Course Code/Course Title

Group Number

Project Topic

Link to GitHub Repository

Group Leader  
  
**Group Information**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SN | Member’s Name | Registration  Number | Team Role | % Participation |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
| 6 |  |  |  |  |

INTRODUCTION  
  
In Today’s world, the way people discover, choose, demand from restaurant has fundamentally transformed. Our Restaurant Aggregator System aims to bring out solutions providing a bridge between hungry customers and diverse dinning establishment across the city. Unlike traditional food delivery apps that focus primarily on order fulfillment, our aggregator creates a complete ecosystem that enhances the entire dining decision process. This platform serves as a centralized hub where users can browse menus, compare options, read reviews, and place orders from a vast array of restaurants—all through a single, intuitive interface. For restaurants, especially small and medium-sized establishments with limited resources for digital marketing, our platform offers unprecedented visibility and access to a wider customer base without the need for significant technological investment. For customers, it eliminates the frustration of navigating multiple apps and websites, providing a one-stop solution for all their dining needs—whether they're looking for quick delivery, planning a reservation, or seeking new culinary experiences.   
As urban dining landscapes continue to evolve and consumer expectations rise, our Restaurant Aggregator App stands at the intersection of technology and gastronomy, transforming how restaurants operate and how customers discover their next memorable meal.  
  
Aim and Objective

1.Waterfall Development Methodology  
 It is one of the oldest and traditional software development methodologies. It follows a linear and sequential manner. In this model, the outcome of one stage is the input of the next stage and development of the next phase only starts when the previous phase is completed. This model was highly popular during the days of programming due to its certainty in project scope but however, the rigidity of its structure also contributes to high rate of failures for many projects.  
  
2. Agile Development Methodology  
 It is a combination of the iterative and incremental approach. It prioritizes user satisfaction and communication using short sprints and frequent feedback to make software changes. The Agile development Methodology is best for projects with changing requirements, especially in new areas of software development. It lets the developers include new ideas based on market needs assuming they are independent and prefer to work fast.  
  
3. Scrum  
 Scrum is a lightweight framework for agile development. It is subset of agile software development process which is nothing but an iterative and incremental software development technology. Scrum divides the software development process into fixed-length iterations called sprints, which typically last from one to four weeks.

Each sprint consists of four phases:

1. **Sprint planning**: In this stage, the Product owner defines a goal for the sprint at starting time of the sprint.
2. **Daily Scrum Meeting**: Scrum team gathers together in a meeting for around 15 minutes. They specify the progress report and roadblocks.
3. **Sprint Review**: In this stage, the team demonstrates that what is ready for delivery to the customers. A meeting gets conducted between the customer, product owner, and the Scrum Team.
4. **Sprint Retrospective**: In this stage, Feedback is taken about the team that how they worked in the last sprint and what action needs to be taken for improvement.  
     
   4. Kanban  
   kanban is an effective to manage projects and make your workflow even better. The word “kanban” originates from a Japanese language translating to “visual signal” or “card”.  
   The board is divided into sections like “To Do”, “In Progress”, “In Review”, “Done”.  
   It’s easy to understand and helps to keep track of everything and also helps the team to monitor and control each stage and apply changes to improve efficiency.  
     
   5. Extreme Programming (XP)  
   It is an agile project management methodology that targets the speed and simplicity with short development cycle and less documentation. XP’s design allows developers to respond to customer stories, adapt and change in real-time but XP is much more discipline using frequent code review and unit testing to make changes quickly. It’s also highly creative and collaborative, prioritizing teamwork during development stages.  
     
    Difference Between Software Development Methodology  
     
      
   Difference between Agile and Waterfall Software Development Methodology  
     
   1. Approach  
    Waterfall is linear and sequential that is each phase must be completed before the next creating a rigid structure while Agile is iterative and flexible, here development happens in short cycles allowing for flexibility and adaptation.  
     
   2. Planning  
    Waterfall emphasizes upfront planning that is requires detailed planning and documentation before development begins while Agile focuses on planning for short iterations (sprints).  
     
   3. Feedback  
    Waterfall has limited user feedback while Agile incorporates frequent feedback and stakeholder input throughout the project.  
     
   4. Change Management  
    Waterfall makes it difficult to make changes after initial phases while Agile allows for more flexibility in adapting to changes.  
     
   Differences between Scrum and Extreme Programming  
    Scrum is another common type of the Agile methodology managed by a Scrum master. Similar to XP, scrum runs sprints off user stories to develop new products of software features. However, XP is more rigid than Scrum, with strict rules and guidelines that encourage constant contact between developers and the customer. Also, you can use scrum for any process that requires iteration and customer input, whereas you’d only use XP programming.

Reason For choosing The Scrum Methodology  
  
Flexibility  
 Scrum’s iterative and incremental approach allows teams to change requirement and priorities throughout the project. Every 1-4 weeks depending on the duration of the sprint, the scrum team decides what will be worked on in the next sprint. A t any time prior to planning the work of a sprint, the product owner and a number of developers works to refine the product backlog ensuring the most valuable work is available for selection.   
  
Enhanced Collaboration  
 Scrum promotes a collaborative environment through the application of scrum. Scrum events and clear accountabilities associated with scrum roles help to bring collaboration to life. Developers in scrum are multifunctional that is most of the work can be done most people in the team. For more complicated aspects of work e.g solving problem then a collaboration between two or three developers will lead to better outcomes and achieved quicker than one person working alone.  
  
Higher Productivity  
 Self-organizing, collaborative teams – those where developers decide collaboratively who is best placed to perform what work and to support each other in doing it – typically outperform teams managed on a task-by-task basis by a manager. The uplift in productivity arises from collective focus on most important work, openness about challenges involved, courage to make the right decisions.  
  
Continuous Improvement  
 Regular retrospectives allow teams to reflect on their processes and performance and identify areas for improvement. One or two potential improvements in ways of working are introduced into the next Sprint as experiments. If, at the following review, the experiments lead to improvement, they become part of the team's normal way of working, and failures are dropped, potentially in favor of a new experiment. This commitment to continuous improvement helps teams optimize their workflows and enhance overall outcomes.

Review of Related Concepts With Respect To Chosen Topics  
  
User Interface  
 The user interface is designed to enable users to move between screens (home, menus, profiles, carts, checkout) using intuitive tabs. The UI also provides easily accessible search bars and filters (cuisine, price, distance) which help users to find restaurants and dishes quickly. Also, it provides important elements (offers, menu items, carts) which are highlighted, built-in feedback forms and support channels to improve user satisfaction.  
  
Algorithm used  
I) Search algorithm  
Keyword search: Matches user input with restaurant/dish names, description and tags  
Fuzzy Matching: Finds similar results with typos  
Popularity and Relevance: Sorts results by popularity, distance or user rating  
II) Filtering options  
Cuisine, Price, Distance: Users filter by type of food, price range and proximity.  
Dietary Restrictions: Filters for vegetarian, vegan, gluten-free etc  
Rating and Reviews: Users can filter by star rating or number of reviews.  
  
Data Management  
I) Databases: Uses Firebases, Supabase or PostgreSQL for storing user profiles, restaurant information, orders and reviews.  
II) Data Synchronization: Real-time updates ensure users see current menus, prices and order status.  
III) Security and Privacy: User data is encrypted and protected that is privacy policies are transparent.  
  
Review Of Related Literature In Respect To Related Project  
  
Integration Of APIs  
Mapping APIs: Google maps or mapbox for location and directions  
Payment APIs: Stripes, PayPal etc  
Notification APIs: Firebase cloud messaging for order updates  
Analytics APIs: Google Analytics or Mixpanel for tracking user behaviour

METHODOLOGY AND MATERIAL  
  
Research Methodology  
 This section outlines the research method adopted to develop the Restaurant Aggregated System (RAS). The Scrum Agile Methodology is used as a primary research and development framework for developing and designing the Restaurant Aggregated System.  
The Scrum framework was selected because of its iterative nature, flexible in handling changing requirements and emphasis on continuous feedback.  
  
SYSTEM REQUIREMENTS  
Functional Requirements  
1.User management  
2.Restaurant operation like Restaurant subscription, open/close menu, operating hours.  
3.Booking system that is enable table/spot bookings.  
4.Search and Discovery that is the location  
5. Rating system: Allows users to rate the system on a scale 1-5  
6. Payments  
  
Non Functional Requirements  
1.Performance  
2.Security  
3.Reliability  
4.Usuability  
5.Maintainability  
6.Scalability  
  
Technical Requirements  
Hardware: End user devices like computers  
Software: Programming language, databases  
SS  
SYSTEM DESIGN