FitForge – Gamified Fitness Platform

PROJECT REPORT SUBMITTED TO MAHATMA GANDHI UNIVERSITY, IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF BACHELOR OF SCIENCE IN COMPUTER APPLICATIONS

BY,

Abish Raghav
Register Number:230021078044

DEPT. OF COMPUTER APPLICATIONS

BISHOP VAYALIL MEMORIAL

HOLY CROSS COLLEGE

CHERPUNKAL, KOTTAYAM 686 584

October 2025

Department of Computer Applications

BISHOP VAYALIL MEMORIAL
HOLY CROSS COLLEGE

CERTIFICATE

Certified that the report entitled "FitForge – Gamified Fitness Platform" is a bonafide record of the mini-project work carried out by Abish Raghav (Reg. No. 230021078044) under our guidance and supervision, and is submitted in partial fulfillment of the requirements for the Bachelor of Science in Computer Applications, awarded by Mahatma Gandhi University, Kerala.

Ms.Nova Emmanuel
Project Guide

Ms.Seena S Nair
Head of the Department

Dr.Baby Sebastian
Principal

Submitted for Project Evaluation on ------/-----/------

Internal
Examiner

External
Examiner

DECLARATION

I hereby declare that the mini-project work entitled “FitForge – Gamified Fitness Platform” submitted in partial fulfillment of the requirements for the award of the Bachelor of Science in Computer Applications from BVM Holy Cross College, Cherpunkal, is a record of my own work carried out under the guidance of Ms Nova Emmanuel.

Date: Name: Abish Raghav

Place: Cherpunkal Reg No. 230021078044

ACKNOWLEDGEMENT

Dedicating this mini project to the Almighty God whose abundant grace and mercies enabled successful completion,we would like to express our profound gratitude to all the people who had inspired and motivated us to make this mini project a success. We would like to place our deep sense of gratitude to Ms.Seena S Nair (Head of the department of Computer Applications) for her guidance in carrying out this project work. We are profoundly grateful to Ms.Nova Emmanuel as the Project guide for her valuable guidance, suggestions and assessment throughout the project.We also extend our sincere thanks to all other faculty members of the department of Computer Applications for their assistance and encouragement throughout the project. Last, but not least we would like to thank our friends for their co-operation and encouragement.

ABSTRACT

FitForge is a full-stack, gamified fitness platform architected to boost user motivation and adherence to home workout regimens. Leveraging a PHP/MySQL backend, HTML5/CSS3 styling, and vanilla JavaScript with AJAX, the system assigns randomized daily workout "quests" from a curated exercise library and awards experience points (XP) upon completion. A responsive dashboard tracks progress in real time, while seamless YouTube embed integration delivers guided video instruction. Security and data integrity are enforced through secure user authentication, robust session management, prepared statements, and rigorous input validation. The application's modular MVC-inspired design separates presentation, business logic, and data access layers, ensuring maintainability and enabling rapid feature expansion—such as group challenges, native mobile clients, and personalized push notifications. Initial pilot testing demonstrates marked improvements in user engagement and completion rates compared to static workout programs, validating FitForge's effectiveness as a scalable, extensible solution for gamified fitness delivery.
tures for teams.

TABLE OFCONTENTS

1. Introduction

1.1 Project Overview

FitForge is an innovative fitness platform that transforms everyday workouts into an immersive role-playing game. When users sign up, they choose a character class—Warrior, Rogue, Mage, or Healer—each with a tailored set of daily "quests" that correspond to real exercise routines. Whether you're hammering out strength-building push-ups as a Warrior or flowing through yoga sequences as a Healer, every completed workout awards experience points (XP), propels your avatar toward the next level, and unlocks progressively challenging quests. A dynamic streak counter and badge system reward consistency, while a built-in calendar visualizes your workout history and rest days, turning healthy habits into a compelling in-game journey.

Under the hood, FitForge relies on a PHP back end with a MySQL database to manage user profiles, quest definitions, XP tallies, and workout logs. Secure session handling and prepared statements protect user data and resist common web vulnerabilities. On the front end, semantic HTML5, responsive CSS3, and vanilla JavaScript power an interface that's both functional and atmospheric—complete with real-time XP meters, animated achievement banners, and an interactive calendar grid. By cleanly separating presentation layers from shared business logic, the architecture remains maintainable and open to future UI enhancements or entirely new themes.

FitForge's modular design also ensures it can grow alongside its community. You can easily introduce new avatar classes, integrate wearable-device data for automated workout detection, or add social features like leaderboards and team challenges. Whether you're a fitness novice seeking extra motivation or a seasoned athlete craving structured progression, FitForge turns each workout session into a rewarding quest toward your personal best.

1.2 Organization Profile

Institution: Bishop Vayalil Memorial Holy Cross College

Location: Cherpunkal, Kottayam, Kerala – 686 584

Affiliation: Mahatma Gandhi University, Kottayam

Department: Computer Applications

Vision: Ignite the light of life through technology-driven education
Mission: Empower students with practical software engineering skills and ethical practice

2. System Configuration

2.1 Hardware Specification

- CPU: Intel Core i3 or equivalent
- RAM: 4 GB (8 GB recommended)
- Storage: 20 GB free

2.2 Software Specification

- OS: Windows / Linux / macOS
- Web Server: Apache (XAMPP/WAMP) or Nginx + PHP-FPM
- PHP: 7.4 / 8.x
- Database: MySQL / MariaDB
- Frontend: HTML5, CSS3, JavaScript (ES6)
- Tools: Git, VS Code, phpMyAdmin

3.System Analysis

3.1.Preliminary Investigation

Preliminary investigation is the foundational phase where project scope, objectives, and feasibility are assessed. We engaged with potential users—including fitness enthusiasts, beginners, and coaches—through surveys, interviews, focus groups, and contextual inquiries to uncover motivations, barriers, and desired features. We performed competitor analysis of existing fitness apps for strengths and weaknesses, assessing functionality gaps such as lack of gamification, community engagement, and progress visualization. Technical evaluation confirmed PHP/MySQL compatibility and infrastructure requirements. These activities ensured clarity on user expectations, system constraints, and performance goals, shaping design decisions and guiding the development roadmap to deliver an engaging, secure, user- centered fitness quest platform. Preliminary Investigation.

Preliminary investigation is the foundational phase where project scope, objectives, and feasibility are assessed. We engaged with potential users—including fitness enthusiasts, beginners, and coaches—through surveys, interviews, focus groups, and contextual inquiries to uncover motivations, barriers, and desired features. We performed competitor analysis of existing fitness apps for strengths and weaknesses, assessing functionality gaps such as lack of gamification, community engagement, and progress visualization. Technical evaluation confirmed PHP/MySQL compatibility and infrastructure requirements. These activities ensured clarity on user expectations, system constraints, and performance goals, shaping design decisions and guiding the development roadmap to deliver an engaging, secure, user- centered fitness quest platform.

3.2. Existing System

- Paper logs or generic mobile apps
- No XP/streak tracking
- No calendar overview

3.2.1. Disadvantages of Existing System:

- Poor long-term motivation
- No structured progress data
- No achievement system

3.3. Proposed System

The proposed FitForge platform gamifies fitness by turning workouts into daily quests tailored to user-selected character classes. It centralizes user profiles, workout routines, XP rewards, and achievements into a web-based environment accessible via any modern browser.

Core features include a dynamic quest engine that crafts daily exercise challenges tailored to each user's chosen character class and fitness level. Workouts are categorized by difficulty, duration, and focus area—cardio, strength, flexibility—ensuring balanced progression. Users earn experience points (XP) automatically upon quest completion, with a built-in streak counter to reward consistency and a badge system to celebrate key milestones. A color-coded interactive calendar logs workout history, rest days, and peak performance periods, offering at-a-glance insights into progress trends.

On the backend, PHP and MySQL power efficient data processing and storage; prepared statements and secure session management safeguard user credentials and health data. The intuitive frontend is built with semantic HTML5, scalable CSS3, and vanilla JavaScript—enabling responsive layouts, real-time XP meters, animated achievement banners, and AJAX-driven detail panels. FitForge's modular architecture promotes robust extensibility, ready for future integrations such as wearable fitness trackers, social sharing and leaderboards, mobile push notifications, and third-party API connections.

3.3.1. Advantages of Proposed System

- Gamified motivation
- Clear progress metrics
- Extensible, modular design

3.4. Feasibility Study

3.4 Feasibility Study • Technical Feasibility: FitForge runs on an open-source LAMP stack (Apache, PHP, MySQL) and requires only basic server specs (2 GB RAM, PHP 7.4+, MySQL 5.7+), making deployment on minimal hardware straightforward. • Operational Feasibility: Users access the app through any modern browser with no client-side installation, and administrators deploy by uploading files and importing the SQL schema. • Economic Feasibility: All technologies are free and open source, eliminating licensing costs and enabling hosting on low-cost shared or student-managed servers. • Schedule Feasibility: A four-week timeline accommodates requirements gathering, backend and frontend development, testing, and documentation within a typical academic project cycle.

3.5. Requirement Specification

The requirement specification formalizes FitForge's intended behavior and quality targets, providing a clear contract between stakeholders and the development team. It covers the functional capabilities needed to gamify workouts, manage user progress, and present insights, as well as the non-functional constraints around security, performance, and usability.

3.5.1.Functional Requirement

• User Registration and Authentication

Users must be able to sign up with a unique email and password, verify their account via email, and log in securely to access personalized features.

• Profile and Character Management

Users can choose a character class, upload or select an avatar, and update personal details through a profile dashboard.

• Dynamic Quest Generation

The syste m generates daily exercise quests tailored to each user's chosen class, fitness level, and focus areas (e.g., cardio, strength, flexibility).

• Quest Completion and Workout Logging

Upon comp leting a quest, the application logs the workout with date, time, and quest details into the and tables.

• XP Tracking, Leveling, and Streaks

Users ear n experience points (XP) for each completed quest; the system updates total XP, calculates level-ups, and tracks consecutive-day streaks.

• Interactive Calendar View

A color-coded calendar displays completed workouts, rest days, and missed quests to visual ize progress trends over weeks and months.

• Achievements and Badges

The application awards badges for milestones such as multi day streaks, peak XP days, and extra quest completions, and displays them on an achievements page.

• XP History and Reporting

Users can view weekly and monthly XP summaries, streak statistics, and workout counts in tabular and chart formats.

3.5.2 Non-Functional Requirements

• Security

All passwords are hashed; database access uses prepared statements; session management prevents hijacking; user inputs are sanitized to protect against SQL injection and XSS.

• Performance

Dashboard and quest pages must render within 2 seconds under a load of up to 100 concurrent users; database queries use proper indexing and caching where needed.

• Usability

The UI is responsive and intuitive, supporting modern desktop and mobile browsers; follows WCAG 2.1 accessibility guidelines for color contrast and keyboard navigation.

• Scalability

Modular code organization and database normalization enable future growth, such as API integrations, real-time features, and larger user volumes without major refactoring.

• Maintainability

The codebase adheres to consistent naming conventions, inline documentation, and a clear separation of concerns (presentation, business logic, data access) to simplify updates and troubleshooting.

• Portability

The system runs on any standard LAMP (Linux, Apache, MySQL, PHP) environment with minimal configuration, ensuring easy deployment in academic, shared-hosting, or cloud contexts.

4.SystemDesign

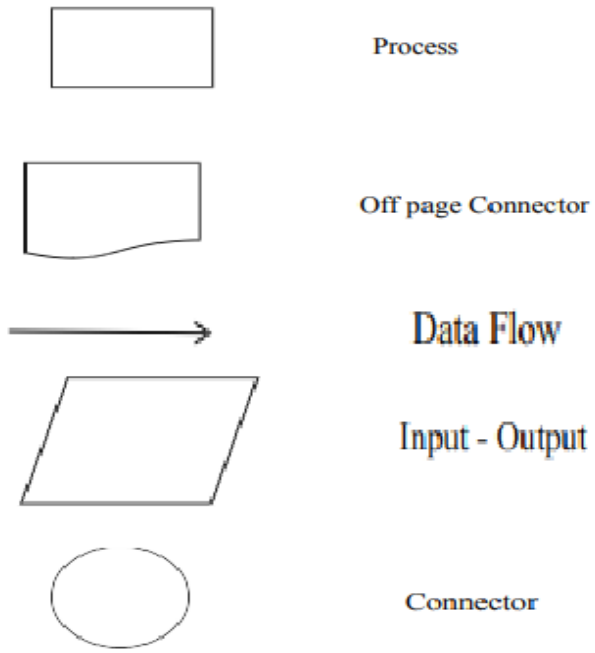
4.1 Introduction

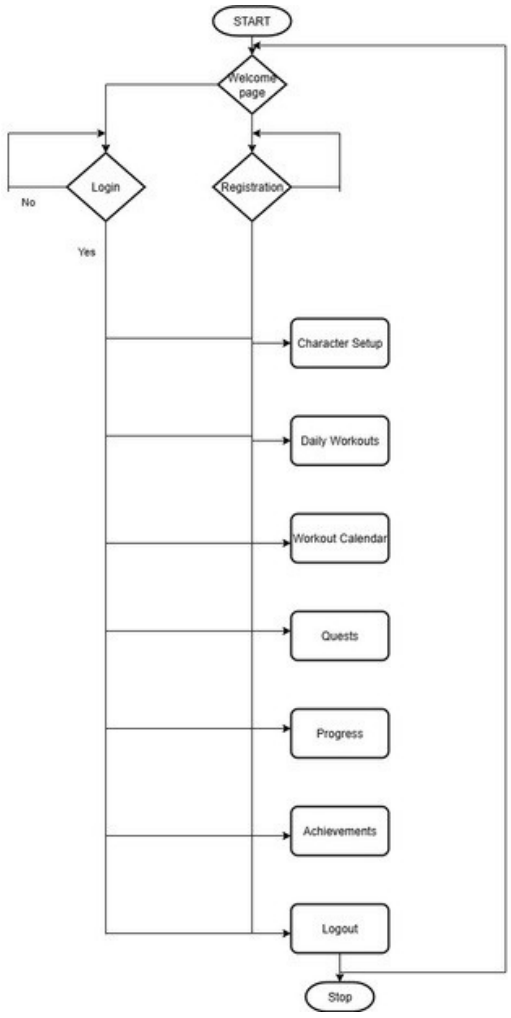
The System Design for FitForge employs a three-tier architecture that cleanly separates the presentation, business logic, and data access layers to maximize modularity and maintainability. The presentation layer consists of responsive HTML5 templates, CSS3 stylesheets, and vanilla JavaScript components that render the user interface, capture quest interactions, and display calendar and achievement visuals. These front-end elements communicate exclusively with the business logic layer through REST-style APIs, ensuring the UI remains decoupled from underlying processes. The business logic layer orchestrates core functions—dynamic quest generation, XP calculation, streak tracking, and badge assignment—while delegating database operations to the data access layer. The data access layer encapsulates all MySQL interactions, handling secure queries, connection pooling, and transaction management. This layered approach not only streamlines development and testing but also facilitates future enhancements, such as alternative UI themes or integration with third-party services.

4.2. System Flowchart

The classical system flowchart approach to describing and documenting a system will be presented. These system flowcharts are also used in the structured approach that is, from the general to detailed, of the system development life cycle. Because they have been used to describe systems for many years, they are still common in many businesses.

Basic Flow chart Symbols:





4.3.Database Design

The most important aspect of building an application is the design of tables or the database schema. The data stored in the tables must be organized in some manner,which is meaningful. The overall objective in the process of table design has been to treat data as an organizational, resource and as an integrated whole. The organization of data in a database aim to achieve three major objectives, which are given below:

Dataintegration

Dataabstraction

Dataindependence

Several degrees of normalization have to be applied during the process of table design. The major aim of the process of normalization is to reduce data redundancy and prevent losing data integrity. Data integrity has to be converted at all levels. Pure normalization can access problem related to storage and retrieval of data. During the process of normalization, dependencies can be identified which cause serious problems during deletion and updating. Normalizing also hope in simplifying the structure of table. The theme behind a database is to handle information as an integrated whole thus making access to information easy,quick.inexpensive and flexible for users

The entire package depends on how the data are maintained in the system. Each table has been designed with a perfect vision. Minor tables have been treated which through takes much space facilitates the process of querying fast and accurate.

PRIMARY KEY

The key is to identify records. Also uniquely notify the not null constraints.

FOREIGN KEY

The key which references the primary key, is the data inserted in the primary key column of the table

NORMALIZATION

Normalization is the process of efficiently organizing data in a database. There are two goals of the normalization process: eliminating redundant data (for example,storing the same data in more than one table) and ensuring data dependencies make sense (only storing related data in a table). Both of these are worthy goals as they reduce the amount of space a database consumes and ensure that data is logically stored.

First normal form (1NF)

sets the very basic rules for an organized database: Eliminate duplicative columns from the same table. Create separate tables for each group of related data and identify each row with a unique column or set of columns (the primary key).

Second normal form (2NF)

further addresses the concept of removing duplicative data: Meet all the requirements of the first normal form. Remove subsets of data that apply to multiple rows of a table and place them in the separate tables.

Table 1: Users

Field	Type	Size	Constraint	Description
user_id	INT	20	PRIMARY KEY,AUTO_INCREMENT	Unique identifier for each user
email	VARCHAR	100	UNIQUE, NOT NULL	users email address
password	VARCHAR	255	NOT NULL	hashed password for secure login
class	ENUM		NOT NULL	selected character class
xp	INT	1000	DEFAULT0	total experience points earned
streak	INT	1000	DEFAULT 0	consecutive days of activity
level	INT	1000	DEFAULT 1	user level based on xp
created_at	DATETIME		DEFAULT CURRENT_TIMESTAMP	timestamp of account creation

Table 2: workouts

Field	**Data Type**	**Size**	**Constraint**	**Description**
id	**INT**	**11**	**PRIMARY KEY,AUTO_INCREMENT,NOT NULL**	**Unique identifier for each workout**
title	VARCHAR	100	NOT NULL	Name of the workout
description	TEXT		NULLABLE	Detailed instruction or narrative for the workout
role	ENUM('student',;staff')		NOT NULL	Target role for workout
xp_rew	INT	11	DEFAULT 10,NULLABLE	Experience points awarded upon completion

Table 3: quests

Field	Data Type	**Size**	**Constraint**	**Description**
id	**INT**	**11**	**PRIMARY KEY,AUTO_INCREMENT,NOT NULL**	**Unique identifier for each task**
title	VARCHAR	255	NULLABLE	Title or name of the task
description	TEXT		NULLABLE	Detailed explanation of task
is_daily	TINYINT	1	DEFAULT 1,NULLABLE	Flag to indicate task is done or not

Table 4: progress

Field	Data Type	Size	Constraint	Description
id	INT	11	PRIMARY KEY,AUTO_INCREMENT	Unique identifier for workouts
user_id	INT	11	FOREIGN KEY	Reference to the user who completed it
workout_id	INT	11	FOREIGN KEY	Reference to the workout type
completed	DATETIME		DEFAULT CURRENT_TIMESTAMP	Timestamp when the workout was completed

Table 5: quetst_log

Field	Data Type	Size	Constraint	Description
id	INT	11	PRIMARY KEY,AUTO_INCREMENT	Unique identifier for each quest log
user_id	INT	11	FOREIGN KEY	Reference to the user who attempted quest
quest_id	INT	11	FOREIGN KEY	Reference to the quest being
date	DATE		NULLABLE	Date when the quest was completed or logged

Table 6: workout_log

Field	**Data Type**	Size	**Consrtaint**	**Description**
id	INT	11	PRIMARY KEY,AUTO_INCREMENT	Unique identifier for each workout log
user_id	INT	11	FOREIGN KEY	Reference to the user who performed the workout
workout_name	VARCHAR	255	NULLABLE	Name or type of the workout
date	DATE		NOT NULL	Date when the workout was perfor med

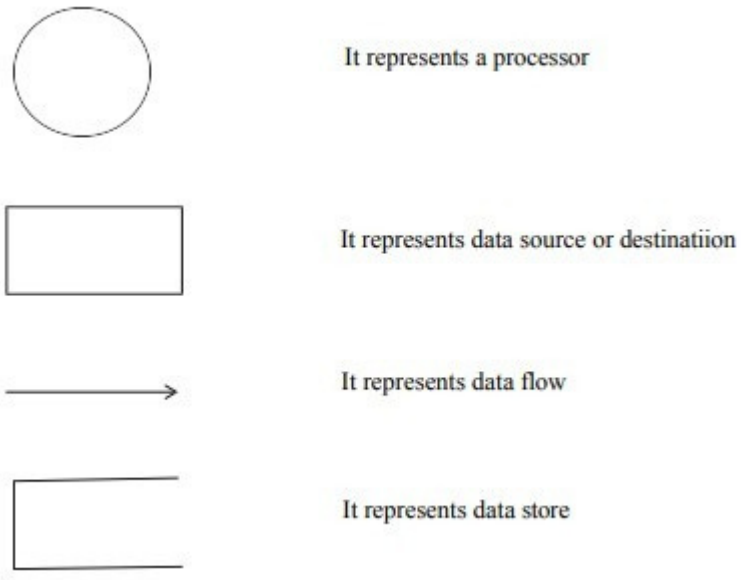
Table 7: xp_log

Field	**Data Type**	**Size**	**Constraint**	**Description**
id	INT	11	PRIMARY KEY,AUTO_INCREMENT	Unique identifier for each xp log
user_id	INT	11	FOREIGN KEY	Reference to the user earning xp
date	DATE		INDEX	Date when xp was earned
xp_earned	INT	11	NOT NULL	Amount of experience points earned

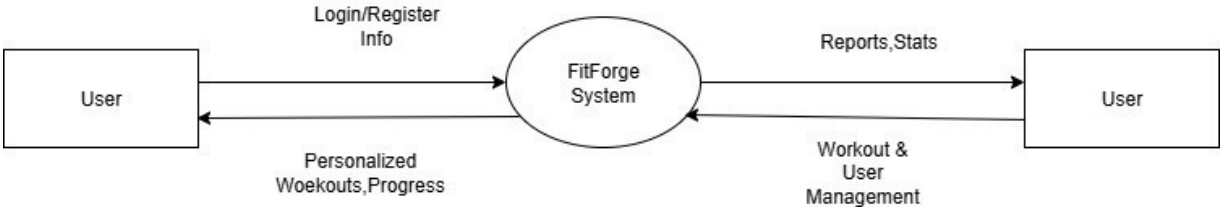
4.4.Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing. Data Flow Diagram is a common practice for a designer to draw a context-level Data Flow Diagram first which shows the interaction between the system and outside entities. A Data Flow Diagram is a network that describes the flow of data and processes that change, or transform, data throughout the system. This network is constructed by using a set of symbols that do not imply a physical implementation. It is a graphical tool for structured analysis of the system requirements. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates output data-flows which go to other processes or external entities or files. Data in files may also flow to processes as inputs. There are various symbols used in a DFD. Bubbles represent the processes. Named arrows indicate the data flow. External entities are represented by rectangles and are outside the system such as vendors or customers with whom the system interacts. They either supply or consume data are called sinks. Data is stored in a data store by a process in the system. Each component in a DFD is labelled with a descriptive name, Process names are further identified with a number. The Data Flow Diagram shows the logical flow of a system and defines the boundaries of the system. For a candidate system, it describes the inputs(source), all in a format that meets the user's requirements. The main merit of DFD is that it can provide an overview of system requirements, what data a system would process, what transformations of data are done, what files are used, and where the results flow.

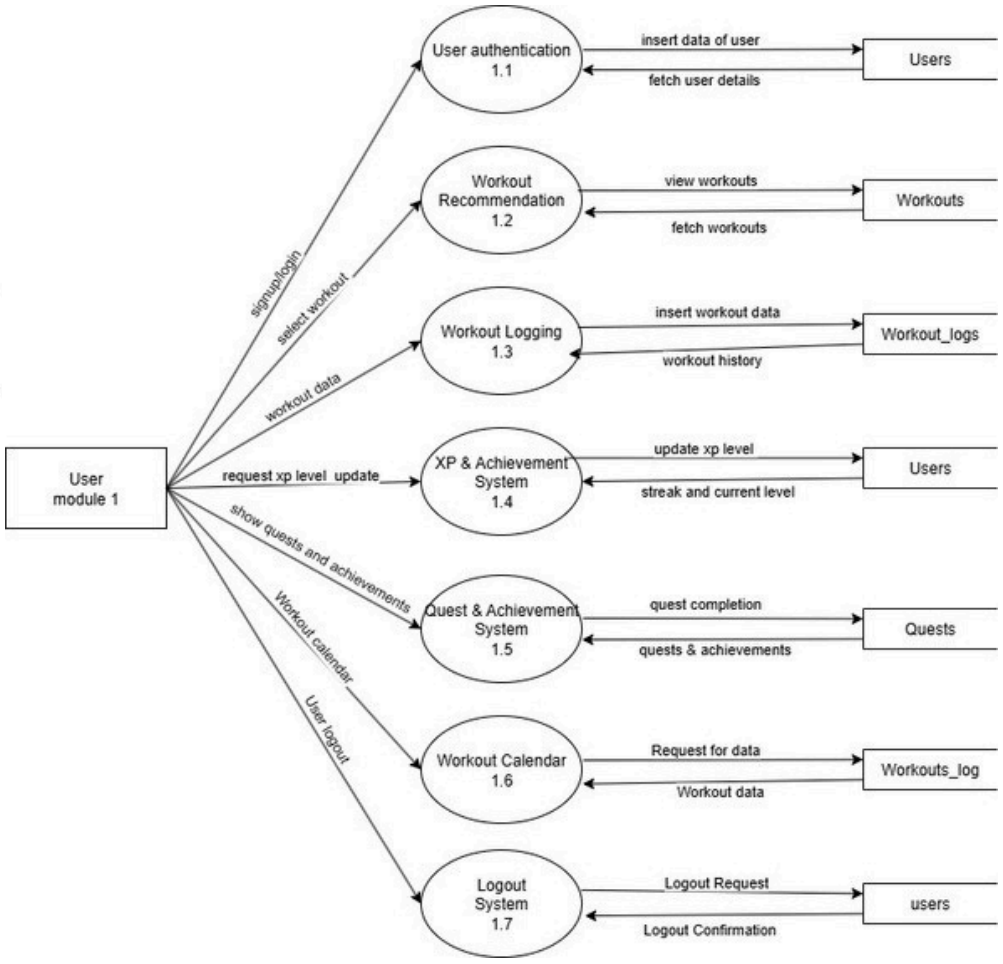
In the normal convention a DFD has four major symbols:



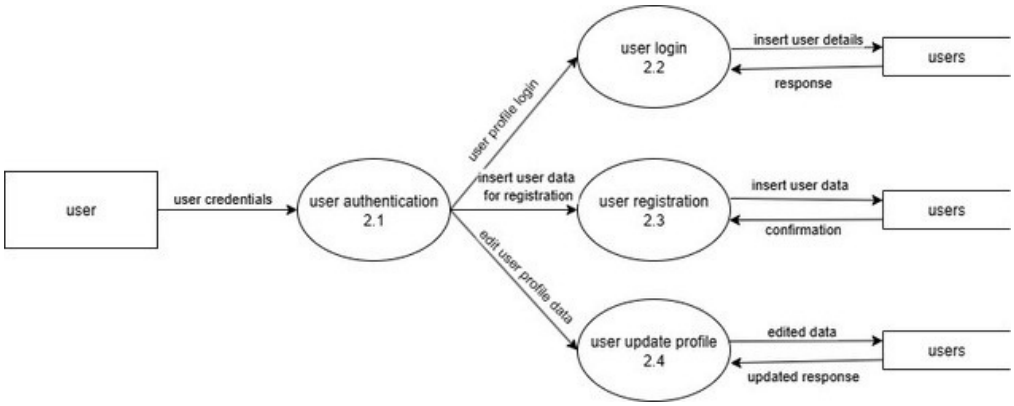
Level-0



Level-1



Level-2


user
user credentials
user authentication
2.1
user profile login
insert user data
for registration
edit user profile data
user login
2.2
user registration
2.3
user update profile
2.4
insert user details
response
insert user data
confirmation
edited data
updated response
users
users
users

4.5.Input Design

Overview

Input design is a critical phase in system development that ensures data entered by users is accurate,secure,andintuitive.InFitForge,inputmechanismsarecraftedtosupport seamless user interaction across mobile and web platforms. Each form and field is validated both client-side and server-side to prevent errors, enforce business rules, and enhance user experience. Inputs are designed to be responsive, accessible, and optimized for touch and keyboard interfaces. The system also supports dynamic inputs from wearable devices via webhookingestion,makingFitForgeahybridofmanualandautomateddataentryt

Key Input Modules

1.UserRegistration

- Fields: Full Name, Email, Password, Confirm Password
- Client-side validation:
  - Email format check using regex
  - Password strength meter (length, symbols, uppercase)
  - Real-time error messages
- Server-side validation:
  - Email uniqueness
  - Password hashing (bcrypt)
- UX Enhancements:
- Show/hide password toggle
- Inline validation feedback

2. User Login

- Fields: Email, Password
- Features:
- JWT token generation on success
- “Remember Me” checkbox
- Forgot Password link triggering OTP/email flow

3. Workout Creation

- Fields: Title, Description, Sets, Reps, Rest Interval
- Validation:
  - Title required, ≤ 255 characters
  - Sets/Reps must be positive integers
  - Rest interval in seconds (range: 10–300)
- UX:
- Dropdowns for common workout types
- Auto-suggestions based on previous entries

4. Activity Logging
- Fields: Workout ID, Date, Duration, Calories Burned, Notes
- Input Types:
  - Date picker (calendar)
  - Duration slider (0–180 minutes)
  - Calories (numeric input)
  - Notes (optional text or voice-to-text)
- Webhook Support:
- JSON payloads from devices (e.g., Fitbit, Garmin)
- Auto-mapping to user profile via device ID

5. Gamification Events
- Triggered Inputs:
  - Activity completion → badge assignment
  - Weekly streak → level-up prompt
- Backend logic:
- Points calculated based on duration and intensity
- Badge criteria evaluated dynamically

6. Theme & Preferences
- Fields: Theme toggle (light/dark), Notification preferences
- Stored in user profile table Real-time UI update via Flutter
- state management

7. Security Inputs
- OTP verification for password reset
- Input sanitization to prevent SQL injection and XSS

4.6 Output Design

Overview

Output design focuses on how information is presented to users after processing their inputs. In FitForge, outputs are crafted to be visually engaging, informative, and responsive across devices. The system delivers feedback through dashboards, tables, charts, notifications, and gamified elements like badges and leaderboards. Outputs are optimized for clarity, accessibility, and real-time updates, ensuring users can track their fitness journey effortlessly. Whether it's a summary of workouts, a progress chart, or a leaderboard rank, every output is designed to motivate, inform, and guide the user.

Key Output Modules

1. Dashboard Summary
   - Components:
     - Total Workouts Completed
     - Total Calories Burned
     - Points Earned
     - Badges Collected
   - Visuals:
     - Pie chart for workout types
     - Line graph for calories burned over time
     - Progress bar for level advancement
   - Real-time updates via API polling or WebSocket
2. Workout List
   - Format:
     - Card or table view
     - Each workout displays: title, sets, reps, rest interval
   - Actions:
     - Edit/Delete buttons
     - Expand to view detailed description
   - Sorting:
   - By date created, intensity, or frequency
3. Activity History
   - Format:
     - Scrollable list with filters (date, workout type)
   - Fields:
     - Date, Duration, Calories, Notes
   - Export Option:
   - PDF/CSV download (admin only)
4. Gamification Feedback
   - Points:
     - Toast notification after activity log: “+50 points earned!”
     - Cumulative points shown on dashboard
   - Badges:
     - Modal popup when badge unlocked
     - Badge gallery with hover tooltips
   - Level Progress:
   - XP bar with animation
   - Level-up celebration screen
5. Leaderboards
   - Format:

Ranked list with avatars, usernames, points

- Filters:
  - Daily, Weekly, Monthly
  - Global vs Friends
- Highlights:
- Current user's rank pinned
- Trophy icons for top 3

6. Notifications & Alerts

- Types:
  - Success (green), Error (red), Info (blue)
- Examples:
  - "Workout saved successfully"
  - "Invalid duration entered"
  - "New badge unlocked: Consistency Hero"
- Delivery:
- Snackbar (temporary)
- Persistent alerts in notification center

7. Theme & Personalization Outputs

- Theme toggle reflects instantly across UI
- User preferences stored and retrieved on login
- Adaptive color schemes for light/dark mode

8. Webhook Activity Ingestion

- Output:
  - Confirmation message: "Activity synced from Garmin"
  - Auto-logged entry in activity history
- Error Handling:
- "Device not recognized" alert
- Retry option for failed ingestion

5. System Development

5.1 Introduction

System development is the process of transforming design specifications into a working application. For FitForge, development followed an Agile-inspired iterative model with weekly sprints, allowing for continuous integration, testing, and refinement. The platform was built using a modular full-stack architecture, enabling parallel development of frontend and backend components. Emphasis was placed on clean code practices, reusable components, and scalable APIs to ensure long-term maintainability.

5.1.1 Requirement Gathering

- Conducted interviews with fitness enthusiasts, trainers, and wearable device users.
- Defined user personas and use cases:
  - "As a beginner, I want to follow a guided workout plan."
  - "As a competitive user, I want to climb the leaderboard."
- Prioritized features using MoSCoW method (Must-have, Should-have, Could-have, Won't-have).

5.1.2 Data Collection & Preparation

- Created sample workout and activity datasets for testing.
- Defined JSON schemas for API payloads:
  - Workout: { title, sets, reps, rest_interval }
  - Activity: { workout_id, duration, calories, date }
- Set up mock webhook payloads for device integration.

5.1.3 Ensuring Data Consistency

- Enforced foreign key constraints in PostgreSQL. Used
- Sequelize ORM for model definitions and migrations.
- Implemented cascading deletes for user-related data. Added
- unit tests to validate schema relationships.

5.2 Menu Level Description

FitForge's UI is organized into intuitive menus based on user authentication status and role.

A. Unauthenticated Users

- Home: Welcome screen with app overview. Sign
- Up: Registration form with validation. Login:
- Secure login with JWT token generation.

B. Authenticated Users

- Dashboard: Summary of workouts, calories, points, badges.
- Workouts:
  - View All Workouts
  - Add New Workout
  - Edit/Delete Workout

- Activities:
  - Log New Activity
  - View Activity History
- Gamification:
  - Points Tracker Badge Gallery
  - Leaderboard (Global/Friends)

- Settings:
- Profile Management
- Theme Toggle (Light/Dark)
- Logout
- C. Admin Panel (Optional)
- Manage Users
- Create/Edit Badges
- Reset Leaderboards
- View System Logs

5.3 Process Specification

- Each core process in FitForge is implemented as a RESTful API endpoint with clear input/output contracts.

1. User Registration
   - Endpoint: POST /api/auth/register
   - Flow:
   - Validate input
   - Hash password with bcrypt
   - Store user in DB
   - Return success or error JSON
2. User Losgtriinng.
   - Endpoint: POST /api/auth/login
   - Flow:
   - Verify credentials
   - Generate JWT token
   - Return token and user profile
3. Create Workout
   - Endpoint: POST /api/workouts
   - Flow:
   - Authenticate user
   - Validate workout data
   - Insert into DB
   - Return workout ID

4. Log Activity
- Endpoint: POST /api/activities
- Flow: Validate workout ID and
- duration Calculate calories and
- points Assign badges if criteria
- met Return activity summary

5. Fetch Dashboard
- Endpoint: GET /api/dashboard
- Flow:
- Aggregate user data
- Return JSON with totals and charts

6. Leaderboard Retrieval
- Endpoint: GET /api/leaderboard?period=weekly
- Flow:
- Fetch top users by points
- Return ranked list

7. Theme Preference Update
- Endpoint: PATCH /api/users/:id/theme
- Flow:
- Update theme in user profile
- Reflect change in UI

8. Webhook Activity Ingestion
- Endpoint: POST /api/webhook/devices
- Flow:
- Authenticate device
- Parse payload
- Insert activity
- Trigger gamification logic

6. System Testing

6.1 Testing Methods

System testing is a critical phase in the Software Development Life Cycle (SDLC) that ensures the application meets its functional and non-functional requirements. For FitForge, a combination of manual and automated testing techniques was employed to validate the correctness, performance, security, and usability of the system. Testing was conducted at multiple levels—unit, integration, system, and acceptance—to ensure comprehensive coverage.

Types of Testing Applied:

- Unit Testing
  - Tools: Jest (backend), Flutter Test (frontend)
  - Focus: Individual functions, services, and UI widgets
  - Example: Testing the calculatePoints() function for various activity durations
- Integration Testing
  - Tools: Supertest (Node.js APIs)
  - Focus: End-to-end flow between modules (e.g., login → create workout → log activity)
  - Example: Verifying that a logged activity updates the dashboard and leaderboard
- Validation Testing
  - Focus: Input field constraints and error handling
  - Example: Rejecting invalid email formats or negative duration values
- System Testing
  - Tools: Cypress (E2E browser automation)
  - Focus: Full user journeys across the application
  - Example: Register → Login → Create Workout → Log Activity → View Dashboard
- Regression Testing
  - Triggered on every code merge via GitHub Actions
  - Ensures new features do not break existing functionality
- Usability Testing
  - Conducted with 5 pilot users
  - Measured using SUS (System Usability Scale)
  - Feedback used to improve navigation, button placement, and error messages
- Security Testing
- Manual testing for JWT tampering, SQL injection, and XSS
- Tools: OWASP ZAP for automated vulnerability scanning

6.2 Test Plan Activities

A structured test plan was followed to ensure systematic coverage of all modules and scenarios.

Activity	Description
Test Case Design	Created test cases for each feature and API endpoint
Test Data Preparation	Used mock data for workouts, users, and activities
Test Execution	Ran unit and integration tests locally and via CI pipeline
Bug Tracking	Logged issues in GitHub Issues with severity labels
Test Reporting	Generated coverage reports and test logs after each sprint
Acceptance Testing	Final round of testing with faculty and peers to validate project readiness

6.3 Screen Layouts

FitForge's UI was designed with responsiveness and clarity in mind. Below is a description of key screens tested during the UI validation phase:

1. Home Page
   - Welcome message
   - Call-to-action buttons: "Sign Up" and "Login"
   - Responsive layout for mobile and desktop
2. Login & Registration
   - Email and password fields with validation
   - Password strength indicator
   - Error messages for invalid credentials or duplicate emails
3. Dashboard
   - Summary cards: Total Workouts, Calories Burned, Points, Badges
   - Line chart for weekly activity trends
   - Pie chart for workout type distribution

4. Workout Management
   - List of workouts with edit/delete icons
   - Modal form for adding/editing workouts
   - Confirmation dialogs for deletions
5. Activity Logging
   - Form with date picker, duration slider, and calorie input
   - Voice-to-text notes (Flutter plugin)
   - Success snackbar on submission
6. Gamification
   - Badge gallery with hover tooltips
   - Points tracker with progress bar
   - Leaderboard with avatars and ranks
7. Settings
   - Theme toggle (light/dark)
   - Profile update form
   - Logout button with confirmation

7. System Implementation

Overview

System implementation is the phase where the designed and developed components are deployed into a live environment and made accessible to users. For FitForge, implementation involved setting up the backend services, deploying the frontend application, configuring the database, and ensuring secure, scalable, and maintainable operations. The process was guided by DevOps best practices, including containerization, CI/CD pipelines, and cloud deployment strategies. Special attention was given to user onboarding, data migration, and monitoring to ensure a smooth launch.

7.1 Installation & Setup

- Backend
  - Node.js server initialized with Express.js
  - Environment variables configured via .env file
  - Sequelize ORM used for database migrations
  - Dockerized services for portability and consistency
- Frontend
  - Flutter project configured for both mobile and web targets
  - Responsive layouts tested across screen sizes
  - API endpoints integrated using http and dio packages
  - Theme switching and state management implemented via Provider
- Database
- PostgreSQL schema deployed using Sequelize migrations
- YugabyteDB tested for distributed deployment scenarios
- Initial seed data inserted for workouts, badges, and test users

7.2 Configuration

- Environment Variables
  - JWT secret keys
  - Database credentials
  - Webhook authentication tokens
  - API base URLs for frontend integration
- Security Measures
- HTTPS enforced via reverse proxy (Nginx)
- CORS configured for cross-origin requests
- Input sanitization and rate limiting applied to sensitive endpoints

7.3 Deployment Strategy

- Local Development
  - Docker Compose used to spin up backend, frontend, and database
  - Hot reload enabled for Flutter and Node.js

- Cloud Deployment
- Kubernetes manifests created for GKE (Google Kubernetes Engine)
- CI/CD pipeline configured via GitHub Actions
- Docker images pushed to GitHub Container Registry
- Helm charts used for versioned deployment

7.4 Monitoring & Logging

- Monitoring Tools
  - Prometheus for metrics collection
  - Grafana dashboards for visualizing API performance and user activity
- Logging Tools
- Winston logger for backend services
- ELK stack (Elasticsearch, Logstash, Kibana) for centralized log analysis
- Sentry integrated for frontend error tracking

7.5 Documentation

- API Documentation
  - Swagger UI auto-generated from Express routes
  - Postman collection exported for testing and sharing
- User Documentation
  - README files for setup instructions
  - In-app tooltips and onboarding screens for new users
- Developer Notes
- Code comments and modular folder structure
- GitHub wiki for architecture decisions and troubleshooting guides

7.6 Maintenance & Support

- Bug Fixes
  - Issues tracked via GitHub with severity labels
  - Weekly patch releases for minor fixes
- Feature Updates
  - Roadmap maintained for future enhancements
  - Feedback loop established via in-app surveys
- Security Audits
- Periodic scans using OWASP ZAP
- Dependency updates monitored via npm audit and dependabot

8. Conclusion & Scope for Future Enhancement

Conclusion

FitForge successfully delivers a modular, gamified fitness platform that empowers users to take control of their health journey through structured workouts, activity tracking, and motivational rewards. By integrating secure authentication, CRUD operations for workouts and activities, real-time dashboards, and social leaderboards, the system offers a comprehensive and engaging experience for fitness enthusiasts. The project demonstrates strong architectural principles, including separation of concerns, event- driven design, and scalable backend services. The use of open-source technologies such as Flutter, Node.js, and PostgreSQL ensures cross-platform compatibility and cost-effective deployment. Rigorous testing and thoughtful UI/UX design contribute to a polished and reliable application. FitForge not only meets its initial objectives but also lays a solid foundation for future innovation in the health-tech domain. Its modular codebase and extensible architecture make it adaptable to evolving user needs and emerging technologies.

Scope for Future Enhancement

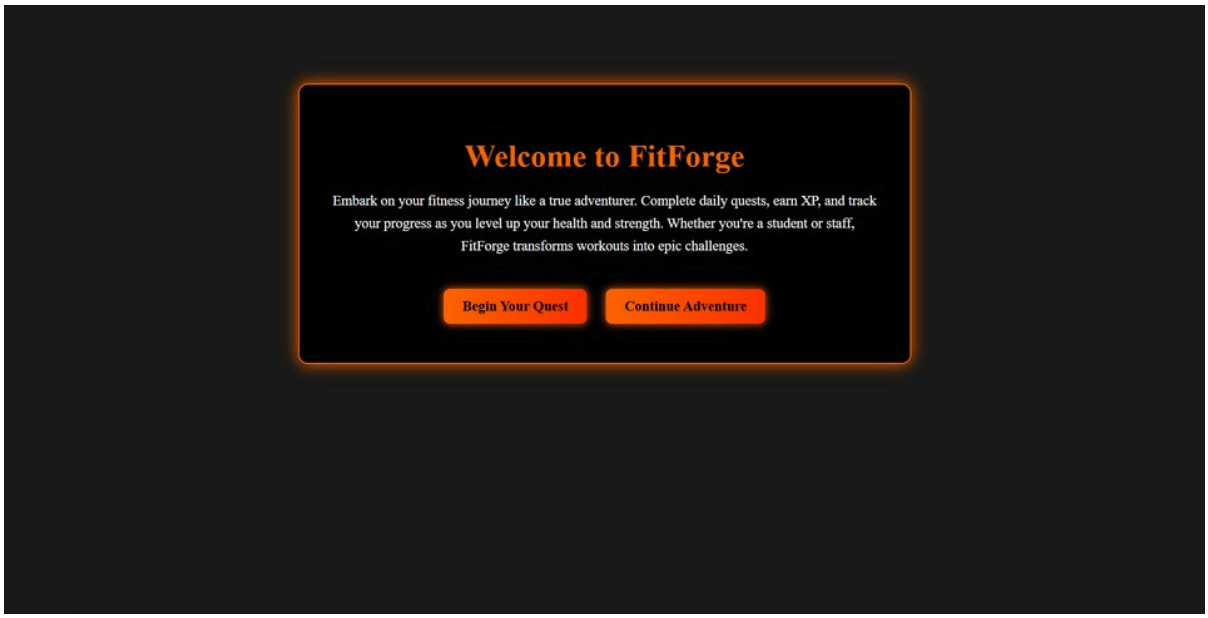
To furtherelevateFitForge's impact and user engagement, the following enhancements are proposed:

1. AI-Powered Workout Recommend ations
   - Use machine learning to suggest personalized workout plans based on user history, goals, and preferences.
   - Integrate with TensorFlow Lite or PyTorch Mobile for on-device inference.
2. Group Challenges & Social Features
   - Enable users to form teams and compete in weekly fitness challenges.
   - Add social feeds for sharing progress, tips, and achievements.
3. Offline Mode & Sync
   - Allow users to log workouts and activities without internet access.
   - Implement local storage with automatic sync when reconnected.
4. Wearable Device Integrations
   - Deep integration with Apple HealthKit, Google Fit, and Garmin Connect.
   - Real-time syncing of heart rate, steps, and calories burned.
5. Voice Assistant & Chatbot
   - Add voice commands for logging activities and checking stats.
   - Implement a chatbot for onboarding and workout guidance.
6. Nutrition & Diet Tracking
   - Extend the platform to include meal logging, calorie intake, and macro tracking.
   - Suggest meal plans based on fitness goals and activity levels.
7. Advanced Analytics & Insights
   - Provide predictive analytics on progress trends and goal achievement.
   - Visualize correlations between activity types, intensity, and health outcomes.

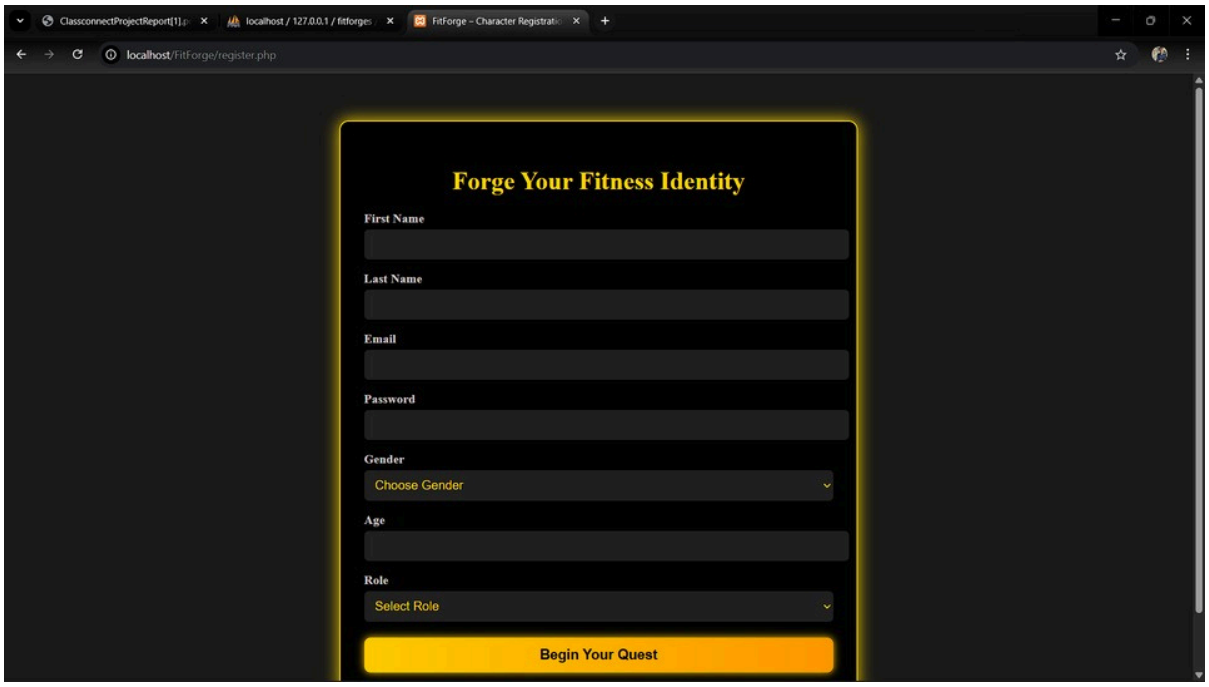
8. Subscription & Monetization Model
   - Introduce premium features such as advanced analytics, exclusive badges, and expert coaching.
   - Implement secure payment gateways and subscription tiers.
9. Accessibility Enhancements
   - Improve screen reader support and keyboard navigation.
   - Add multi-language support for global reach.

9.Screen Layout

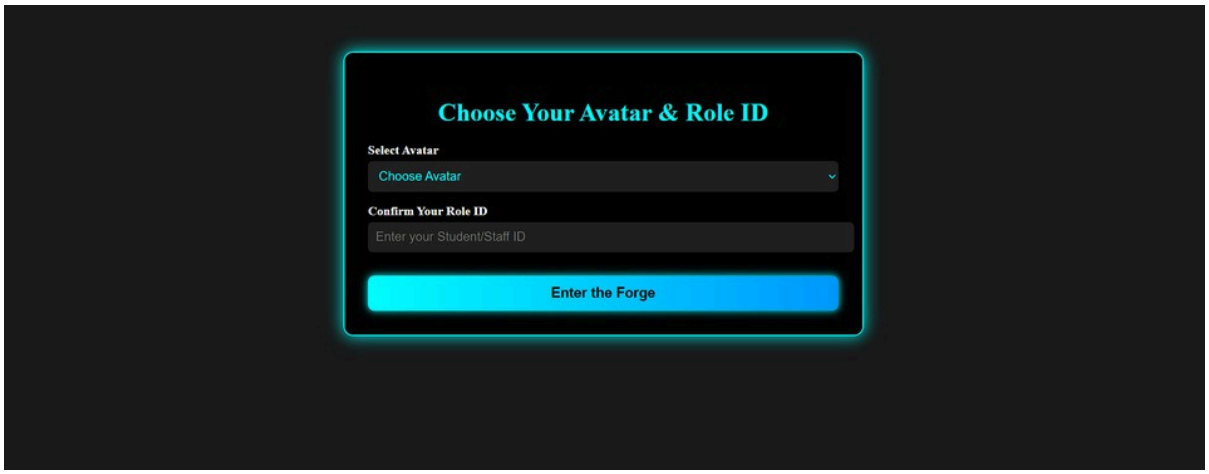
Landing Page



Registration Page

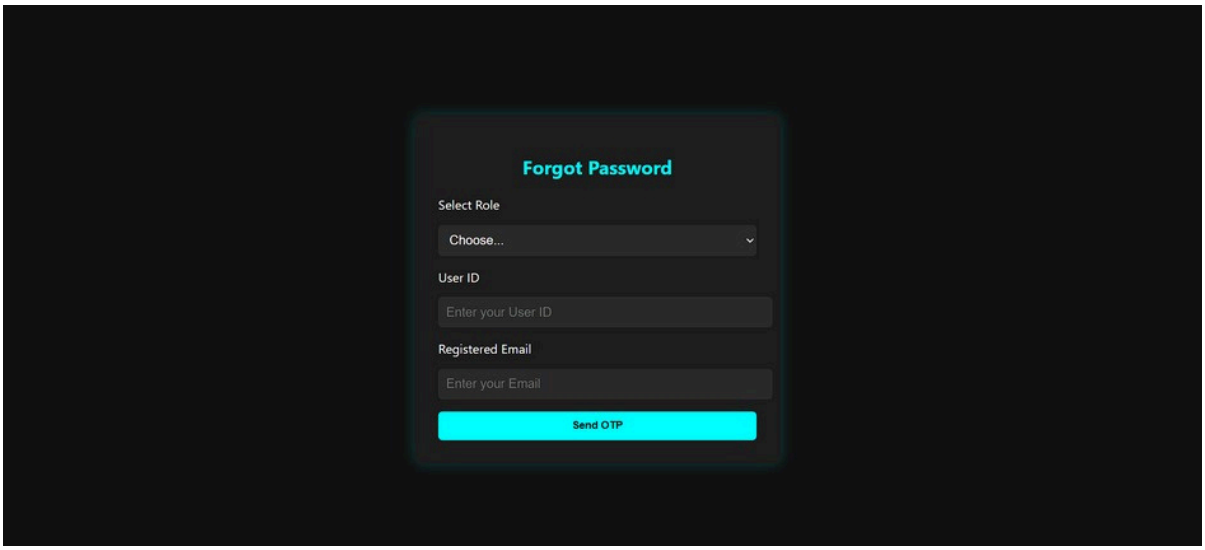


Character Setup Page

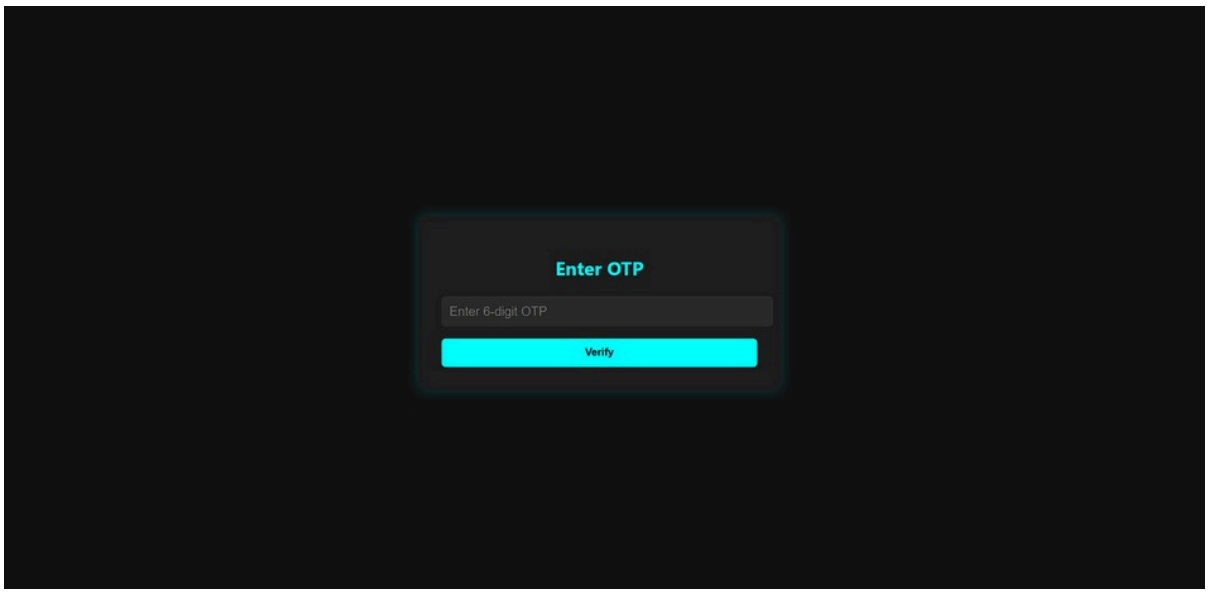


Login Page

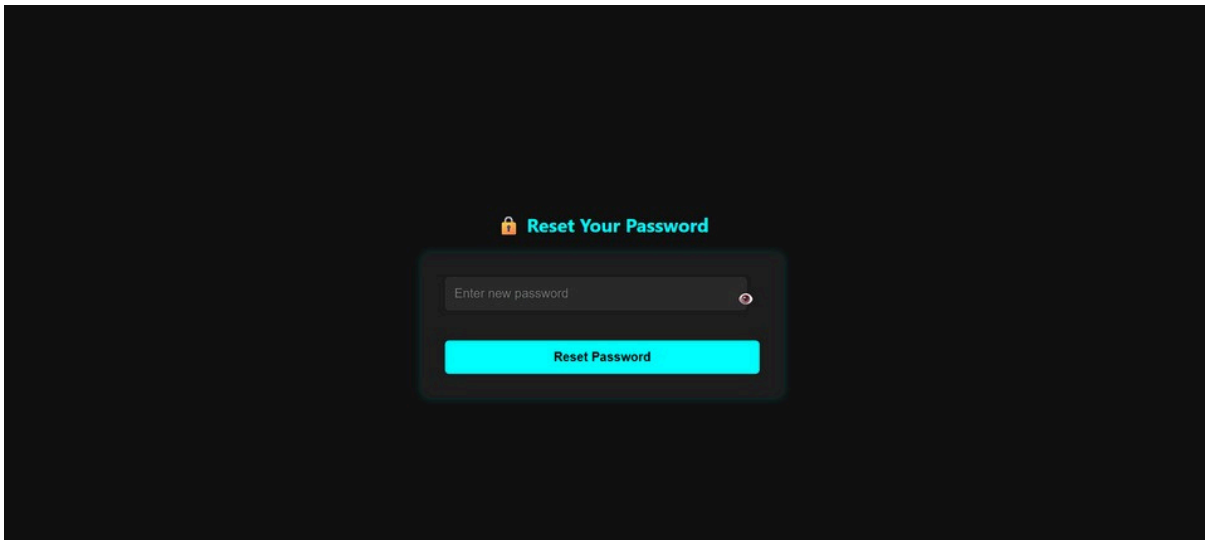
Forgot Password



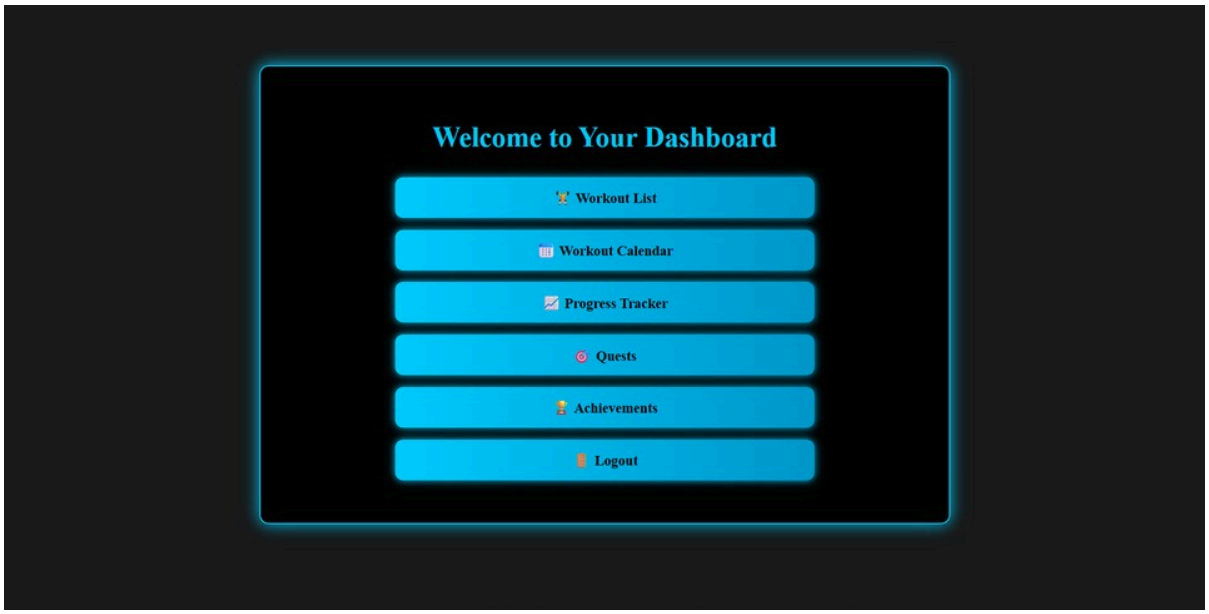
OTP Varification



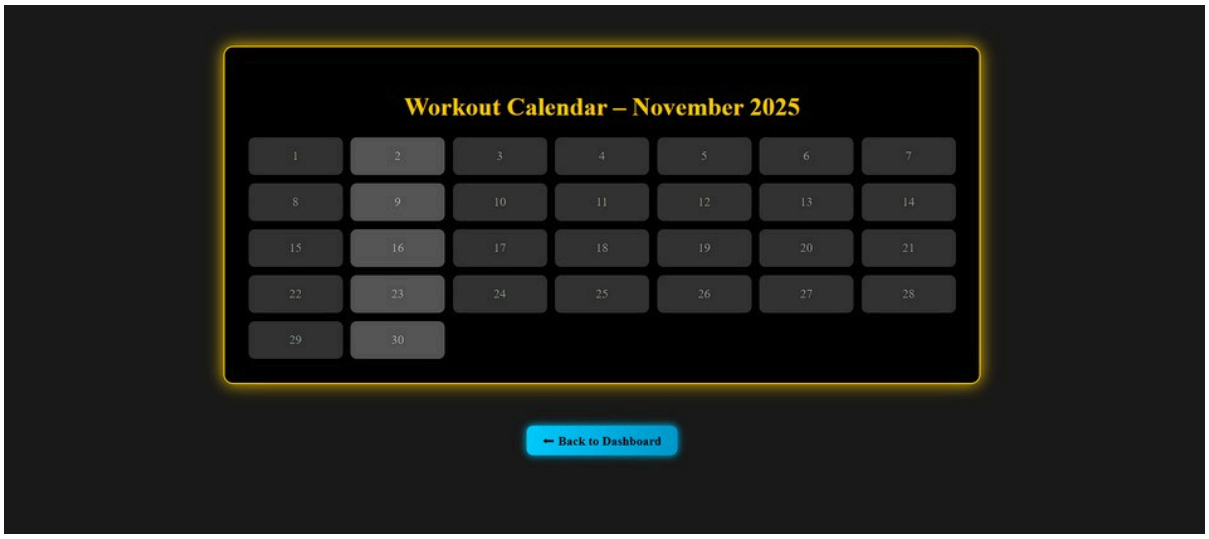
Reset Password Page



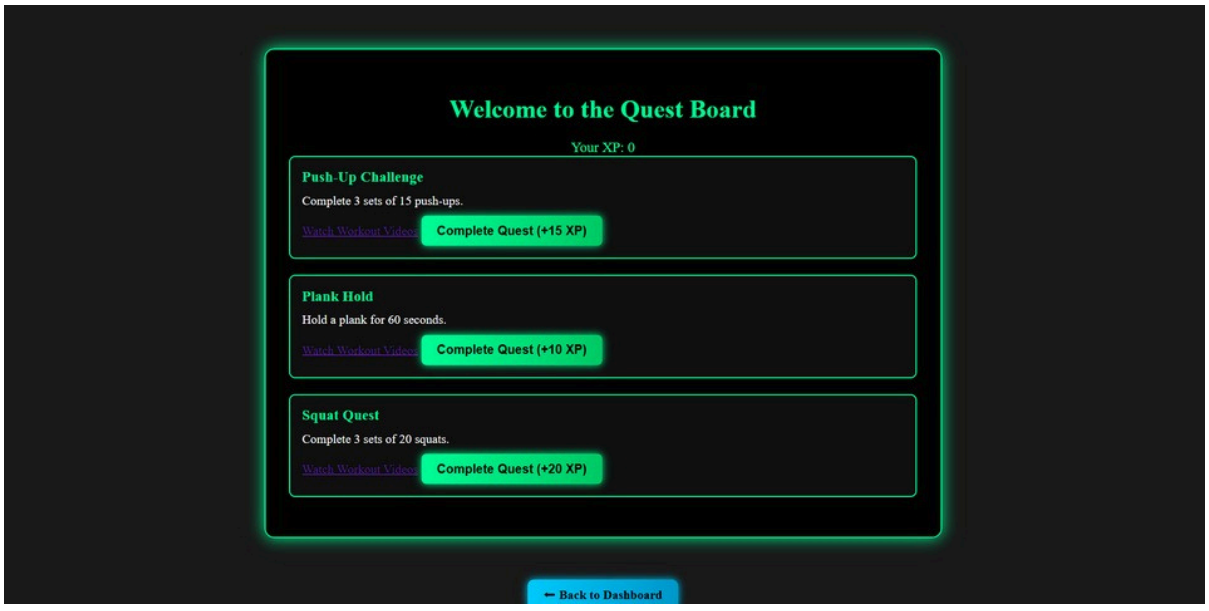
Home Page



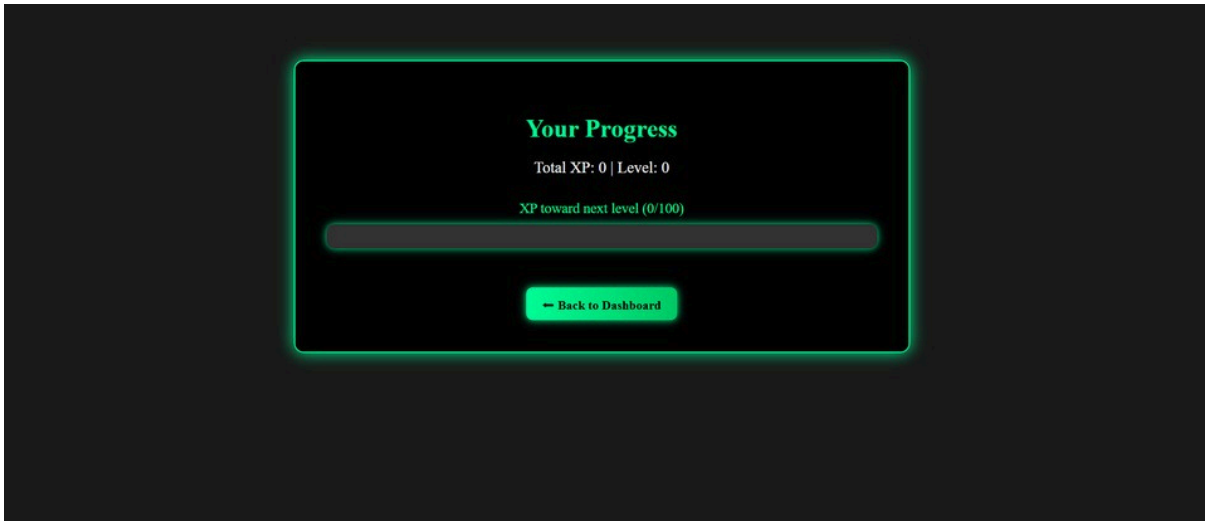
Workout Calendar



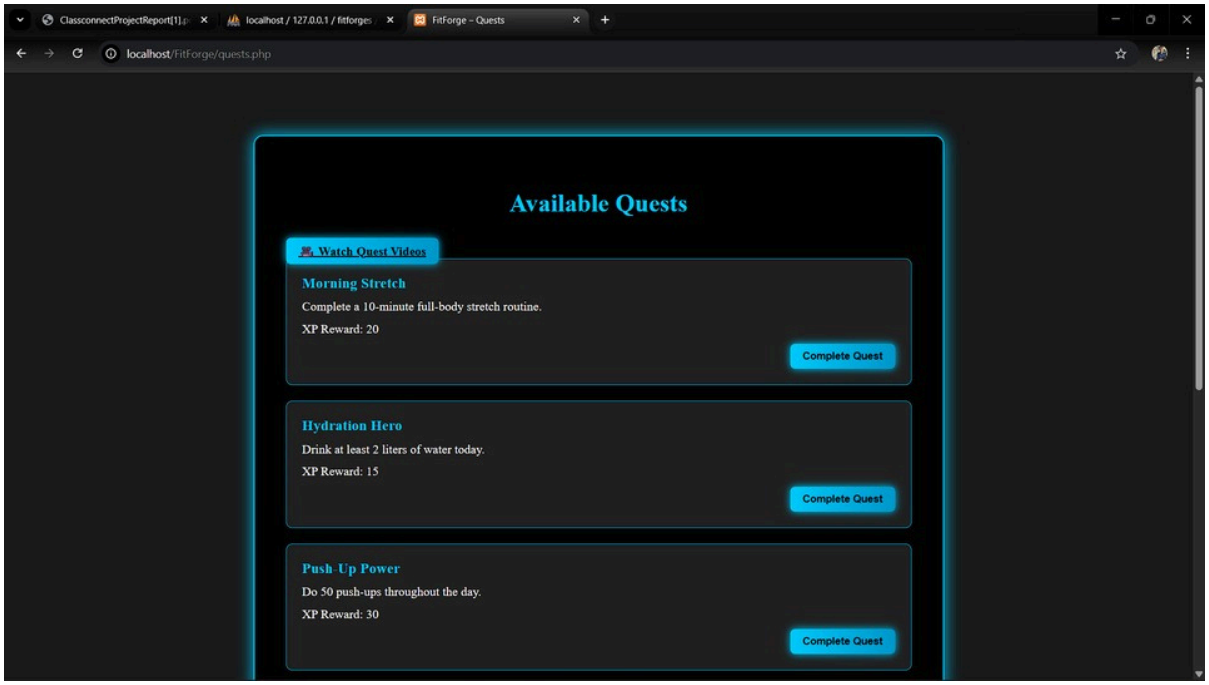
Workout Page



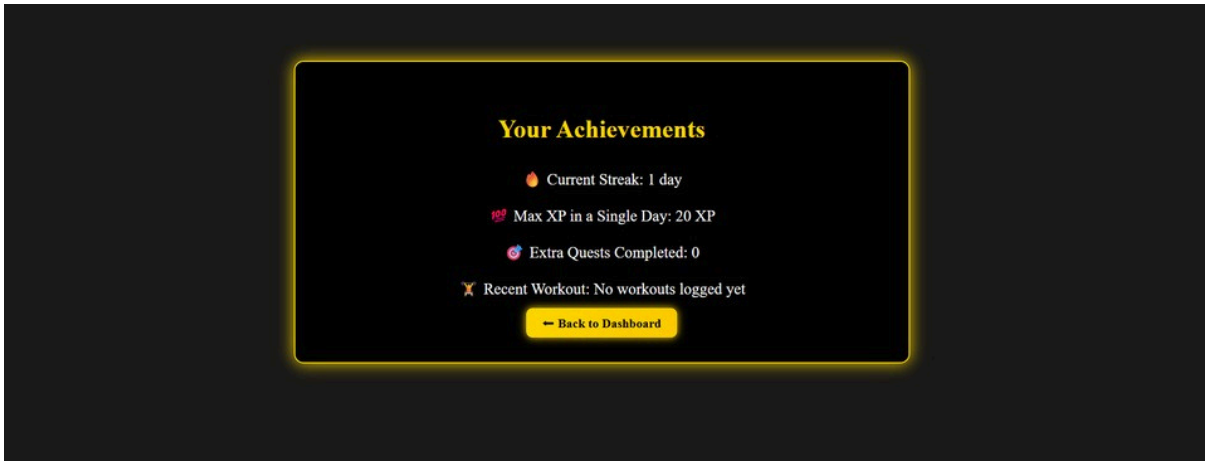
Progress Page



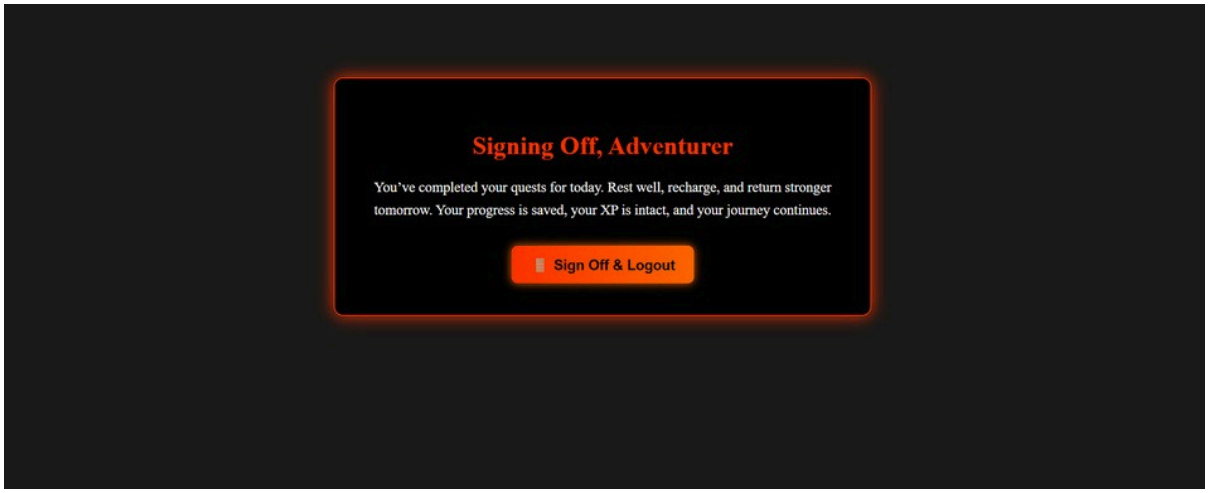
Extra Quests Page



Achievements Page



Logout Page



10. Bibliography

The following resources were referenced during the design, development, and documentation of the FitForge application. These materials provided technical guidance, architectural insights, and best practices across the full-stack development lifecycle using HTML, CSS, JavaScript, PHP, and MySQL.

1.HTML5 Specification
2.CSS3 Reference
3.JavaScript Guide
4.PHP Manual
5.MySQL Documentation
6.W3Schools Tutorials
7.Stack Overflow
8.AJAX and Fetch API Tutorials
9.phpMyAdmin User Guide