Team 2

2425-CT5179 Capstone Cloud Computing Group Project

Project: Virtual Fitness Community Application

Project Plan

Version: 1.2
Date 30.03.2025
Names: Conor Daly, Craig Quinn, James Sheridan

Version: 1.2
Date 30.03.2025
Names: Conor Daly, Craig Quinn, James Sheridan

Introduction

Maintaining an active and healthy lifestyle is essential, yet many individuals struggle to stay motivated and track their fitness progress effectively. While there are numerous fitness applications available, many are either focused solely on individual workout tracking or require expensive subscriptions to access premium features. Additionally, existing fitness platforms often lack strong community-driven engagement, making it difficult for users to connect, compete, and stay accountable.

The Virtual Fitness Community (VFC) aims to bridge this gap by providing a cloud-based, social-driven fitness platform where users can log workouts, participate in challenges, and track their progress over time. Unlike many existing solutions, VFC integrates community engagement, fitness tracking, and cloud automation, allowing users to set goals, compete in challenges, and stay motivated through a real-time leaderboard system.

Currently, users rely on fragmented solutions—some use dedicated tracking apps, while others turn to social media for accountability. However, there is no seamless, all-in-one platform that integrates social motivation, progress tracking, and DevOps-powered automation to enhance the fitness experience. VFC eliminates this fragmentation by combining workout logging, challenge-based fitness tracking, and real-time progress monitoring into a single, cloud-powered solution.

By leveraging modern cloud computing, automation, and CI/CD pipelines, VFC ensures scalability, real-time updates, and continuous deployment of new features—making fitness tracking more accessible, engaging, and interactive.

Project Scope

The scope of the Virtual Fitness Community (VFC) project is as follows:

1. **Develop a Cloud-Based Fitness Application**

- Create a web-based platform that allows users to register, log workouts, and participate in fitness challenges.
- Store user data securely in a cloud-hosted MySQL database.

2. **Facilitate Community-Driven Fitness Challenges**

- Implement a challenge system where users can create, join, and compete in structured fitness challenges.
- Develop a leaderboard system to track progress and rankings in real time.

3. **Develop a Back-End System for Data Management**

- Design and implement a Java-based backend to handle user authentication, workout logging, and leaderboard updates.
- Ensure efficient database management using MySQL and phpMyAdmin.

4. **Integrate RESTful APIs for Fitness Data Syncing**

- Connect with external APIs (e.g., Strava, Fitbit) to allow users to sync workout data from wearables.
- Process and store third-party fitness data in the cloud.

5. **Implement a Notification System**

- Develop a messaging service to send automated reminders for upcoming challenges, milestones, and leaderboard updates.
- Use email or push notifications to keep users engaged and informed.

6. **Develop a CI/CD Pipeline for Automated Deployment**

- Use Jenkins and Maven to implement a CI/CD pipeline, ensuring seamless integration, testing, and deployment of new features.
- Automate cloud deployment on AWS EC2, ensuring the platform remains scalable and up-to-date.

Objectives and Success Criteria

The primary objective of the Virtual Fitness Community (VFC) project is to develop a cloud-based fitness application that fosters user engagement, encourages physical activity, and leverages automation to enhance the overall fitness experience.

Objectives:

1. Develop a User-Friendly Cloud-Based Application

- Create a responsive web platform that allows users to log workouts, track progress, and engage in community challenges.
- Ensure a clean and intuitive UI/UX design for ease of use.

2. Enable Community-Driven Fitness Challenges

- Implement a challenge system where users can compete in predefined or custom challenges.
- Develop a leaderboard and ranking system for real-time performance tracking.

3. Implement Secure and Scalable Cloud Infrastructure

- Store user data securely using MySQL on a cloud-hosted database (AWS RDS).
- Utilize AWS EC2 instances for scalable hosting.

4. Automate Workflows and Deployment Using DevOps

- Integrate Jenkins and Maven to automate builds, testing, and deployments.
- Implement CI/CD pipelines to ensure seamless feature rollouts and updates.

5. Integrate API Connectivity for Fitness Data Syncing

- Enable users to sync fitness data from wearables and external fitness apps (e.g., Strava, Fitbit API).
- Process and store third-party workout data efficiently.

6. Develop a Notification and Engagement System

- Automate email and push notifications to remind users of challenge progress, leaderboard updates, and milestone achievements.

Success Criteria:

- The platform successfully allows users to register, log workouts, and join challenges with minimal friction.
- The challenge leaderboard updates in real time based on user progress.
- Users can sync fitness data from external apps, improving accuracy and convenience.
- Automated CI/CD pipelines successfully deploy new features without downtime.
- The system maintains scalability and reliability, handling increasing user demand without performance degradation.
- Users receive notifications about challenges and milestones, increasing their engagement with the app.

Research and References

To support the development of the Virtual Fitness Community (VFC) application, various sources have been reviewed to analyze existing solutions.

Competitor Analysis

- Strava – A leading fitness tracking and social platform for runners and cyclists. Provides leaderboards, challenges, and API integration with wearable devices.
  - Website: https://www.strava.com
- Fitbit App – A wearable-integrated fitness tracking platform that offers workout logging, health insights, and cloud-based data storage.
  - Website: https://www.fitbit.com
- MyFitnessPal – A fitness and nutrition tracking application that provides calorie logging, workout tracking, and social engagement features.
  - Website: https://www.myfitnesspal.com

Strava

First Launch: July 2009

General Overview:

Strava is a fitness tracking platform primarily designed for runners and cyclists. It enables users to record their activities via GPS, analyse performance metrics, and engage with a community of athletes. The platform offers both a website and mobile application for activity tracking and social interaction.

Core Features:

- Activity Tracking: Records running, cycling, swimming, and other sports activities using GPS devices or smartphones.
- Social Networking: Allows users to follow friends, share workouts, and comment on each other's activities.
- Challenges and Competitions: Hosts monthly and custom challenges to motivate users and foster competition.
- Leaderboards: Ranks users based on performance on specific routes or segments.
- Third-Party Integration: Supports synchronization with devices and platforms like Garmin, Fitbit, and Apple Health.

Fitbit

First Launched: 2007
General Overview:
Fitbit is a pioneer in wearable fitness technology, offering devices that monitor various health metrics. The Fitbit app complements these devices by providing detailed insights into physical activity, sleep patterns, and overall health.
Core Features:

- Activity Monitoring: Tracks daily steps, distance, calories burned, and active minutes.
- Heart Rate Monitoring: Provides continuous heart rate tracking to gauge exercise intensity and monitor health.
- Sleep Tracking: Analyses sleep duration and quality, offering insights to improve rest.
- Smartphone Integration: Delivers call, text, and app notifications directly to the wearable device.
- Community Engagement: Enables users to connect with friends, join groups, and participate in challenges.

MyFitnessPal

First Launched: 2005
General Overview:
MyFitnessPal is a comprehensive nutrition and fitness tracking application that helps users monitor their diet and exercise routines. It offers a vast food database and integrates with various fitness apps and devices to provide a holistic view of one's health.
Core Features:

- Calorie Counting: Allows users to log daily food intake and track calorie consumption.
- Exercise Logging: Enables tracking of workouts and physical activities, estimating calories burned.
- Macro and Micronutrient Tracking: Provides detailed insights into nutrient intake, including proteins, carbohydrates, fats, vitamins, and minerals.
- Goal Setting: Assists users in setting and monitoring weight loss, gain, or maintenance goals.
- Community Support: Offers forums and groups for users to share experiences, recipes, and motivation.

Each of the competitor applications analysed—Strava, Fitbit, and MyFitnessPal—offers many features, but they primarily focus on specific aspects of fitness tracking. Strava focuses on performance tracking and social competition, Fitbit focuses on health monitoring, and MyFitnessPal specializes in detailed nutrition tracking. However, users looking for a comprehensive, all-in-one fitness solution often must switch between multiple platforms to track their workouts, monitor their nutrition, and engage in community-driven fitness challenges.

- The Virtual Fitness Community (VFC) aims to combine the best aspects of all three platforms while introducing automation and cloud-powered scalability to enhance the user experience.

Technologies and Frameworks Used

Backend Development

- Programming Language: Java (Spring Boot Framework)
- Backend Framework: Spring Boot for REST API development, authentication, and business logic.
- Build Tool: Maven for dependency management, build automation, and packaging of the application.
- API Integration: RESTful API communication with external fitness APIs (e.g., Fitbit, Strava) for data syncing.

Frontend Development

- Languages: HTML, CSS, JavaScript
- Frontend Framework:

Database Management

- Database: MySQL (phpMyAdmin for management)
- Cloud Database Hosting:

Cloud Infrastructure & DevOps

- Hosting & Deployment: AWS EC2 instances for application hosting and scaling.
- CI/CD Pipeline: Jenkins + Maven for automated testing, building, and deployment of new features.
- Containerization: Docker for consistent deployment environments across development, staging, and production.
- Monitoring & Logging: AWS CloudWatch for tracking application performance and logging errors.

Security & Authentication

- User Authentication: JWT (JSON Web Token) authentication for secure user sessions.
- Role-Based Access Control (RBAC): To ensure different levels of user permissions (e.g., Admins vs. Regular Users).

Automation & Notifications

- Email & Push Notifications:

AWS Simple Email Service (SES) (Firebase Cloud Messaging (FCM) Possibly) for sending challenge updates and workout reminders.

The Proposed System:

Overview

Users will be able to register, log in, and access their personal dashboard to track their fitness journey. The platform will offer challenges, leaderboards, and API integration with third-party fitness applications such as Fitbit and Strava, allowing users to sync workout data seamlessly.
The backend system will include admin functionalities that allow designated personnel to manage user challenges, track platform engagement, and oversee system performance.
The platform will leverage cloud-based infrastructure for automated feature rollouts, scalable deployment, and efficient data storage.

System Components & Functionalities

User Authentication & Profiles

- Users will register and log in to the platform, with their credentials securely stored in a MySQL database.
- Role-based authentication will classify users as regular users or administrators with different permissions.

Workout Logging & Fitness Tracking

- Users can log workouts manually or sync data from wearable devices via APIs.
- The system will store workout history and progress trends in a cloud-hosted MySQL database.

Community Challenges & Leaderboards

- Users can create, join, and track fitness challenges (e.g., “Run 50km in a month”).
- The leaderboard system will dynamically update based on user activity and performance.
- Challenges will have statuses such as "Active," "Completed," or "Upcoming" based on progress.

Admin Dashboard & Management Panel

- Admins will be able to monitor user engagement, manage fitness challenges, and moderate forum discussions.
- Super-admins will have additional privileges, such as adding or removing challenge categories and managing API integrations.

API Integration & Data Syncing (Optional)

- The platform will support external API integration (e.g., Strava, Fitbit) to allow users to sync fitness data in real-time.
- This ensures a seamless experience for users who already use third-party fitness tracking devices.

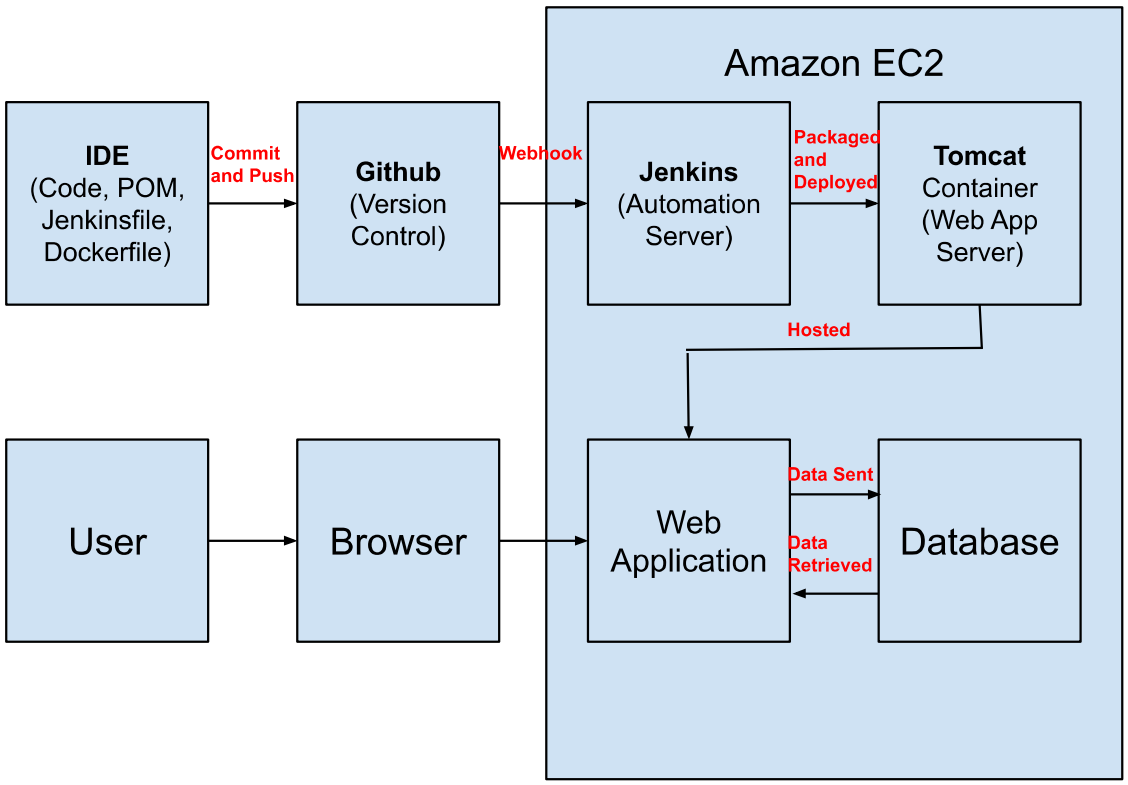
Automated Notifications & Messaging (Optional)

- A notification system will send alerts to users regarding challenge progress, workout reminders, and leaderboard updates.
- Users will receive messages such as "Challenge Started," "Milestone Reached," and "Final Day to Complete Your Challenge."

CI/CD Deployment & Cloud-Based Infrastructure

- The system will be hosted on AWS EC2, ensuring high availability and scalability.
- Using Jenkins and Maven, the CI/CD pipeline will allow automated deployments, testing, and feature updates with minimal downtime.
- Automated backups will ensure data security and recovery in case of failures.

Organisation Chart



Functional Requirements

Requirement 1:

- Users shall be able to register themselves / create profiles
- Users shall be able to log in to access their own information
- Users shall be able to log workouts
- Users shall be able to log meals/ calorie intake
- Users shall be able to delete all of their stored information

Requirement 2:

- The backend shall save user info
- The backend shall process user info
- The backend shall display metrics for the user
- The backend shall update the challenge leaderboards automatically
- User weekly/monthly challenges shall be updated automatically

Non-Functional Requirements

Non-Functional Requirement 1:

- Users can opt into weekly / monthly challenges
- Users can create their own challenges
- Notifications are sent automatically
- Users can opt into Notifications

Division of responsibilities

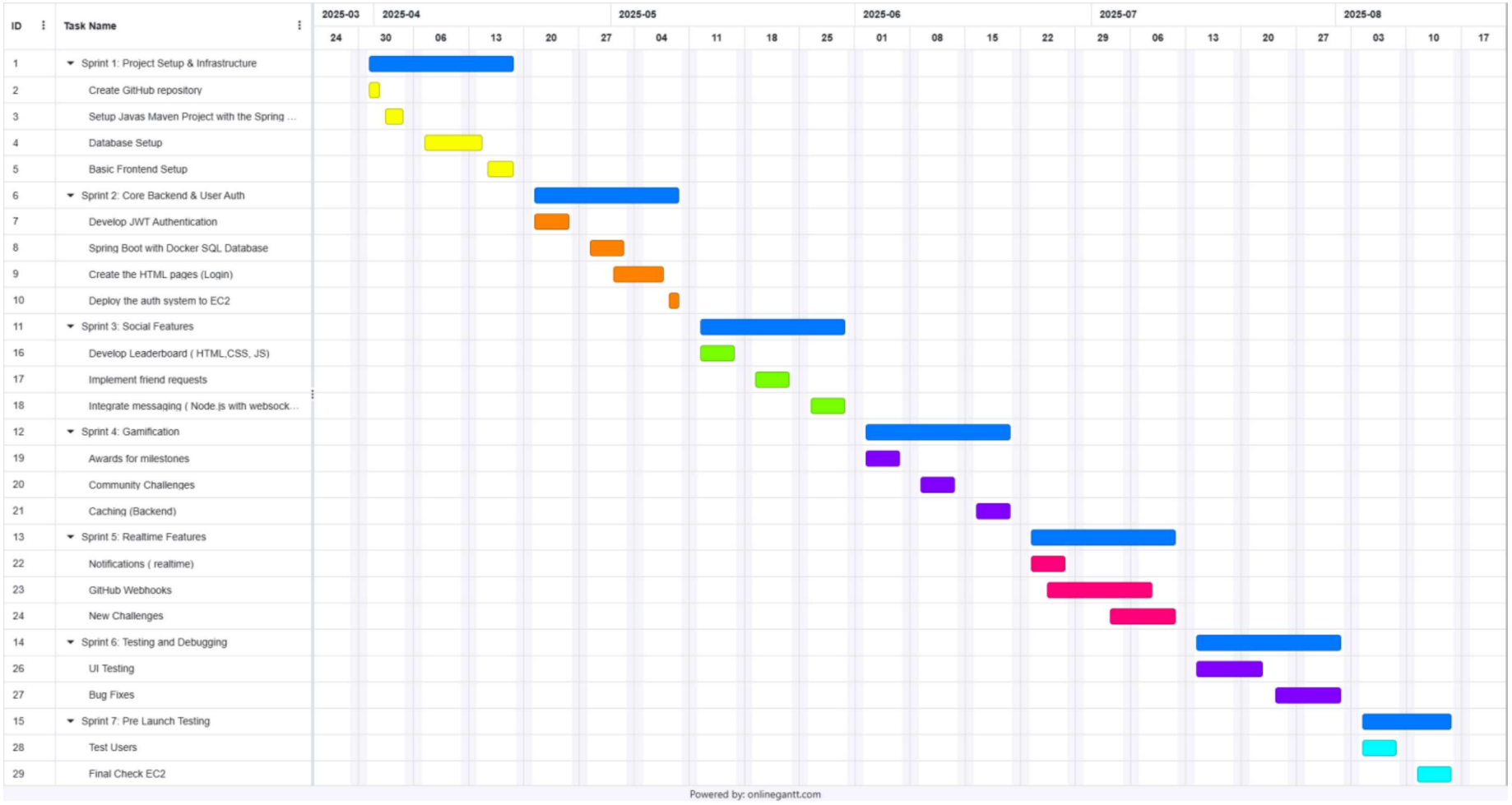
Trello will be used as a Team management system. The team will create tickets for pieces of work/features to be completed and they will be placed in the “Backlog” category. These tickets will be assigned to team members and then moved to the “In Progress” category. Once the assigned team member has completed their work the ticket is moved to the “Testing” category where a different team member will test that the feature has been fully completed and is ready to be deployed. Issues found during the “Testing” phase will push the ticket back to the “In Progress” phase. This system will allow us to track feature progress, share accountability between all team members, and track the overall progress of the project.

Project Schedule

1. Project Setup & Infrastructure
    - Create Github Repository
    - Setup Javas Maven Project with Springboot initialiser
    - Database Setup
    - Basic Frontend Setup
2. Core Backend & User Authorisation
    - Develop JWT Authentication
    - Spring Boot with Docker SQL Database
    - Create the HTML pages (Login)
    - Deploy the Authentication system
3. Social Features
    - Develop Leaderboard
    - Implement Friend Requests
    - Integrate Messaging
4. Gamification
    - Awards for Milestones
    - Community Challenges
    - Caching
5. Realtime Features
    - Notifications
    - Github Webhooks
    - New Challenges
6. CI/CD, Testing and Debugging
    - Jenkins
    - UI Testing
    - Bug Fixes
7. Pre-Launch Testing
    - Test Users
    - Final Check EC2

Gantt Chart

Below is the breakdown of the steps we plan on taking to complete this virtual fitness community application. It gives a brief overview of the different sections that we will be looking into and completing on a weekly basis.



Version: 1.2
Date 30.03.2025
Names: Conor Daly, Craig Quinn, James Sheridan

Technical References

To ensure the best practices in cloud computing, automation, and web application development, the following resources and frameworks have been referenced:

- Spring Boot Documentation: https://spring.io/projects/spring-boot
- Maven Build Automation: https://maven.apache.org/guides/
- Jenkins CI/CD Pipeline Guide: https://www.jenkins.io/doc/
- AWS EC2 & Cloud Deployment: https://aws.amazon.com/ec2/
- Docker & Containerization: https://docs.docker.com/get-started/
- Strava API Documentation: https://developers.strava.com/
- Fitbit API Guide: https://dev.fitbit.com/build/reference/web-api/