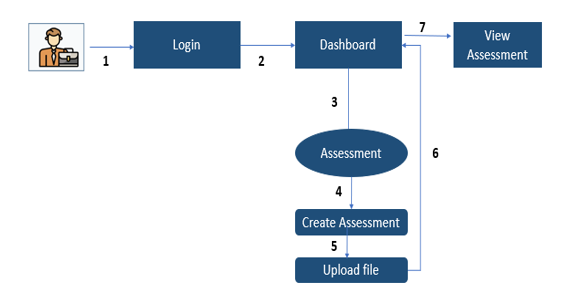
### Grade Report

User requests to get grade report.



**Figure 4.4** Grade Report Diagram

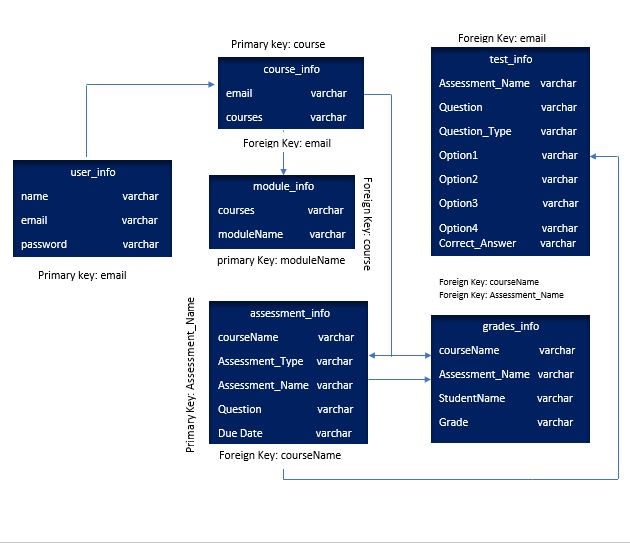
### Automatic Test Creation



**Figure 4.5** Automatic Test Creation Diagram

# Data View

This diagram illustrates the database structure and relationships of the main entities that will be stored by the application in its database. Each element nominally represents a database table. Relationship cardinality is denoted with UML multiplicity notation.



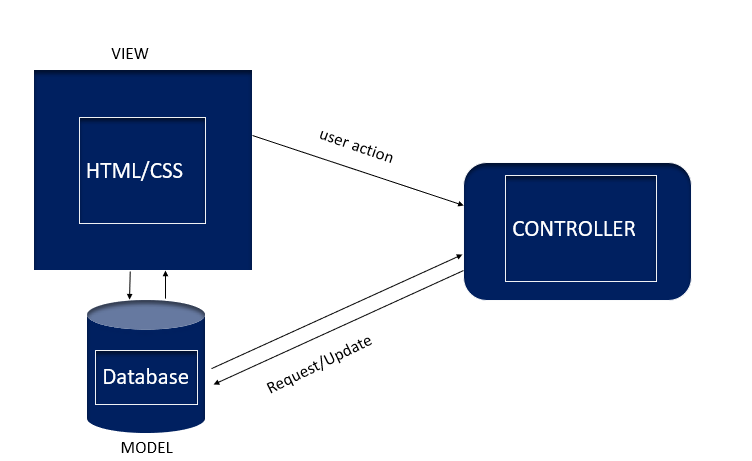
**Figure 5.1** Data Flow Diagram

# Deployment View

The web application will be hosted on a single server. An Apache webserver running a module will be used to serve the application pages. In addition, a MySql use as database.

.

The application’s deployment specifics can be seen below.



**Figure 6.1** Deployment View Diagram

Thank You