Yosemite

Group 3: Automated Answer Checker

Mentor Name : Saurabh Tiwari

Problem Statement

- To develop an automatic answer checker application that checks and marks written answers similar to a human being.
- This software application is built to check subjective answers in an online examination and allocate marks to the user after verifying the answer.
- The system requires you to store the expected answer for the system. This facility is provided to the faculty.
- Both the answers need not be exactly same word to word.

Yosemite

- Yosemite is a Web-Based Automatic Answer-Checker Application that runs on a Machine Learning based Model.
- It looks for Semantic Similarity in the answers provided by the students.
- It provides a platform for students and faculties to Submit and Review the assignments.
- It checks subjective answers provided by the students through an online Assignment and allocate marks to them after verifying the answer.

Functional Requirements

- Student
  - All students will have a unique username and password to access their account.
  - They can submit answers to a particular Assignment if it’s active via the link received.
  - They can also view their previous scores as well from their account.
  - Furthermore, they can also raise a query to a particular question if they feel that the marking is not appropriate.
  - He will be able to see modified marks(if any) after he raised his query flag.

Functional Requirements

- Faculty
  - The faculty will have the ability to create a new Assignment where they'll list the questions and their answers.
  - Other assignment details such as Deadline and Marks(each Question) will also be need to be provided.
  - The faculty gets the similarity measure between the original answer and the submitted answer and the evaluated score as well.
  - Thus, he or she can accordingly allocate the marks provided by the model to the students or can even modify them.
  - The faculty can provide their answers for a particular question which will be later used to check the student's answers.
  - Once an assignment deadline is finished they can request the answer statistics and can accordingly allocate marks (or keep it the same, if they wish to).
  - They can also see the query flags raised by students and change marks if required for particular questions.

Functional Requirements

- Administrator:
  - He or She has access to the entire database and can modify required fields as and when needed.
  - Some of his functionalities include:
    - Add, Update and Remove Users(Faculty and Students)
    - Edit their Profiles
    - Add, Update and Remove Assignments and the Marks allocated
- Machine Learning Model
  - A Natural Language Processing based ML Model will work in the backend to compare the actual answers to the answers given by students. The Model shall have a holistic approach while marking students and will consider Semantic Similarity as well as its relevance to the actual answer while grading.

Non-Functional Requirements

- The service will be used by multiple users, so the system has **High Scalability**, well enough to handle 1000’s of students.
- The service will have 3 possible login **Authentications**: one for each - administration, faculty, and the students.
- **Reliable** and fault-tolerant storage that is hosted in a secure environment.
- **Secure authentication** with different user roles to provide levels of authorizations to access data.
- The website will be **accessible** to all Operating Systems and browsers. It is recommended to use the latest versions of browsers but will also support older versions as well. It will not create any conflict with any other website or software running within the same system.
- 99.99% uptime - High **availability** 24x7

Non-Functional Requirements

- Seamless rollouts of new features, no manual updating required on the user end.
- Natural Language Processing Model will provide a Similarity Index in an acceptable error bound and time limit.
- All students should receive a link(on registered email id) to submit their answers when a particular assignment is assigned to them.

Elicitation Techniques

- Brainstorming
  - Initially several ideas were gathered and healthy discussions were held.
  - Features of the application were conversed.
  - Issues were put on the table and tasks were assigned to each team member
- Observational Data
  - Platform had to be decided based on keeping in mind the already available assignment Platforms like Google Forms, Mettl and moodle .
  - Usability of the platform was given priority
- Literature Survey
  - Online Research papers based on Machine Learning Models were approached
  - Natural Language Processing model which returned a Semantic Similarity Index attracted us the most

Stakeholders

- Students
- Course Instructors (Faculty)
- Administrator

Actors

- Administrator:
- can edit the database
- has full access to the system
- can edit their personal profile details

Actors

- Faculty
- creates a new assignment
- Sends link for answer submission to the desired set of students
- provides the expected answer for each assignment question
- can view the Similarity Index for each answer for each student
- can allocate marks to students based on the software’s evaluation
- can view query flags raised by students
- Can edit their personal profile details

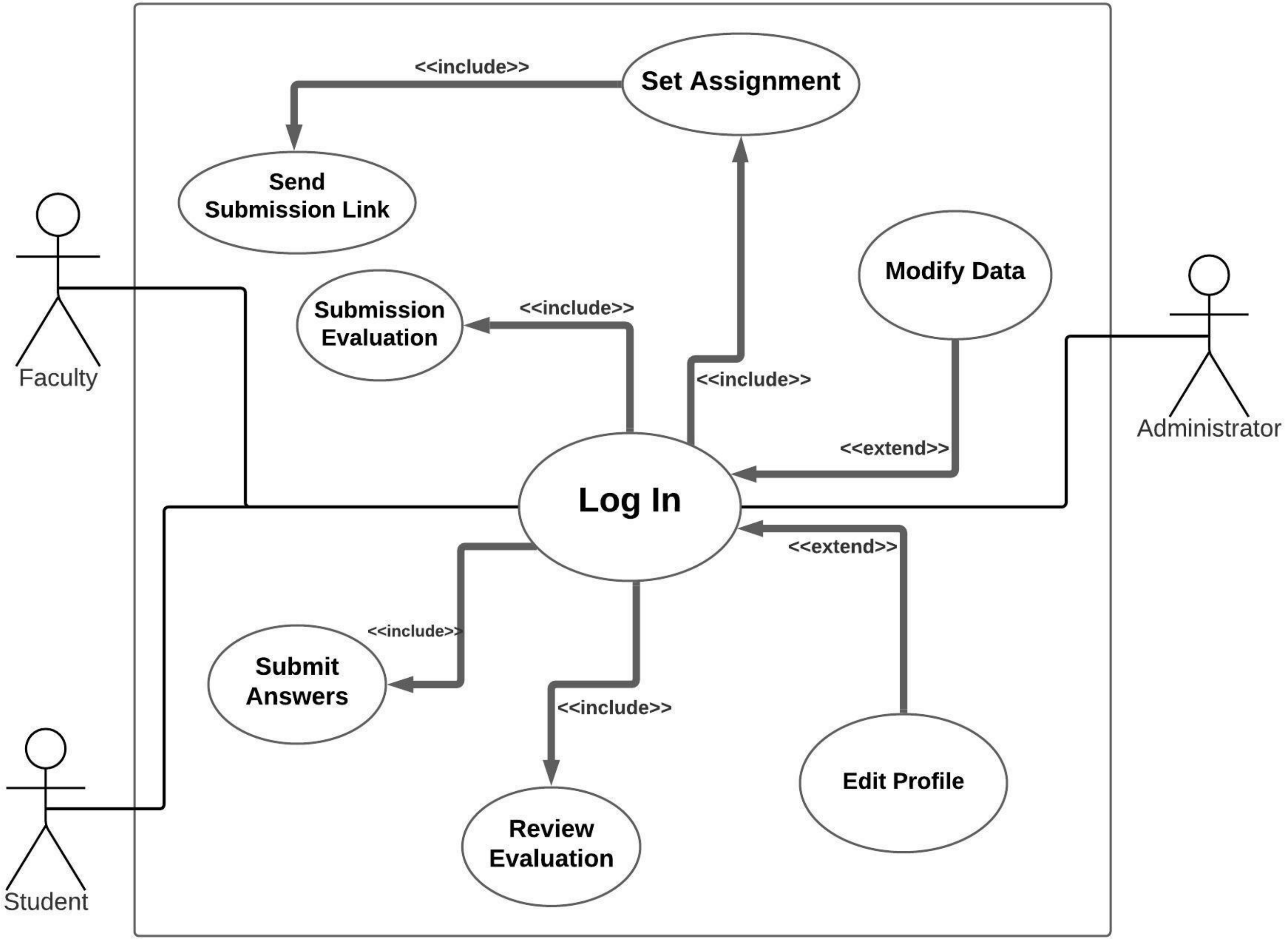
Actors

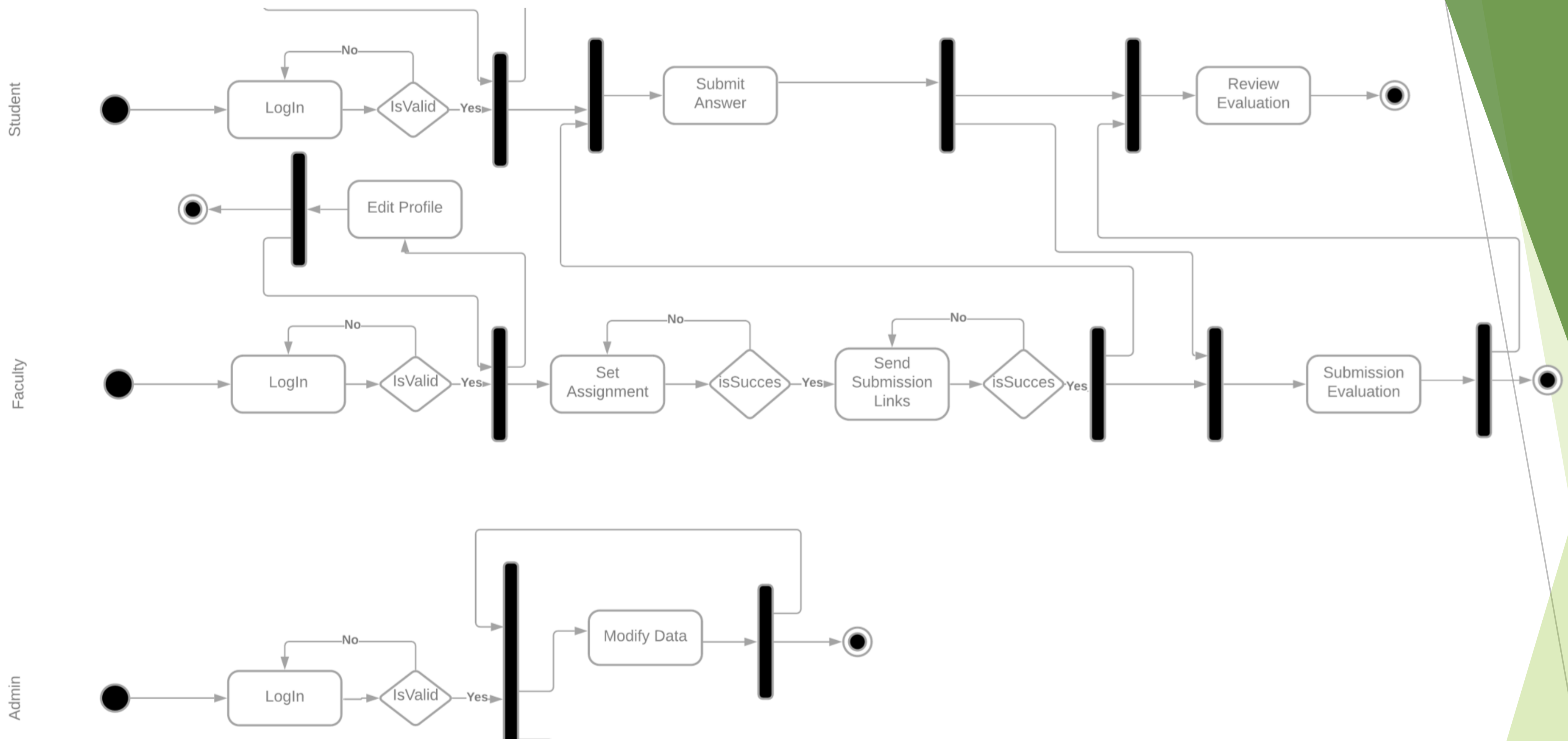
- Students :
  - can attempt the assignment via the given link
  - can view their past assignment’s score and its detailed evaluation
  - can view the answer provided by faculty
  - raise a query for a particular set of questions if any discrepancy is found
  - Can edit their personal profile details

Use Case Model

Use Cases

- Set Assignment
- Send Submission Link
- Submit Answer
- Submission Evaluation
- Review evaluation
- Edit Profile
- Log In
- Modify Data





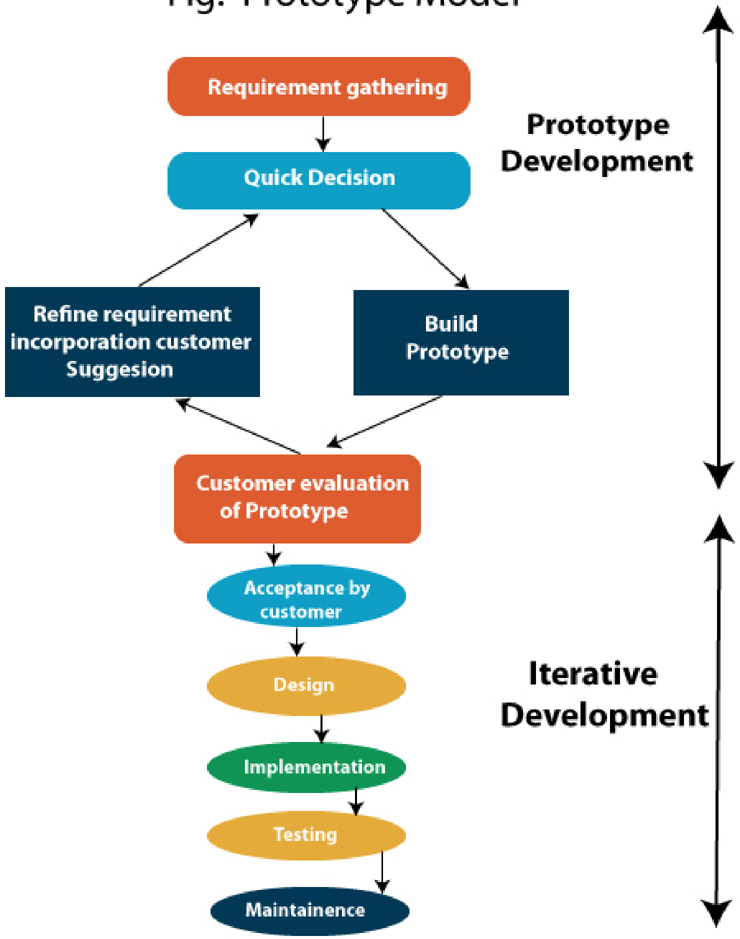
Activity Diagram

Can view full diagram here:
https://drive.google.com/file/d/14j9ybbI47l8adS8xCcFUTLZcOtURC79b/view?usp=sharing

Process Model

- **Phase 1:** Prototype Model
- **Phase 2:** V Model

Fig: Prototype Model



Requirement gathering

Quick Decision

Prototype Development

Refine requirement incorporation customer Suggesion

Build Prototype

Customer evaluation of Prototype

Acceptance by customer

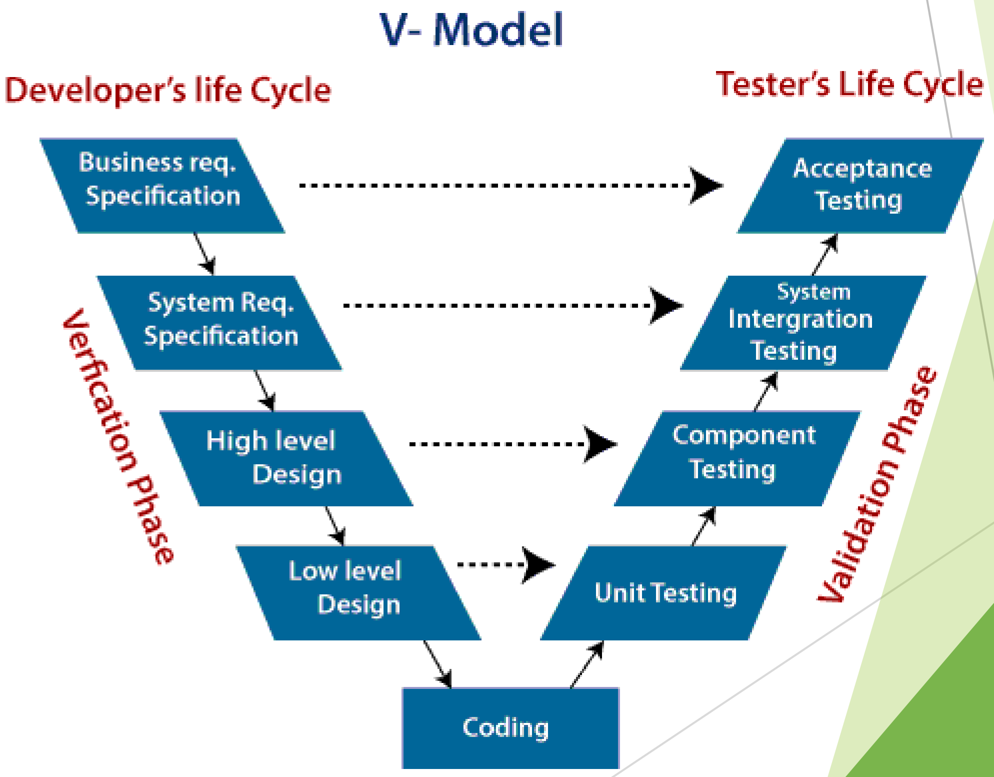
Design

Implementation

Testing

Maintainence

Iterative Development



V- Model

Developer's life Cycle

Tester's Life Cycle

Business req. Specification

Acceptance Testing

System Req. Specification

System Intergration Testing

High level Design

Component Testing

Low level Design

Unit Testing

Coding

Verfication Phase

Validation Phase

Future Developments

- Transition from **Prototype model** to **V model**
- Initiate various kinds of Software Testing & Non-functional Testing
- Finish up implementations of the remaining Functional and Non-Functional Requirements
- Deploy the remaining components of System Architecture
  Employ **DevOps** best practices in the process
- Finalize remaining set of Documentations and Diagrams of system
- Evaluate stakeholder's feedbacks on the system prototype

Team Members and Contributions

Name	Contributions
Meet Patel - 201801415	• Leader, Project Manager • Manage Infrastructure and Deployments • Frontend and Backend Development
Chirag Gupta - 201801188	• Backend APIs development • Database Design
Arkaprabha Banerjee - 201801408	• Machine Learning Model research and development • Develop API for deploying the model
Rahil Shah - 201801252	• Machine Learning Model research and development • Develop API for deploying the model
Kartavi Shah - 201801426	• Database Schemas • Populating dummy data into MongoDB Database • Automated Documentations of API
Archit Agrawal - 201801043	• Frontend Development • Homepage of the website

Team Members and Contributions

Name	Contributions
Amruthsai Jilla - 201801069	• Frontend Development • Homepage of the website • Work on data visualizations on Dashboards for different users
Bhagyesh Ganatra - 201801047	• Frontend Development • Making Create Assignment and Submit Assignment forms • Make them functional through backend
Ridham Suvagiya - 201801006	• Frontend Development • Make Dashboards for all users • Handle client-side routing of webapp • Load and Display user-data provided by backend in tabular format • UI Mocks and designs
Udit Meena - 201801095	• UI Mocks, Graphic designs • Populating dummy data into MongoDB Database • Automated Documentations of API

Yosemite GitHub Repository

Mentor: Prof. Saurabh Tiwari

- Front End Repository:

  https://github.com/meet59patel/Group3-Yosemite-frontend

- Back End Repository:

  https://github.com/meet59patel/Group3-Yosemite-backend/

- Machine Learning API Repository:

  https://github.com/meet59patel/Group3-Yosemite-NLP/

GitHub: Issue Based Workflow

- GitHub allows us to use branches along with the master branch that is used to **automatically deploy** the web app.
- Each member has his own branch and can merge the changes via **Pull Requests**, hence having **multiple parallel workflows**.
- Issues are assigned to keep track of tasks, enhancements and bugs

Link to our website

https://yosemite-sen.vercel.app/

“

Thank You

”

Group -3

IT314 Software Engineering