Project Progress Report

Prudhvi Gudipati – 1002032517, Preetham Reddy Boddu - 1002030625

Project Description:

Smash Tourni is a comprehensive solution designed to streamline the process of managing player registrations and payments for tournaments. Whether you're organizing sporting events, gaming competitions, or any type of tournament, this software will simplify the entire registration and payment workflow.

Objectives:

- To develop a software solution that simplifies and accelerates the player registration process, reducing the time and effort required for both organizers and participants.
- To implement robust security measures to protect payment data and ensure safe and reliable payment processing.
- To ensure the accuracy and integrity of player data by providing features for data validation and correction.
- To enable real-time status updates and notifications to keep all users informed of registration and payment progress.
- To generate comprehensive reports and analytics that offer insights into player registration and payment trends, facilitating informed decision-making.
- To prioritize user satisfaction by creating an intuitive and user-friendly interface that meets the needs of tournament organizers, players, and administrators.
- To develop a highly configurable software solution that can be tailored to the specific requirements of various tournaments and sports.
- To design the software with scalability in mind to accommodate an increasing number of tournaments, players, and payment transactions as the user base grows.

Project Scope

Features and Functionality:

The project scope includes the development of the following features and functionality:

- **Tournament Creation**: Ability for organizers to create and configure tournaments, specifying details like tournament type, date, location, and rules.
- **Player Registration**: A registration system that allows participants to sign up for tournaments, provide essential information, and make payments securely.
- **Payment Processing**: Integration with payment gateways to facilitate secure and seamless payment processing, including options for different payment methods.

- **Bracket Management**: Tools to create and manage brackets, including automated bracket generation and updates as participants register.
- **Notification and Communication**: A system for sending notifications and updates to registered players, including scheduling, rule changes, and event information.
- **Scoring and Results**: A platform for tracking match results, calculating scores, and declaring winners.
- **User Profiles**: User accounts for both organizers and participants, with the ability to manage personal information and tournament history.
- **Data Analytics and Reporting**: Reporting tools to analyze tournament data, participant statistics, and financial information.

User Roles:

The platform will serve the following user roles:

- **Tournament Organizers**: Responsible for creating and managing tournaments, participant registration, and overall event administration.
- **Participants**: Individuals or teams registering for tournaments, making payments, and competing.
- **Administrators**: Responsible for managing system settings, user accounts, and ensuring compliance with regulations.

Feasibility Report

1. Technical Feasibility:

a. Technology Stack:

- The project's technology stack is modern and readily available.

b. Software Development:

- The development of registration, payment processing, bracket management, and other core features is technically feasible.
- Integration with payment gateways and data analytics tools is achievable.
- Cloud-based hosting and scaling options are readily accessible.

c. Security and Compliance:

- Implementing robust security measures and ensuring data privacy can be technically achieved.
- Compliance with data protection regulations is feasible with the right protocols and mechanisms in place.

2. Operational Feasibility:

a. User Adoption:

- The platform is designed to simplify the registration and payment process for tournament participants, making it attractive to users.
- Organizers can efficiently manage tournaments, which can lead to widespread adoption.

b. User Training:

- User training and support systems can be implemented to help users effectively navigate and use the platform.

c. Scalability:

- The platform can be designed for scalability to handle an increasing number of users and tournaments as it gains popularity.

3. Economic Feasibility:

a. Revenue Model:

- The project can generate revenue through various means, such as charging a percentage fee for each registration or offering premium features for organizers.

b. Cost Analysis:

- Costs related to software development, infrastructure, security, and support should be assessed and budgeted.
- Revenue projections should be analyzed to ensure the project is economically viable.

c. Return on Investment (ROI):

- The project can yield a positive ROI through a combination of user adoption and the revenue model, making it economically feasible.

d. Market Potential:

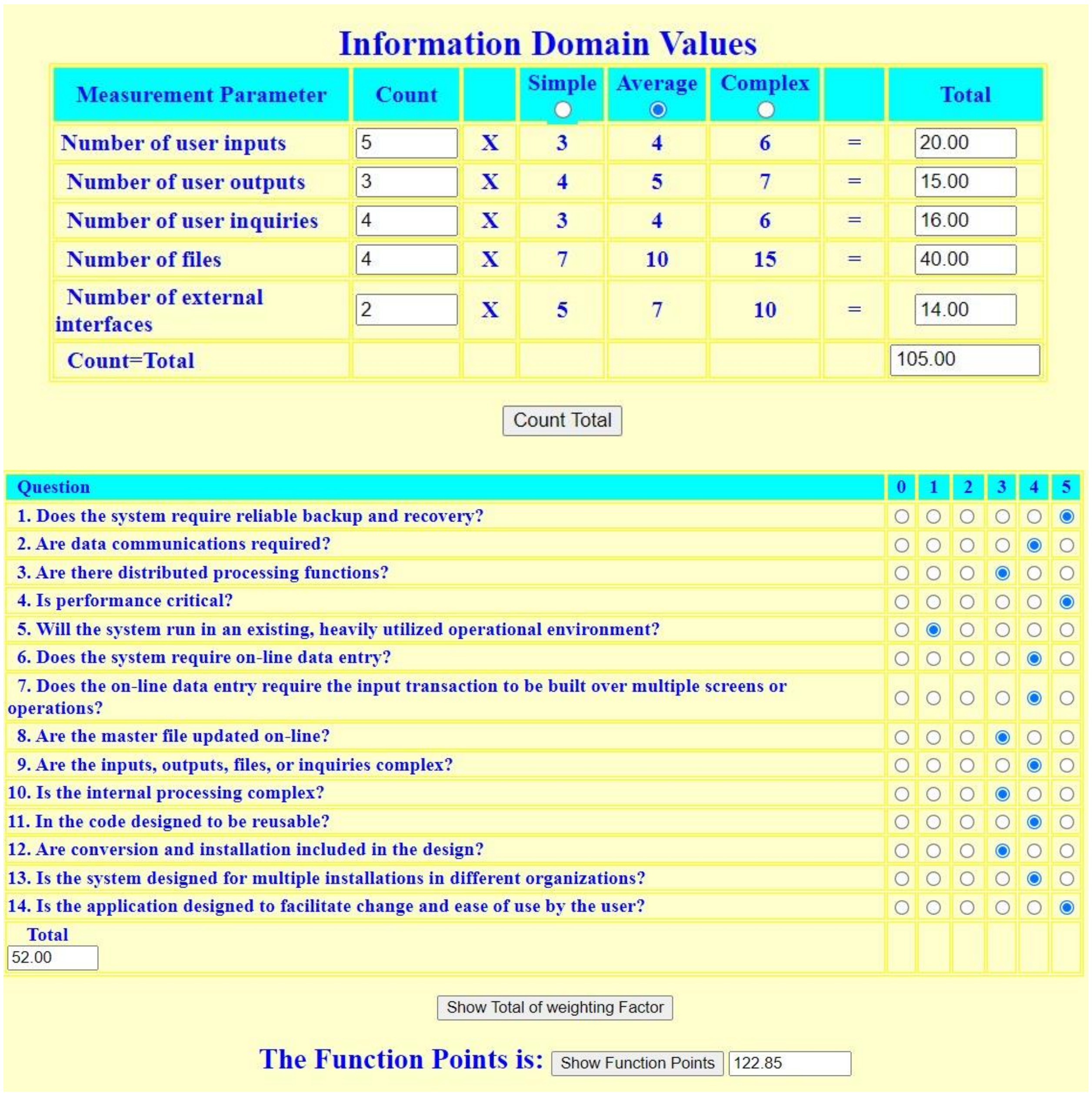
- The demand for tournament management solutions is significant in various sectors, including sports, gaming, and other competitive events, indicating a substantial market potential.

e. Competitive Analysis:

- Assess the competition in the tournament management software space and identify unique selling points and market differentiators.

Cost and time estimation

FP Method



Information Domain Values

Measurement Parameter	Count		Simple ○	Average ●	Complex ○		Total
Number of user inputs	5	X	3	4	6	=	20.00
Number of user outputs	3	X	4	5	7	=	15.00
Number of user inquiries	4	X	3	4	6	=	16.00
Number of files	4	X	7	10	15	=	40.00
Number of external interfaces	2	X	5	7	10	=	14.00
Count=Total							105.00

Question	0	1	2	3	4	5
1. Does the system require reliable backup and recovery?	○	○	○	○	○	●
2. Are data communications required?	○	○	○	○	●	○
3. Are there distributed processing functions?	○	○	○	●	○	○
4. Is performance critical?	○	○	○	○	○	●
5. Will the system run in an existing, heavily utilized operational environment?	○	●	○	○	○	○
6. Does the system require on-line data entry?	○	○	○	○	●	○
7. Does the on-line data entry require the input transaction to be built over multiple screens or operations?	○	○	○	○	●	○
8. Are the master file updated on-line?	○	○	○	●	○	○
9. Are the inputs, outputs, files, or inquiries complex?	○	○	○	○	●	○
10. Is the internal processing complex?	○	○	○	●	○	○
11. In the code designed to be reusable?	○	○	○	○	●	○
12. Are conversion and installation included in the design?	○	○	○	●	○	○
13. Is the system designed for multiple installations in different organizations?	○	○	○	○	●	○
14. Is the application designed to facilitate change and ease of use by the user?	○	○	○	○	○	●
Total 52.00						

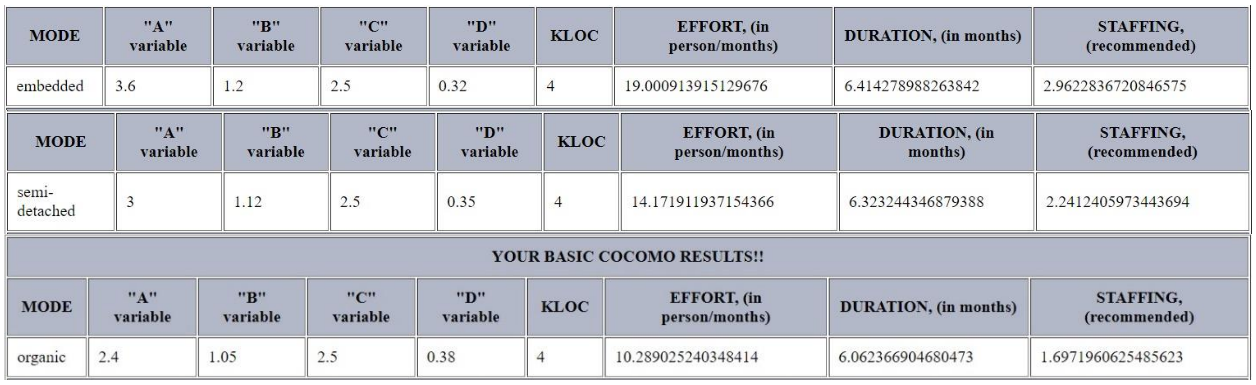
The Function Points is: 122.85

AFP = 122.85

SLOC = AFP * 30 = 3686LOC

COCOMO Model:

The Constructive Cost Model (COCOMO) predicts the effort and time required to complete a software project.



MODE	"A" variable	"B" variable	"C" variable	"D" variable	KLOC	EFFORT, (in person/months)	DURATION, (in months)	STAFFING, (recommended)
embedded	3.6	1.2	2.5	0.32	4	19.000913915129676	6.414278988263842	2.9622836720846575

MODE	"A" variable	"B" variable	"C" variable	"D" variable	KLOC	EFFORT, (in person/months)	DURATION, (in months)	STAFFING, (recommended)
semi-detached	3	1.12	2.5	0.35	4	14.171911937154366	6.323244346879388	2.2412405973443694

YOUR BASIC COCOMO RESULTS!!

MODE	"A" variable	"B" variable	"C" variable	"D" variable	KLOC	EFFORT, (in person/months)	DURATION, (in months)	STAFFING, (recommended)
organic	2.4	1.05	2.5	0.38	4	10.289025240348414	6.062366904680473	1.6971960625485623

Assuming an average monthly rate for a developer is $4000, here are the cost calculations for each mode:

Cost = Effort * Monthly Rate

Organic Mode: $41,160

Semi-Detached Mode: $56,680

Embedded Mode: $76,000

Project Planning

Work Breakdown Structure [WBS]:

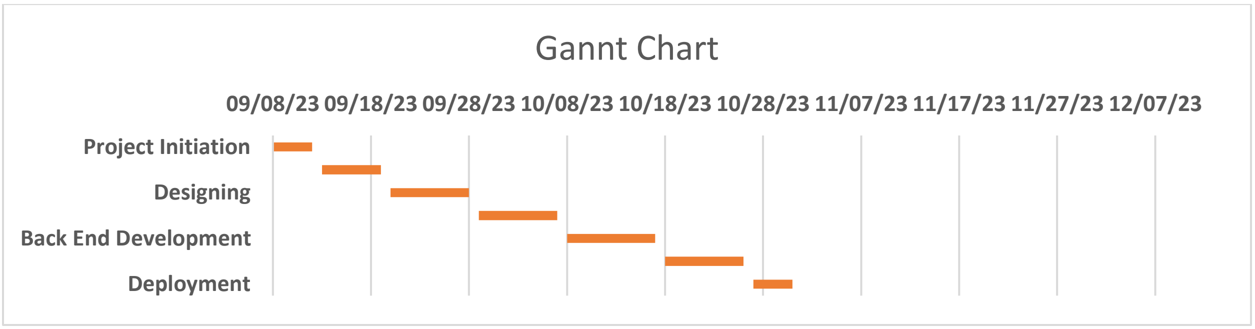
1. Project Initiation
    - 1.1 Define Project Objectives
    - 1.2 Identify Stakeholders
    - 1.3 Develop Project Charter
    - 1.4 Assemble Project Team
    - 1.5 Establish Project Plan
    - 1.6 Obtain Necessary Approvals
    - 1.7 Setup Project Environment
    - 1.8 Kickoff Meeting
2. Requirements Gathering
    - 2.1 Document Requirements
    - 2.2 Review and Approval

3. Designing
    - 3.1 System Architecture Design
    - 3.2 Database Design
    - 3.3 UI/UX Design
4. Front End Development
    - 4.1 HTML/CSS Development
    - 4.2 JavaScript Development
    - 4.3 User Interface Implementation
5. Back End Development
    - 5.1 Server-Side Programming
    - 5.2 Database Integration
    - 5.3 Backend API Development
6. Testing
    - 6.1 Unit Testing
    - 6.2 Integration Testing
    - 6.3 User Acceptance Testing
    - 6.4 Bug Fixes and Refinement
7. Deployment
    - 7.1 Deployment Planning
    - 7.2 Deployment Execution
    - 7.3 Post-Deployment Testing
    - 7.4 Transition to Operations

Project Schedule

Task	Start Date	End Date	Duration	Progress
Project Initiation	09/08/23	09/12/23	4	Completed
Requirements Gathering	09/13/23	09/19/23	6	Completed
Designing	09/20/23	09/28/23	8	Completed
Front End Development	09/29/23	10/07/23	8	Completed
Back End Development	10/08/23	10/17/23	9	In-progress
Testing	10/18/23	10/26/23	8	Pending
Deployment	10/27/23	10/31/23	4	Pending

Gantt Chart



Resource Allocation

Task	Resource	Hardware	Software
Project Initiation	Prudhvi	Laptop/PC, Internet	Project Management Software
Requirements Gathering	Preetham	Laptop/PC, Internet	Collaboration Tools
Designing	Prudhvi, Preetham	Laptop/PC, Design Software	IDE, Design Software
Front End Development	Prudhvi, Preetham	Laptop/PC, Development Servers	IDE, Web Development Tools
Back End Development	Prudhvi, Preetham	Laptop/PC, Development Servers	IDE, DBMS
Testing	Prudhvi, Preetham	Laptop/PC, Testing Servers	Testing Tools, IDE
Deployment	Prudhvi	Laptop/PC, Production Servers	Deployment Tools

Risk Management:

Risk Identification:

1. Technical Risks:

a. **Technology Stack Issues (Technical)**: Unforeseen technical issues or limitations in the chosen technology stack can impact development.

b. **Security Vulnerabilities (Technical)**: Data breaches or security vulnerabilities may compromise user data and trust.

2. Operational Risks:

a. **Low User Adoption (Organizational)**: If users find the platform difficult to use, adoption may be low, affecting the project's success.

b. **Scalability Issues (Technical)**: If the system cannot handle a growing number of tournaments and users, it can lead to performance problems.

c. **Data Loss and Backup (Technical)**: Data loss due to technical failures or other unforeseen circumstances can disrupt operations.

3. Economic Risks:

a. **Revenue Shortfall (External)**: Revenue may not meet projections due to low user adoption or economic downturns.

b. **Competitive Pressure (External)**: Increased competition in the tournament management software market may impact market share and profitability.

Risk Assessment:

For risk assessment, we'll use a matrix that considers likelihood and impact, ranging from Low (L), Medium (M), to High (H).

Risk Matrix:

Risk	**Likelihood**	**Impact**
Technology Stack Issues	M	M
Security Vulnerabilities	M	H
Low User Adoption	M	M
Scalability Issues	M	M
Data Loss and Backup	L	M
Revenue Shortfall	M	H
Competitive Pressure	M	M

Risk Mitigation Plan:

1. Technology Stack Issues:

- **Mitigation**: Maintain regular software updates, employ a skilled development team, and have contingency plans in place for technology limitations.

2. Security Vulnerabilities:

- **Mitigation**: Implement strong encryption, access controls, and conduct regular security audits.

3. Low User Adoption:

- **Mitigation**: Implement user-friendly design, offer training and support, and conduct marketing to increase awareness.

4. Scalability Issues:

- **Mitigation**: Plan for scalable architecture from the start, monitor system performance, and use cloud-based solutions for scalability.

5. Data Loss and Backup:

- **Mitigation**: Perform regular data backups and implement disaster recovery procedures.

6. Revenue Shortfall:

- **Mitigation**: Continuously assess revenue streams and be prepared to adjust the business model as needed.

7. Competitive Pressure:

- **Mitigation**: Stay updated with market trends, differentiate the platform with unique features, and offer competitive pricing.

Project Monitoring and Control

Monitoring Metrics:

- **Task Progress Tracking:** This KPI measures the progress of each task in the project plan. It involves tracking how much of the task is completed against the scheduled duration. By monitoring task progress, we can identify if any tasks are falling behind schedule and take corrective actions. This metric is crucial for assessing whether the project is on track and helps in making timely adjustments.
- **Defect Density:** Defect density measures the number of defects or issues identified in the project's deliverables. It helps ensure the quality of the work. A higher defect density may indicate problems with the development or testing process. By monitoring this KPI, we can catch quality issues early, reducing rework and ensuring that the project's deliverables meet quality standards

- **Resource Utilization:** This metric tracks how efficiently resources, including human resources and equipment, are being utilized. It can include monitoring the allocation and workload of team members, ensuring they are not overburdened, and that resources are used optimally. Effective resource utilization ensures the project stays within budget and on schedule.

Change Control Process:

- **Change Request Submission:** Team members or stakeholders submit change requests, detailing the proposed changes, reasons for the change, and potential impacts on the project, such as scope, schedule, and budget.
- **Change Request Evaluation:** The project manager and relevant stakeholders evaluate the change request. They assess its impact on the project, including schedule and budget implications, and determine if it aligns with the project's objectives.
- **Change Impact Analysis:** A thorough analysis is conducted to understand how the change will affect the project. This includes assessing the ripple effect on other tasks, the schedule, and resource requirements.
- **Change Approval:** Based on the evaluation and impact analysis, the project manager and a change control board (if applicable) decide whether to approve, reject, or defer the change request. If approved, the change is documented.
- **Implementation:** If approved, the change is implemented, and all relevant stakeholders are informed of the change. This may involve updating project documentation and informing the project team.
- **Continuous Monitoring:** The change's effects on the project are continuously monitored, including its impact on the project's original objectives, scope, schedule, and budget. Adjustments are made as necessary.

Quality Assurance:

- **Clear Quality Standards:** Define clear quality standards and acceptance criteria for project deliverables at the beginning of the project. These criteria should be measurable and aligned with project objectives.
- **Regular Testing and Review:** Implement rigorous testing processes, including unit testing, integration testing, and user acceptance testing. Conduct peer reviews and quality assurance checks at key milestones to identify and address defects promptly.
- **Quality Metrics:** Use quality metrics, such as defect density and compliance with predefined standards, to measure the quality of deliverables. Regularly assess whether the project is meeting these quality criteria.
- **Continuous Improvement:** Encourage a culture of continuous improvement, where project team members learn from past mistakes and strive for better quality in each project iteration.
- **Quality Audits:** Periodically conduct quality audits to ensure that processes and deliverables adhere to defined quality standards.

- **Documentation:** Maintain thorough documentation of quality assurance processes, testing results, and corrective actions taken. This documentation helps in demonstrating adherence to quality standards.

Design

Design Goals:

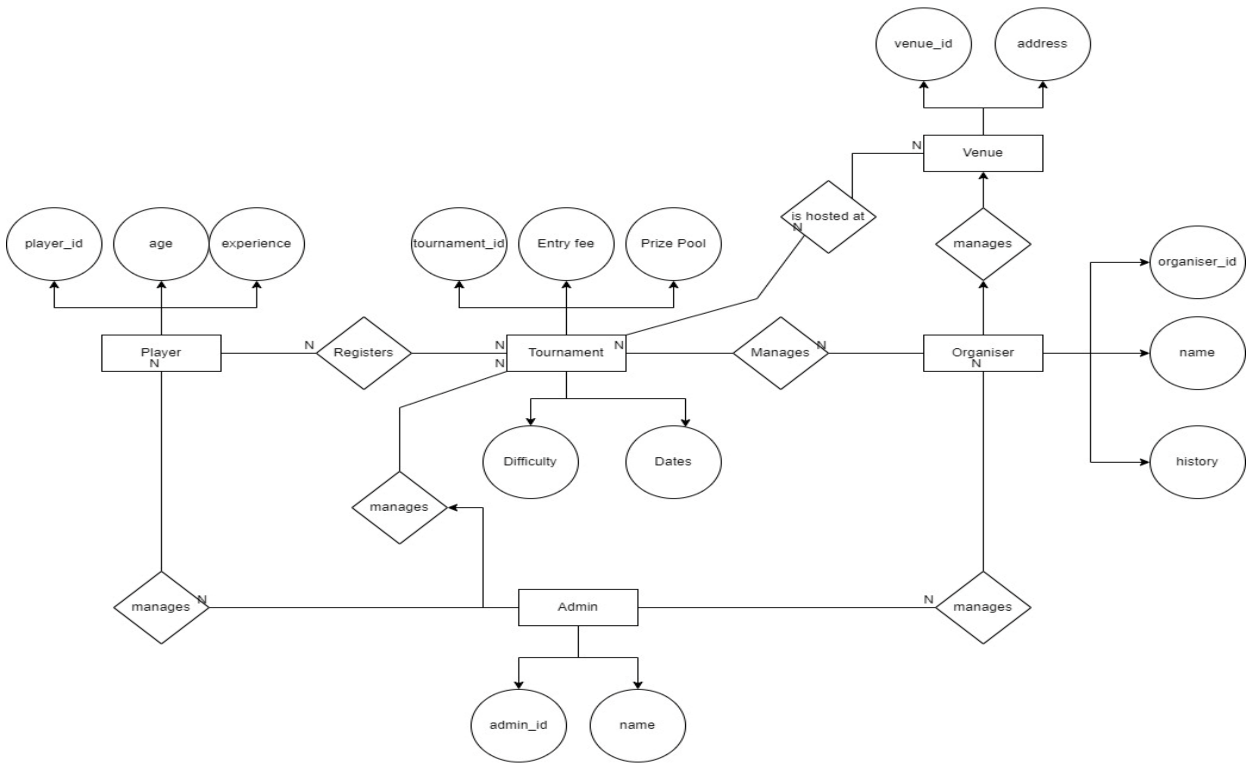
The design phase of a project is critical for shaping the future success of the application. The primary design goals for this project are:

- **User-Centered Design:** The design should prioritize the needs and preferences of the end-users. It should be intuitive, user-friendly, and provide an exceptional user experience.
- **Scalability and Maintainability:** The design should be scalable to accommodate potential future growth and changes. It should also be maintainable to allow for updates and enhancements with minimal disruption.

Design Approach:

- **User-Centered Design (UCD):** We will adopt a user-centered design approach, which involves actively involving end-users and stakeholders in the design process. We will conduct user interviews, surveys, and usability testing to gather feedback and insights. This iterative approach will help refine the design based on actual user needs and preferences.
- **Responsive Design:** We will ensure that the design is responsive, making the application accessible and user-friendly on various devices and screen sizes. This approach aligns with the goal of providing an excellent user experience.
- **Usability Guidelines:** We will follow established usability and design guidelines, such as those provided by Nielsen Norman Group and Material Design (for web and mobile applications), to ensure that the design is intuitive and consistent.
- **Information Architecture:** We will carefully structure the information and content within the application to ensure that it is organized logically and can be easily navigated by users.
- **Accessibility and Compliance:** We will design with accessibility in mind, adhering to WCAG (Web Content Accessibility Guidelines) standards to make the application accessible to a wide range of users, including those with disabilities.

ER Diagram



Implementation

Technology Stack:

- **Front-end Development**: Utilize modern web development technologies such as HTML, CSS, and JavaScript to create a responsive and user-friendly front-end.
- **Back-end Development**: Choose a robust and scalable technology stack for the server-side, which is Java.
- **Database**: We are going to use MySQL as storage.

Testing

The testing is going to be included in the final document.

Management tools

No management tools have been used other than gmail.

Remaining tasks:

The remaining tasks include Back-End Development, Testing, and Deployment.