

**REV-SYS**  
**AUTOMATION OF REVIEW MANAGEMENT**  
A PROJECT REPORT

*Submitted by*

**SRIDEVI M (312216205101)**

**RAJASRI R (312216205076)**

*in partial fulfilment for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

*in*

**INFORMATION TECHNOLOGY**

**SRI SIVASUBRAMANIYA NADAR COLLEGE OF ENGINEERING**

**KALAVAKKAM**

**ANNA UNIVERSITY: CHENNAI 600 025**

**APRIL 2020**

**ANNA UNIVERSITY: CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report “**REV-SYS**” is the bonafide work of **SRIDEVI M(312216205101)** , **RAJASRI R(312216205076)** who carried out the project work under my supervision.

**SIGNATURE**

Dr. T. NAGARAJAN

**HEAD OF THE DEPARTMENT**

INFORMATION TECHNOLOGY

SRI SIVASUBRAMANIYA NADAR  
COLLEGE OF ENGINEERING

Old Mahabalipuram Road  
Kalavakkam – 603 110

**SIGNATURE**

Dr. T. SHANMUGAPRIYA

**SUPERVISOR**

ASSOCIATE PROFESSOR

INFORMATION TECHNOLOGY

SSN COLLEGE OF ENGINEERING

Old Mahabalipuram Road  
Kalavakkam – 603 110  
Tamil Nadu, India

**Submitted for Project Viva Voce held on .....**

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

We are honored to acknowledge the below mentioned people who have been instrumental in the completion of this project. We express our sincere thanks to **Padma Bhushan Dr. Shiv Nadar**, Chairman, SSN Institutions for being a source of inspiration and for instilling a great mission and vision in our minds. We also thank our principal, **Dr. S. Salivahanan** for providing us with all the support we needed.

We sincerely thank the Head of the Department of Information Technology, **Dr. T. Nagarajan** for his valuable guidance. We also thank our project coordinator, **Dr. R. Srinivasan** for providing us with suitable time slots to work with.

We express our deep gratitude to our guide, **Dr.T.Shanmugapriya**, Assistant Professor, Department of Information Technology for her continuous support and guidance and our panel members, Department of Information Technology for their priceless guidance and suggestions.

## ABSTRACT

Rev Sys is a new initiative that aims at effective use of current technology to enhance the project review process. Currently, there is no soft repository available for checking the previous projects done in the same domain, and there is no support system available for recording the review process. Our project has four views namely-admin, panel members, guide and students. The project has been implemented considering the requirements of all the stakeholders.

The key functionality of **REV-SYS** includes the following : Student view includes access to a repository of previous projects done, viewing attendance and comments posted by panel members, uploading abstract and report. The panel member view allows members to add and edit review comments, view the list of project batches in the panel, view abstract approved by the guide. The admin can view the status of all the students, send notification and add panel members. The guide view has features to add attendance, approve abstracts and reports, add and view comments added by the panel members.

# TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	<b>ABSTRACT</b>	<b>iv</b>
	<b>LIST OF FIGURES</b>	<b>vii</b>
	<b>LIST OF ABBREVIATIONS</b>	<b>ix</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Project background	
	1.2 Objective	
	1.3 Project explanation	
	1.4 Applications	
	1.5 Project overview	
<b>2</b>	<b>TOOLS AND ENVIRONMENT</b>	<b>4</b>
	2.1 Software requirement	<b>4</b>
	2.1.1 Development environment	
	2.1.1.1 Software	
	2.1.1.2 Programming languages	
	2.1.1.3 Operating system	
	2.1.1.4 Framework used	

	2.1.2 Development tools	
	2.2 Hardware requirements	9
<b>3</b>	<b>SYSTEM DESIGN</b>	<b>10</b>
	3.1 Working	
	3.2 Database structure	
<b>4</b>	<b>IMPLEMENTATION</b>	<b>16</b>
	4.1 Proposed work	
	4.2 Functionalities	
	4.3 Source code	
	4.3.1 Model	
	4.3.2 View	
	4.3.3 Controller	
<b>5</b>	<b>SCREENSHOTS AND RESULTS</b>	<b>22</b>
	5.1 Panel view	
	5.2 Student view	
	5.3 Guide view	
	5.4 Admin view	
<b>6</b>	<b>CONCLUSIONS</b>	<b>32</b>
	<b>REFERNCES</b>	<b>33</b>

## LIST OF FIGURES

2.1 Programing languages used.....	5
3.1 Usecase diagram for Rev-Sys.....	10
4.1 Rails MVC Architecture.....	19
<b>5.Screenshots and results</b>	
<b>PANEL VIEW</b>	
5.1 Signin page.....	22
5.2 Pannels allocated to panel members.....	23
5.3 List of students belonging to a particular panel.....	23
5.4 Panel members updating marks and comments.....	24
5.5 Editing marks and comments.....	24
5.6 View abstract uploaded by students.....	25
<b>STUDENT VIEW</b>	
5.7 Students entering their project details.....	25
5.8 Students view their project information.....	26
5.9 View of marks and comments updated.....	26
5.10 Uploading abstract and report.....	27
5.11 Status check of the project details.....	27
<b>GUIDE VIEW</b>	
5.12 Guide of requested students list.....	28
5.13 List of students guided by them.....	28
5.14 Guide can view their students mark.....	29
5.15 View abstract and report uploaded.....	29
5.16 Accept or reject the abstract.....	30

## **ADMIN VIEW**

5.17 Add new staff.....	30
5.18 Overview of the review mark summary.....	31
5.19 Overall progress of report and abstract.....	31
5.20 Finalize the abstract and report.....	32



## **LIST OF ABBREVIATIONS**

<b>CSS</b>	- Cascading Style Sheets
<b>HTML</b>	- Hypertext Markup Language
<b>JS</b>	- Java Script
<b>MVC</b>	- Model-View-Controller
<b>OS</b>	- Operating system
<b>RAM</b>	- Random Access Memory
<b>RVM</b>	- Ruby Version Manager
<b>SQL</b>	- Structure Query Language

# **1.INTRODUCTION**

## **1.1 PROJECT OVERVIEW**

The design and implementation of a comprehensive student information and user interface is to replace the current paper records. College Staff are able to directly access all aspects of a student's project progress through a secure, online interface embedded in the college website. The system utilizes user authentication, displaying only information necessary for an individual's duty. Additionally, each sub-system has authentication allowing authorized users to create or update information in that subsystem. All data is stored securely on SQL servers managed by the college administrator and ensures highest possible level of security.

Previously, the college relied heavily on paper records for this initiative. While paper records are a traditional way of managing project data and different work, there are several drawbacks to this method. First, to convey information to the students it should be displayed on the notice board and the student has to visit the notice board to check that information. It takes a very long time to convey the information to the student. Paper records are difficult to manage and track. The

physical exertion required to retrieve, alter, and re-file the paper records are all non-value added activities. This system provides a simple interface for the maintenance ,management and reviewing of students project reports. It can be used by educational institutes or colleges to maintain the records of students project easily. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant and collecting previously done project information may be very time consuming. All these problems are solved using REV SYS thus reducing paper work and automating the review management process in an educational institute.

## **1.2. PROJECT BACKGROUND**

The main aim of this project is to bridge the communication gap among the students , panel members and the guide. Our application is built on **RUBY ON RAILS** which provides MVC structure. For database purpose commercial **Mysql** is used. Our application is designed with popular technologies like angular js, Bootstrap.

## **1.3. OBJECTIVE**

The main objective of this project is to implement a repository for managing project reports and to share the review comments updated by the panel members in an instant to admin, guide and students. To store

and display report statements of attendance and marks instantaneously to all the participants and decrease the space between student and college faculties.

#### **1.4. PROJECT EXPECTATION**

- Providing the online interface for students and faculties.
- Increasing the efficiency of project record management.
- Decrease time required to access and deliver notices.
- To make the system more secure.

#### **1.5. APPLICATIONS**

- It allows the students to view their project details.
- Allows students to upload the abstract and report.
- Allows the admin to push notifications.
- Allows panel members to enter the marks and comments to the students.
- Allows guide to view the progress of their students.

## 2. TOOLS AND ENVIRONMENT

This chapter focuses on the various system specifications and tools, packages which are used for the implementation purpose.

### 2.1. SOFTWARE REQUIREMENTS

The software technology used in this project is Ruby on rails. **Ruby on Rails** is a server-side web **application** framework written in **Ruby** programming language. It is a model-view-controller framework that provides a structure for a database and web pages. One can develop an **application** at least ten times faster with **Rails** than a typical Java framework.

#### 2.1.1. Development Environment

##### 2.1.1.1. Software

- RVM
- Ruby version 2.6.3p62
- Rails version 4.2.11.1
- Ruby mine
- Text editor (eg..sublime or atom)
- MySql

## 2.1.1.2. PROGRAMMING LANGUAGES



Fig.2.1 programming languages used

### 1) HTML

**Hypertext Markup Language (HTML)** is a markup language that web browsers use to interpret and compose text, images, and other material into visual or audible web pages. Default characteristics for every item of HTML markup are defined in the browser, and these characteristics can be altered or enhanced by the web page designer's additional use of CSS.

### 2) CSS

**Cascading Style Sheets (CSS)** is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

### **3) JS**

JavaScript is a programming language commonly used in web development. JavaScript is a client-side scripting language, which means the source code is processed by the client's web browser rather than on the web server. This means JavaScript functions can run after a webpage has loaded without communicating with the server.

### **4) RUBY**

Ruby is an open source programming language. It is also referred to as a scripting language that is dynamic, interpreted and object-oriented as well. It is scalable and projects with large code are easily maintainable. Its main idea is to reduce the complexity of use for the users. Ruby has come up with great features like support to an object-oriented language, inheritance, garbage collection, overloading and exception handling, support to all major platforms, compatible with other languages and scope of variables etc.

### **5) RAILS**

Rails is a development tool which gives web developers a framework, providing structure for all the code they write. The Rails framework helps developers to build websites and applications, because it abstracts and simplifies common repetitive tasks.

## 6) MySql

MySQL, pronounced either "My S-Q-L" or "My Sequel," is an open source relational database management system. It is based on the structure query language (SQL), which is used for adding, removing, and modifying information in the database.

**2.1.1.3. OPERATING SYSTEM** : Windows XP or more.

### **2.1.1.4. FRAMEWORK USED**

#### **1) Bootstrap**

**Bootstrap** is a free and open-source CSS framework directed at responsive, mobile-first front-end web development .Bootstrap is a web framework that focuses on simplifying the development of informative web pages (as opposed to web apps). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project.

#### **2) AngularJS**

**AngularJS-** a-JavaScript-based open-source front-end web framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications. It aims to simplify both the development and the testing of such applications by providing a



framework for client-side model–view–controller (MVC) and model–view–view-model (MVVM) architectures, along with components commonly used in rich Internet applications.

## **2.1.2. Development Tools**

### **1) Devise**

Devise is the cornerstone gem for Ruby on Rails authentication. With Devise, creating a user that can log in and out of your application is so simple because Devise takes care of all the controllers necessary for user creation and for user sessions.

### **2) Figaro**

Figaro is a ruby on rails gem. It uses a YAML file for all your configuration values. It prevents this YAML from being uploaded to the remote repository. Figaro is developed with deployment in mind. Figaro has a friendly interface with Heroku.

### **3) Chartkick**

Chartkick is a gem that integrates with Rails and provides methods to quickly render graphs based on your data. This gem supports Chart.js, Google Charts, and Highchart adapters. Chartkick comes with out-of-the-box support for Groupdate, which simplifies writing some complex grouping queries.

#### **4) Bcrypt**

The Ruby gem, bcrypt, is a secure hash algorithm for safely storing passwords. This will allow you to keep track of which information belongs to which user while managing who has access to it. bcrypt() is a sophisticated and secure hash algorithm designed by The OpenBSD project for hashing passwords. The bcrypt Ruby gem provides a simple wrapper for safely handling passwords.

## **2.2. HARDWARE REQUIREMENTS**

The system chosen for the development of our system consists of Windows OS, in the view of its portability and availability . The system's performance is good when implemented on a machine with Pentium IV with 512MB RAM, and Windows XP or more.

### 3. SYSTEM DESIGN

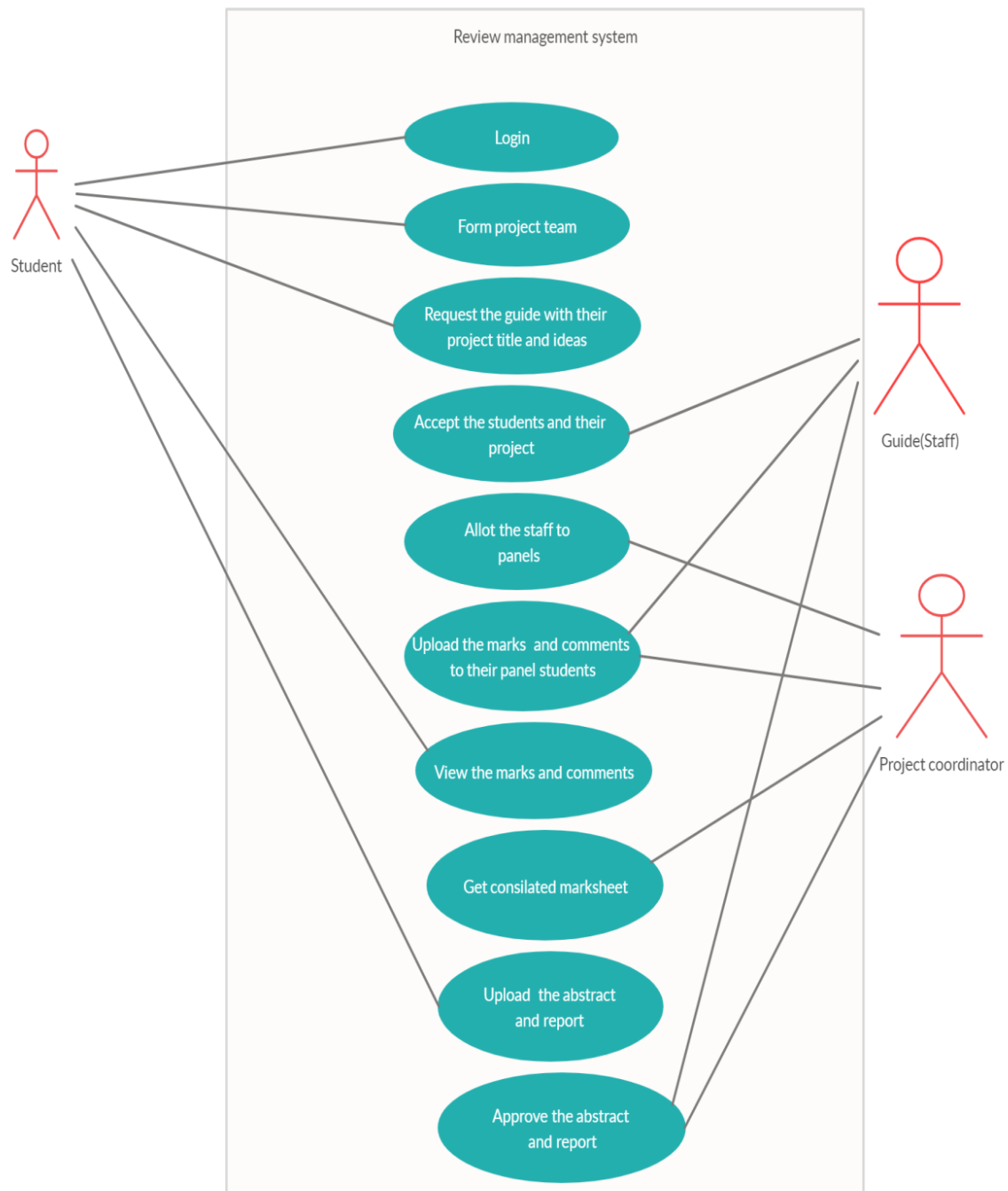


Fig3.1.usecase diagram for rev-sys

### **3.1. WORKING**

In the above use case fig.3.1, there are three actors named student, guide(staff) and a project coordinator. There are a total of ten use cases that represent the specific functionality of a review management system. Each actor interacts with a particular use case.

A student actor can login with their user credentials , form a project team for their project, make a request to their guide with their project title and ideas, view the marks and the comments updated by the panel members and their guide, upload project report and abstract.

The second actor named guide(staff) can login with their username and password, accept the request made by the students and approve their project tile , upload marks and comments to their panel students, approve the abstract and report for the students to whom they guide, view the progress of the students guided by them.

The third actor named project coordinator (admin) can login, allocate staff to their respective panels, upload the marks and comments to their panel students, approve the abstract and report that was approved by their guide, get consolidated marksheet.

### **3.2.DATABASE STRUCTURE**

```
# encoding: UTF-8
# Note that this schema.rb definition is the authoritative source for your
# database schema.
```

```
ActiveRecord::Schema.define(version: 20200223075305) do
```

```

create_table "assignments", force: :cascade do |t|
  t.integer "user_id"
  t.integer "role_id"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

add_index "assignments", ["role_id"], name: "index_assignments_on_role_id"
add_index "assignments", ["user_id"], name: "index_assignments_on_user_id"

create_table "associates", force: :cascade do |t|
  t.integer "student_id"
  t.integer "staff_id"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

create_table "groups", force: :cascade do |t|
  t.string "name"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

# Could not dump table "messages" because of following NoMethodError
#  undefined method `[]' for nil:NilClass

create_table "panel_staffs", force: :cascade do |t|
  t.integer "staff_id"
  t.integer "panel_id"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

add_index "panel_staffs", ["panel_id"], name: "index_panel_staffs_on_panel_id"
add_index "panel_staffs", ["staff_id"], name: "index_panel_staffs_on_staff_id"

create_table "panels", force: :cascade do |t|
  t.string "name"
  t.text "description"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

```

```

create_table "panels_staffs", force: :cascade do |t|
  t.integer "staff_id"
  t.integer "panel_id"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

```

```

add_index "panels_staffs", ["panel_id", "staff_id"], name:
"index_panels_staffs_on_panel_id_and_staff_id", unique: true
add_index "panels_staffs", ["panel_id"], name: "index_panels_staffs_on_panel_id"
add_index "panels_staffs", ["staff_id"], name: "index_panels_staffs_on_staff_id"

```

```

create_table "products", force: :cascade do |t|
  t.string "name"
  t.text "description"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

```

```

create_table "progresses", force: :cascade do |t|
  t.integer "student_id"
  t.string "staff_id"
  t.integer "marks"
  t.string "comments"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
  t.integer "review0"
  t.integer "review1"
  t.integer "review2"
  t.integer "review3"
  t.boolean "approved"
end

```

```

create_table "projects", force: :cascade do |t|
  t.string "name"
  t.string "description"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
  t.integer "staff_id"
  t.integer "panel_id"
  t.integer "group_id"
end

```

```

create_table "relates", force: :cascade do |t|
  t.integer "student_id"

```

```

    t.integer "staff_id"
end

add_index      "relates",      ["student_id",      "staff_id"],      name:
"index_relates_on_student_id_and_staff_id", unique: true

create_table "relations", force: :cascade do |t|
  t.integer "student_id"
  t.integer "staff_id"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

add_index      "relations",      ["student_id",      "staff_id"],      name:
"index_relations_on_student_id_and_staff_id", unique: true

create_table "roles", force: :cascade do |t|
  t.string "name"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

create_table "staff_panels", id: false, force: :cascade do |t|
  t.integer "staff_id"
  t.integer "panel_id"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

add_index "staff_panels", ["panel_id"], name: "index_staff_panels_on_panel_id"
add_index "staff_panels", ["staff_id"], name: "index_staff_panels_on_staff_id"

create_table "staffs", force: :cascade do |t|
  t.string "name"
  t.string "email"
  t.string "emp_id"
  t.boolean "admin"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
  t.string "password_digest"
end

create_table "staffs_students", id: false, force: :cascade do |t|
  t.integer "student_id", null: false
  t.integer "staff_id", null: false

```

```

end

add_index      "staffs_students",      ["student_id",      "staff_id"],      name:
"index_staffs_students_on_student_id_and_staff_id", unique: true

create_table "staffs_students", force: :cascade do |t|
end

create_table "student_groups", force: :cascade do |t|
  t.integer "student_id"
  t.integer "group_id"
  t.datetime "created_at", null: false
  t.datetime "updated_at", null: false
end

add_index "student_groups", ["group_id"], name: "index_student_groups_on_group_id"
add_index      "student_groups",      ["student_id"],      name:
"index_student_groups_on_student_id"

create_table "students", force: :cascade do |t|
  t.string "name"
  t.string "email"
  t.integer "reg_no",      limit: 8
  t.datetime "created_at",      null: false
  t.datetime "updated_at",      null: false
  t.string "password_digest"
end

create_table "users", force: :cascade do |t|
  t.string "name"
  t.string "password_digest"
  t.string "email"
  t.datetime "created_at",      null: false
  t.datetime "updated_at",      null: false
end

end

```



## **4. IMPLEMENTATION**

### **4.1. PROPOSED WORK**

The proposed Online “REV SYS” An Web Based application that Manages project review System is designed to provide more easier way to the students and the faculties to add and retrieve information quickly. Once the user opens this web application, at the front end all the schedule/notifications are available to everyone in a precise manner. There are mainly four types of users. They are students, guide, admin, panel members. The administrator is the master user; he gets the most number of priorities than the other users. The different functions involve in the case of an administrator are updating information, approvals etc. The administrator can view and approve the various records. Students can use the application with the some authentication. Students can view and enter information about their project. Students can edit their project Profile and update them constantly. Students can very flexibly upload their project title, abstract and report using this application. Student will be kept in touch by a notification using messages. Panel members can view the deadlines met by the students and can upload the marks and attendance for the review process. Guide can take Attendance using the

application, and also update the performance of student. Administrator is connected with this system using web application. He has centralized control on this system. He will have the authority to allocate the panel members to their respective domains. He arranges the review dates and sends the notices using this system.

## **4.2.FUNCTIONALITIES**

### **ADMIN VIEW**

- Post notifications.
- View the documents uploaded and approved by the guide.
- View the marks and attendance uploaded by the panel.
- View instantaneous reports after every review.

### **STUDENT VIEW**

- View the attendance and marks scored by them .
- Upload the project title(needed to be approved by the guide).
- Upload the abstract (needed to be approved by the guide).
- Upload the report (needed to be approved by the guide).

- View the presentation formats.
- Receive notifications.

#### **GUIDE VIEW:**

- Year wise projects guided by them.
- Approve the documents uploaded by their project students.
- Monitor Project and view the following details like project title, report, abstract, student details.

#### **PANEL VIEW:**

- To Upload the review marks and attendance.
- View the deadlines met by the students.
- Update the comments.
- Updating the current status as excellent, satisfied and needs improvement.

### **4.3. SOURCE CODE:**

#### **MODEL-VIEW-CONTROLLER**

**Rails** is also based on **MVC** pattern. It basically works as following: Requests first come to the controller, controller finds an appropriate view and interacts with model which in turn interacts with database and send response to controller. This is done to separate internal representations of

information from the ways information is presented to and accepted from the user. This kind of pattern is used for designing the layout of the page.

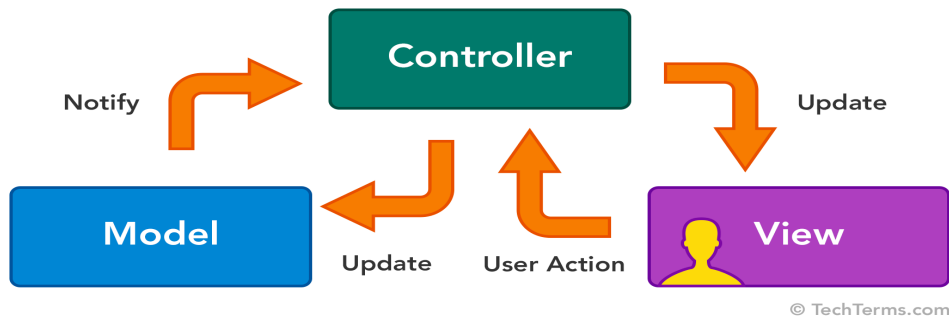


Fig4.1. Rails MVC architecture

#### 4.3.1. MODEL

PANEL\_STAFF

```
class PanelsStaff < ActiveRecord::Base
  validates_uniqueness_of :panel_id, :scope => :staff_id
  belongs_to :staff
  belongs_to :panel
  def emp_id
  end
  def emp_id=(formemp)
    @sta=Staff.find_by_emp_id(formemp)
    @staid=@sta.id
    self.staff_id= @staid
  end
end
```

```

end
USER
class User < ActiveRecord::Base
has_many :assignments
has_many :roles, through: :assignments
def role?(role)
roles.any? { |r| r.name.underscore.to_sym == role }
end
end

```

#### 4.3.2.VIEW

```

<%= form_for(@associate ,url: associate_index_path) do |f| %>
  <% if @associate.errors.any? %>
    <div id="error_explanation">
      <h2><%= pluralize(@associate.errors.count, "error") %> prohibited relations
from being saved:</h2>
      <ul>
        <% @associate.errors.full_messages.each do |message| %>
          <li><%= message %></li>
        <% end %>
      </ul>
    </div>
  <% end %>
<div class="field">
  <%= f.label :Student_regno %><br>
  <%= f.number_field :regno %>

```

```

</div>
<div class="field">
  <%= f.label :Staff_emp_id %><br>
  <%= f.text_field :emp_id %>
</div>
<div class="actions">
  <%= f.submit %>
</div>
<% end %>

```

### 4.3.3.CONTROLLER

#### APPLICATION CONTROLLER

```

class ApplicationController < ActionController::Base
  # Prevent CSRF attacks by raising an exception.
  # For APIs, you may want to use :null_session instead.
  protect_from_forgery with: :exception

  def current_user
    @current_user ||= Staff.find(session[:user_id]) if session[:user_id]
  end

  def signed_in?
    !current_user.nil?
  end

  def admin_user
    redirect_to root_path unless current_user.admin?
  end

  def current_student

```

```
@current_student ||= Student.find(session[:student_id]) if session[:student_id]
end
def signed_in_student?
  !current_student.nil?
end
def current_user?(user)
  user == current_user
end
```

## 5. SCREENSHOTS AND RESULTS

### 5.1 PANEL VIEW

Sign in page shown in Fig.5.1. allows students and staff to securely login using their user credentials

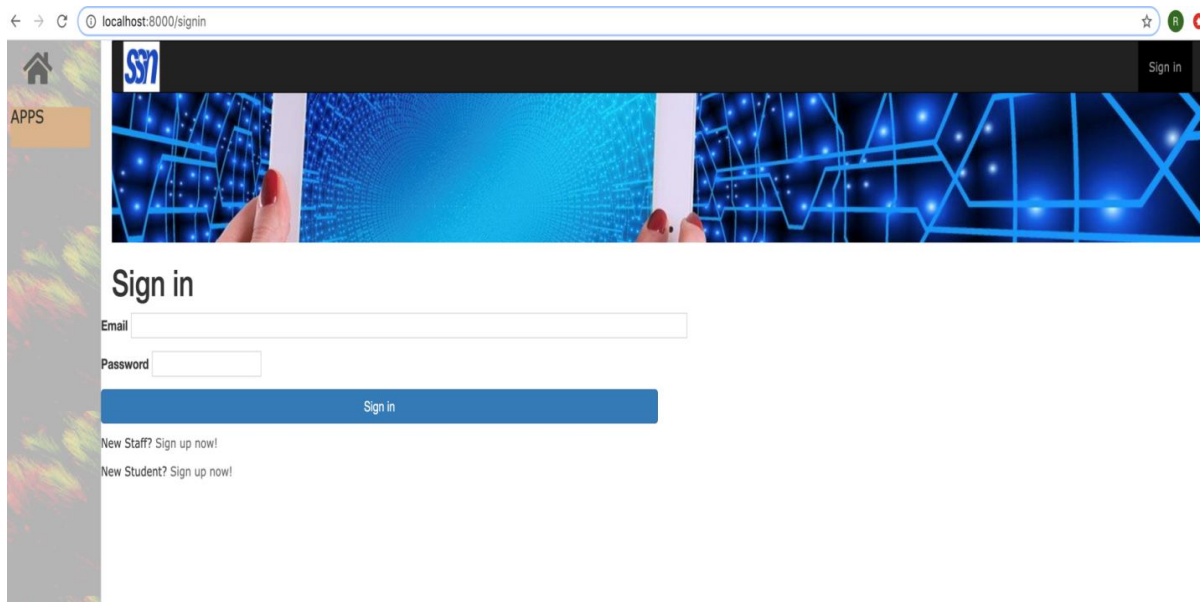


Fig.5.1.Signin page

Fig.5.2. Exhibits the Panel members view of the panels allocated to them by the project coordinator

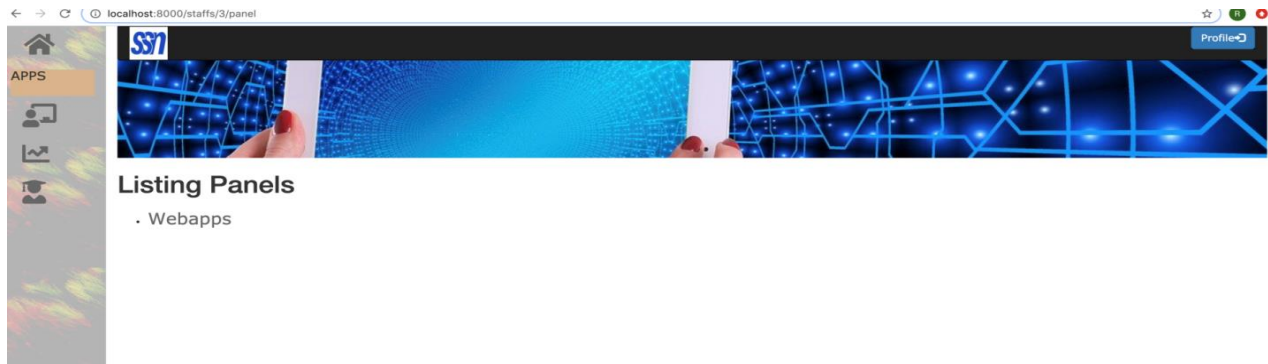


Fig.5.2. Panels allocated to panel members

Fig.5.3.Shows the list of students belonging to a particular panel to the respective panel member and they can also view the profile of the students.

Project details	Student id	Student name	Guided by	Feedback
<b>Title:</b> Context Management using IoT <b>Description:</b> Context Management using IoT	312216205004	Aishwarya S	Dr.S.Sasirekha	Fill feedback
<b>Title:</b> Detection of Diabetic Retinopathy <b>Description:</b> Detection of Diabetic Retinopathy	312216205031	Hemnath D	Dr. K.R. Uthayan	Fill feedback
<b>Title:</b> Data analysis <b>Description:</b> Data analysis	312216205010	Anuraag R	Dr. K.R. Uthayan	Fill feedback
<b>Title:</b> Data analysis <b>Description:</b> Data analysis	312216205013	Ashwin Dhanasamy	Dr. K.R. Uthayan	Fill feedback
<b>Title:</b> AR hotel booking <b>Description:</b> AR hotel booking	312216205003	Achyuth Ramesh	Dr. K.R. Uthayan	Fill feedback
<b>Title:</b> AR hotel booking <b>Description:</b> AR hotel booking	312216205017	Balaji V	Dr. K.R. Uthayan	Fill feedback
<b>Title:</b> Owasp mobile vulnerabilities	312216205029	Harini M	Dr. T. Shanmugapriya	Fill feedback

Fig.5.3.List of students belonging to a particular panel



Fig.5.4.exhibits the panel members functionality of providing the marks and comments to their panel students.

NAME:  
Aishwarya S

REGISTER NUMBER:  
312216205004

Marks

Comments

Record ur Opinion

Fig.5.4.Panel members updating marks and comments

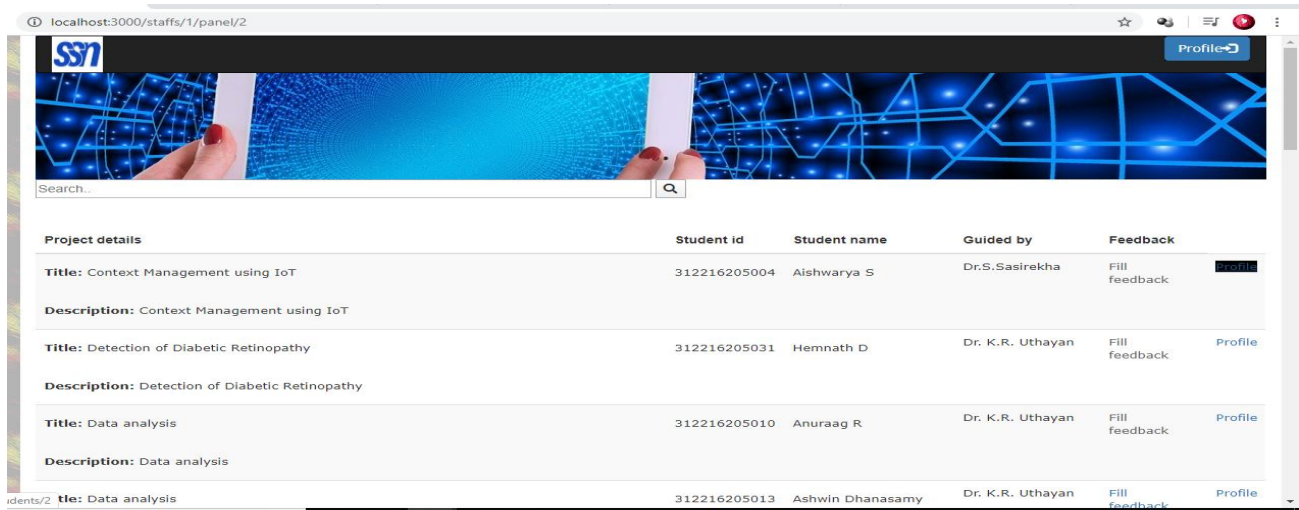
Functionality of the panel members to edit the marks and comments provided by them is shown in Fig.5.5

Listing progresses by you

Student Name	Marks	Comments	Created at	Last Updated	
Aishwarya S	68	good	2020-02-06 17:09:58 UTC	2020-02-06 17:09:58 UTC	<a href="#">Edit</a>
Hemnath D	89	fair	2020-02-06 17:10:27 UTC	2020-02-06 17:10:27 UTC	<a href="#">Edit</a>
Anuraag R	30	good	2020-02-06 17:10:46 UTC	2020-02-06 17:10:46 UTC	<a href="#">Edit</a>
Ashwin Dhanasamy	90	good	2020-02-06 17:11:03 UTC	2020-02-06 17:11:03 UTC	<a href="#">Edit</a>
Achyuth Ramesh	56	good	2020-02-06 17:11:34 UTC	2020-02-06 17:11:34 UTC	<a href="#">Edit</a>
Balaji V	89	good	2020-02-06 17:11:53 UTC	2020-02-06 17:11:53 UTC	<a href="#">Edit</a>

Fig.5.5.Editing marks and comments

Fig.5.6.Shows the panel members view of the abstract uploaded by their panel members.

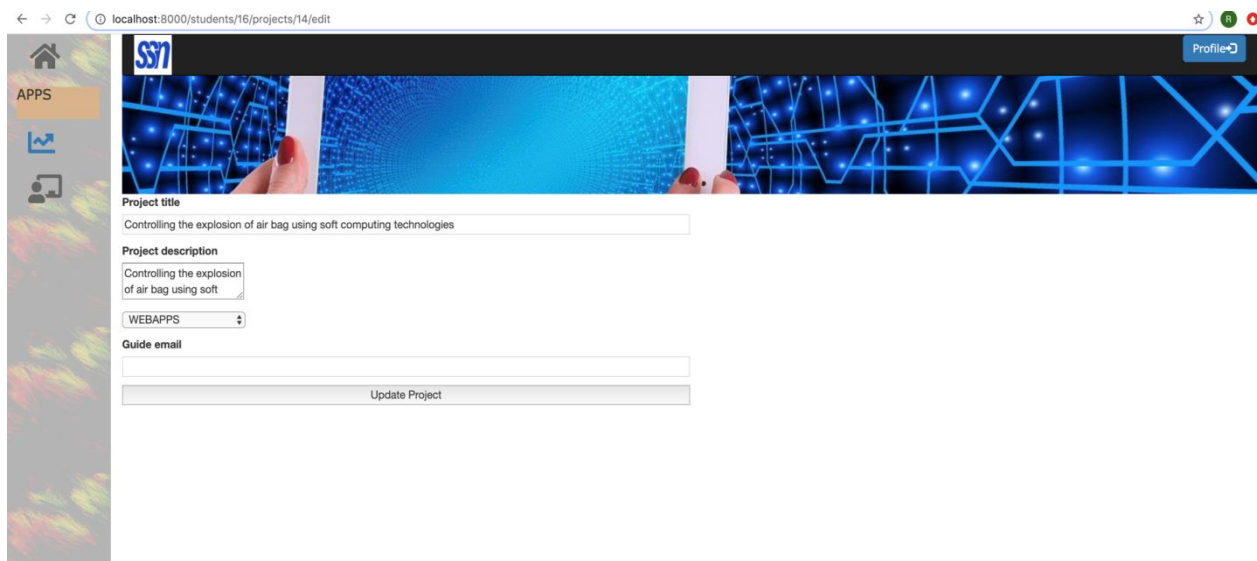


Project details	Student id	Student name	Guided by	Feedback
<b>Title:</b> Context Management using IoT <b>Description:</b> Context Management using IoT	312216205004	Aishwarya S	Dr.S.Sasirekha	<a href="#">Fill feedback</a> <a href="#">Profile</a>
<b>Title:</b> Detection of Diabetic Retinopathy <b>Description:</b> Detection of Diabetic Retinopathy	312216205031	Hemnath D	Dr. K.R. Uthayan	<a href="#">Fill feedback</a> <a href="#">Profile</a>
<b>Title:</b> Data analysis <b>Description:</b> Data analysis	312216205010	Anuraag R	Dr. K.R. Uthayan	<a href="#">Fill feedback</a> <a href="#">Profile</a>
<b>Title:</b> Data analysis	312216205013	Ashwin Dhanasamy	Dr. K.R. Uthayan	<a href="#">Fill feedback</a> <a href="#">Profile</a>

Fig.5.6.View abstract uploaded by students

## 5.2. STUDENT VIEW

Fig.5.7.Shows the students functionality of choosing their respective domain and entering the project details.



← → ↻ localhost:8000/students/16/projects/14/edit

Project title  
Controlling the explosion of air bag using soft computing technologies

Project description  
Controlling the explosion of air bag using soft

WEBAPPS

Guide email

Update Project

Fig.5.7.Students entering their project details

Fig.5.8.Describes the ability of the students to view their updated project information.

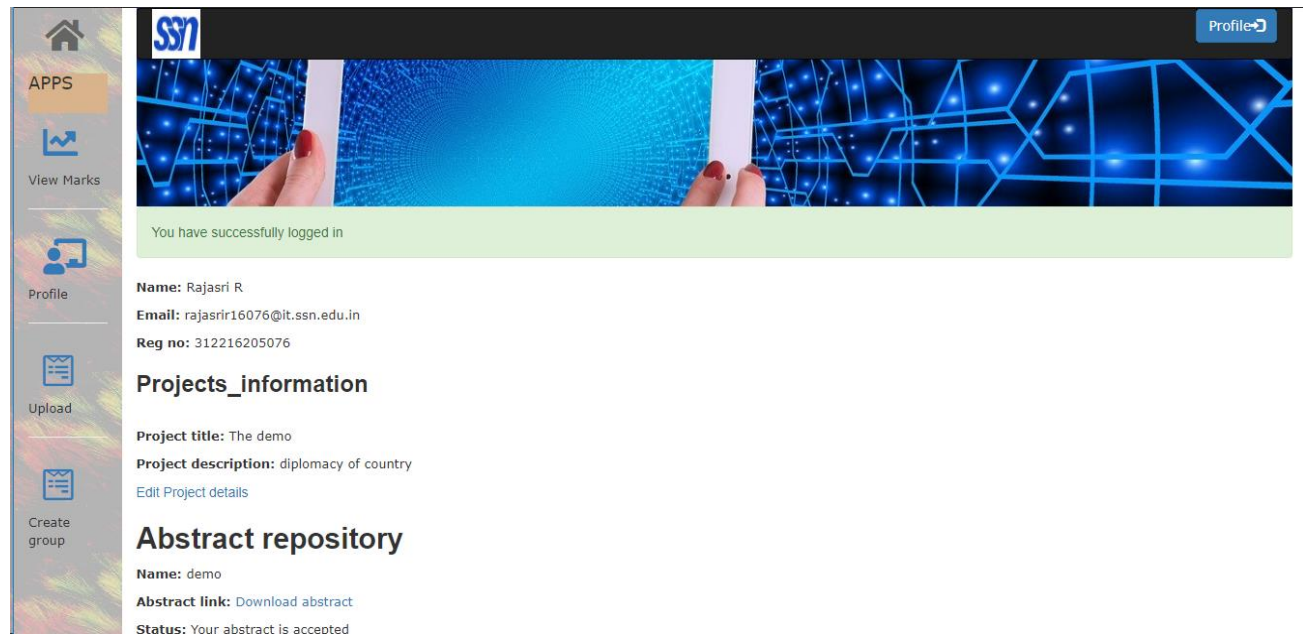


Fig.5.8.Students viewing their project information

Fig.5.9.Exhibits students view of seeing their marks and comments updated by the panel members.

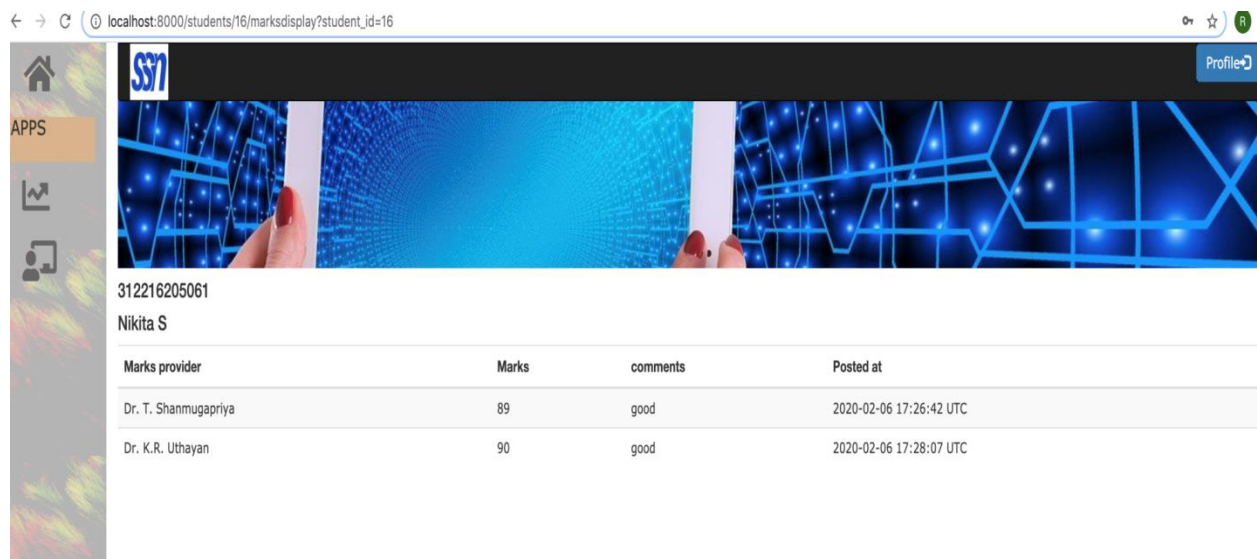


Fig.5.9.View of marks and comments updated

Fig.5.10 Describes the students functionality of uploading their project abstract and report.

Fig.5.10.Uploading the abstract and report

Fig.5.11.Describes the students functionality of checking the status of their abstract and report whether they are approved ,rejected or waiting for review by the guide and the admin.

Fig.5.11.Status check of project details



### 5.3.GUIDE VIEW

Fig.5.12. Shows the guide view of seeing the list of requested students and can accept the students request and make them as their team.

We request you as our guide

Team Name: the lit

Team Members

**Member 1:**  
Sridevi.M  
312216205101

**Member 2:**  
Rajasri R  
312216205076

**Project information**  
**Project Name:** The demo  
**Description:** diplomacy of country

ACCEPT

Fig.5.12.Guide view of the requested students list

Fig.5.13.Shows the functionality of the guide of viewing the list of students guided by them.

Student id	Student name	Project	Feedback
312216205029	Harini M	<b>Title:</b> Owasp mobile vulnerabilities <b>Description:</b> Owasp mobile vulnerabilities	<a href="#">Enter Attendance</a> Marks of the student
312216205007	Akash B A	<b>Title:</b> Owasp mobile vulnerabilities <b>Description:</b> Owasp mobile vulnerabilities	<a href="#">Enter Attendance</a> Marks of the student
312216205101	Sridevi.M	<b>Title:</b> review management system <b>Description:</b> review management sysytem	<a href="#">Enter Attendance</a> Marks of the student
312216205076	Rajasri R	<b>Title:</b> review management system <b>Description:</b> review management system	<a href="#">Enter Attendance</a> Marks of the student

Fig.5.13.Guide view of the list of students guided by them

Fig.5.14.Shows the functionality of the guide to view their students mark and download overall mark details in the excel format.

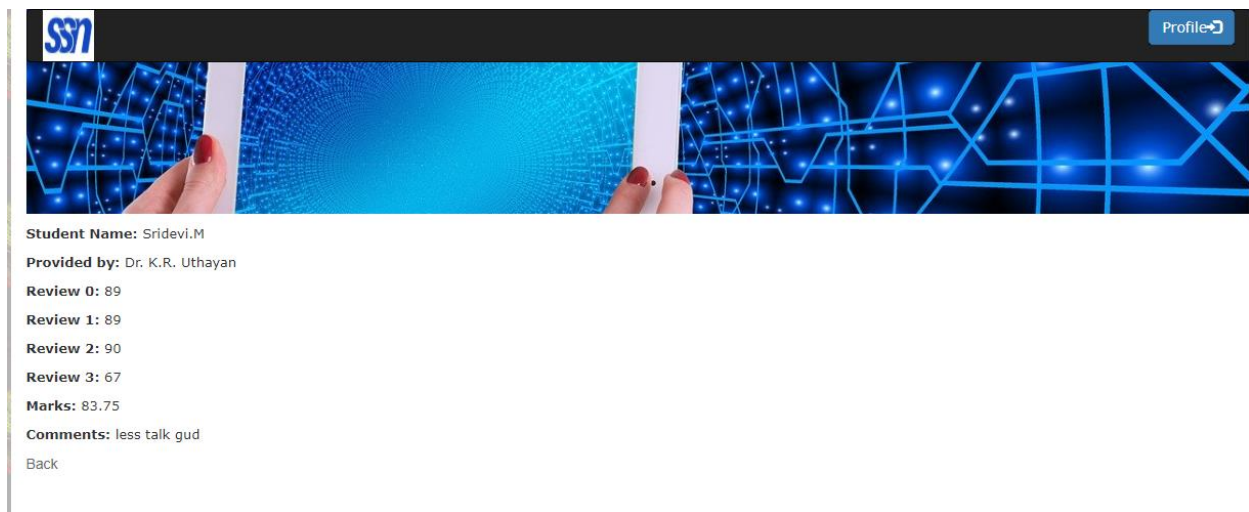


Fig.5.14.Guide can view their students marks

Fig.5.15.Shows the ability of the guide to view the abstract and report uploaded by their team.

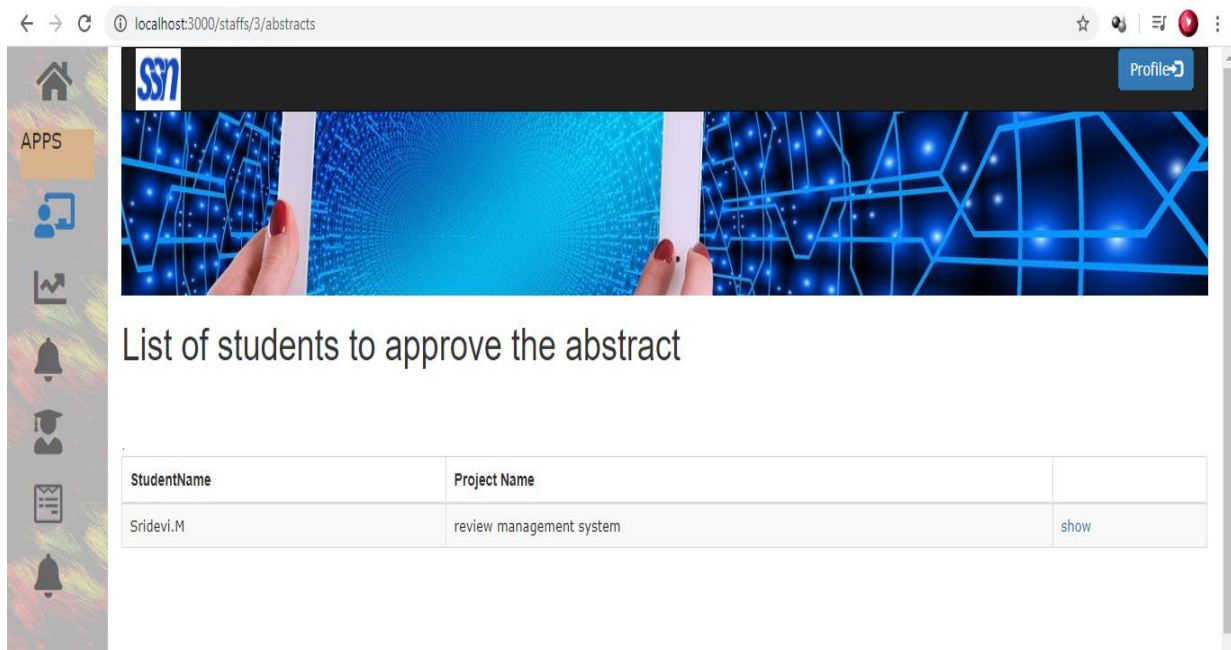


Fig.5.15.View abstract and report uploaded.

Fig.5.16.Describes the functionality of the guide to either accept or reject the abstract uploaded by their students.

localhost:3000/staffs/3/abstracts/31

Email: sridevim16101@it.ssn.edu.in  
Reg no: 312216205101

Projects\_information

Project title: review management system  
Project description: review management sysstem

**Abstract repository**

Name: revurej  
Abstract link: Download abstract Uploaded at 2020-03-12 10:25:44 +0530 and approved\_at 2020-03-12 10:27:12 +0530.

Name: abstract  
Abstract link: Download abstract Uploaded at 2020-03-13 13:25:23 +0530 and approved\_at 2020-03-13 13:27:35 +0530.

**ABSTRACT**

first review abstract  
[Download abstract](#)

ACCEPT  
REJECT

Fig.5.16.Accept or reject the abstract

## 5.4. ADMIN VIEW

Fig.5.17.Shows the ability of the admin to add new staff to the panel.

localhost:8000/staffs/new

APPS

**To upload bunch of staff**

[Choose file](#) No file chosen

Create Staff

**New Staff**

Name  
Email  
Emp

Create Staff

Back

Fig.5.17.Admin can add new staff

Fig.5.18.Shows the overview of the review marks summary that can be viewed by the admin for the respective student.

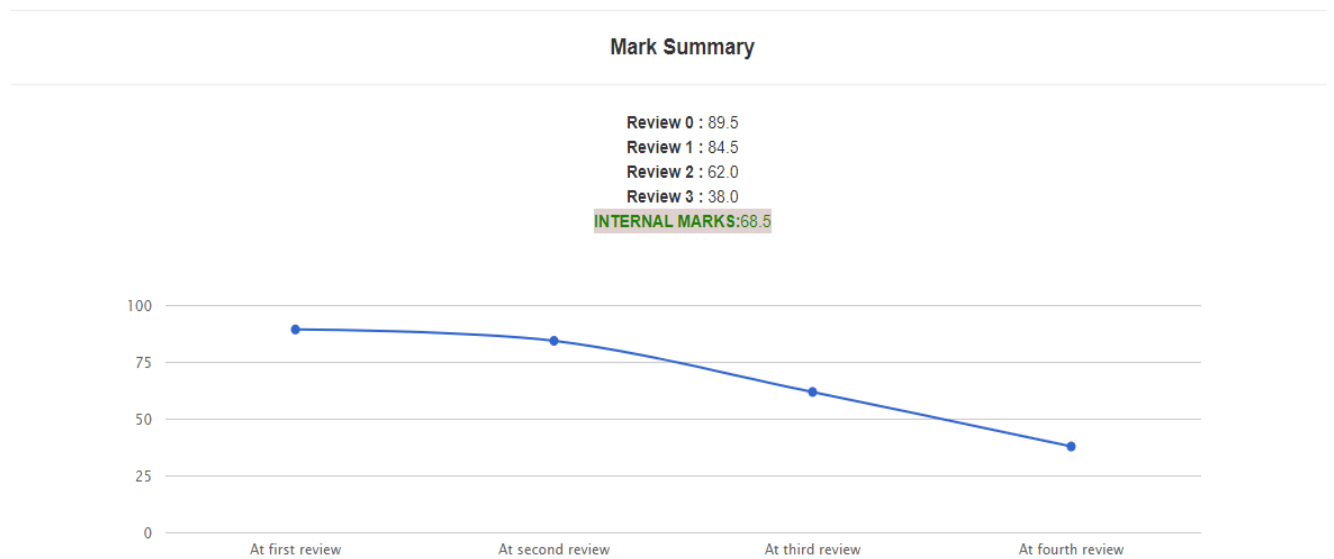


Fig.5.18.Overview of the review marks summary

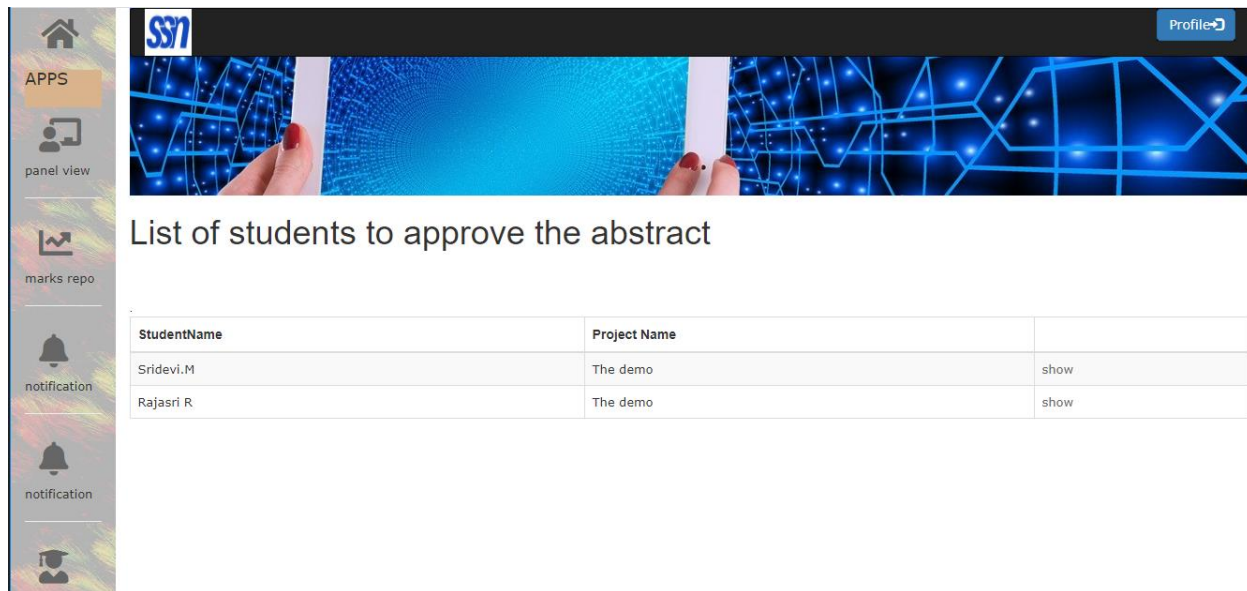
Fig.5.19.Shows the ability of the admin to view the overall progress of the report and abstract by viewing the uploaded and approved dates.



Fig.5.19.Overall progress of report and abstract



Fig.5.20.Describes the functionality of the admin to finalize the uploaded abstract and report.



The screenshot displays the REV SYS application interface. On the left is a sidebar with icons for Home, APPS, panel view, marks repo, and two notification icons, along with a graduation cap icon at the bottom. The main content area features a header banner with the 'SSM' logo and a 'Profile' button. Below the banner, the title 'List of students to approve the abstract' is shown. A table lists two students: Sridevi.M and Rajasri R, both with 'The demo' as their project name. Each row has a 'show' button in the third column.

StudentName	Project Name	
Sridevi.M	The demo	show
Rajasri R	The demo	show

Fig.5.20.Finalize the abstract and report

## 6. CONCLUSION

This REV SYS application is automating the existing manual system. This is a paperless work. It can be monitored and controlled remotely. In this project a repository for managing project reports has been implemented. Along with the features like sharing the review comments updated by the panel members in an instant to admin, guide and students has been implemented. And it also provide facilities to store and display report statements of attendance and marks instantaneously to all the participants and decrease the space between student and college faculties.

It reduces the man power required and provides accurate information always. All years together done projects can be saved and can be accessed at any time. Thus it is a great need for a fast, reliable, efficient and easy automated system which will help in updating and provide the best way for interaction in short duration of time. Thus increasing the efficiency of college project review management. . So it is better to have a Web based project review management system.

## **REFERENCES**

[1] M.A. Norasiah and A. Norhayati.“Intelligent student information system”. 4th International conference on telecommunication technology proceedings, Shah Alam, Malaysia, 0-7803-7773-7/03 2003 IEEE.

[2] Jin Mei-shan<sup>1</sup> Qiu Chang-li<sup>2</sup> Li Jing<sup>3</sup>. “The Designment of student information management system based on B/S architecture”. 978-1-4577-1415-3/12 2012 IEEE.

[3] Zhibing Liu, HuixiaWang,HuiZan“Design and implementation of student information management system.” 2010 International symposium on intelligence information processing and trusted computing.978-0-7695-4196- 9/10 IEEE.

[4]A Survey on “SMART CONNECT” an Android and Web Based Application for College Management System International Journal of

Science, Engineering and Technology Research (IJSETR), Volume 3, Issue 11, November 2014.

[5] “Mobile Phone Based Attendance System”, by Shraddha S.Chawhan<sup>1</sup>, Mangesh P. Girhale<sup>2</sup>, Gunjan Mankar<sup>3</sup>, IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN:2278-0661, p- ISSN: 2278-8727 Volume 10, Issue 3 (Mar. -Apr. 2013).