

**SIMATS ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**CHENNAI-602105**

**CAPSTONE PROJECT REPORT**

**Real-Time Ticket Booking System:**

**Agile Development with Kanban Framework**

**Submitted by**

**T GODWIN ABILASH**

**192321012**

**V PRANAV**

**192311029**

**CSA1086 – SOFTWARE ENGINEERING FOR PROGRAMMING**

**Table of Contents**

1. **Abstract**
2. **Project Definition and Scope**
   * Problem Statement
   * Goals
   * Proposed Technical Approach
3. **Requirements Engineering and Design**
   * Functional Requirements
   * Non-Functional Requirements
   * Design Architecture
4. **Development and Kanban Workflow**
   * Kanban Framework Implementation
   * Task Breakdown
5. **Testing, Quality Assurance, and Deployment**
   * Testing and QA
   * Deployment and Monitoring
6. **Challenges and Solutions**
   * Concurrency Issues
   * API Rate Limits
   * Responsiveness
   * Payment Security
7. **Ethics, UX, and Sustainability**
   * Ethics
   * User Experience (UX)
   * Sustainability
8. **User Interface**
9. **Conclusion**
10. **References**

**Real-Time Ticket Booking System: Agile Development with Kanban Framework**

**Abstract**

This capstone project aims to develop a real-time **Ticket Booking System** using an Agile methodology based on the **Kanban framework**. The software allows users to browse, book, and manage event tickets in real-time with features like secure payment processing and responsive design. By integrating live APIs for event data and payment gateways, the system ensures a seamless and user-friendly experience. Hosted on **AWS** and managed using **GitHub**, the project emphasizes continuous workflow improvement and incremental updates. The **Kanban framework** facilitates visual task management, seamless collaboration, and flexibility to address challenges such as managing high concurrency, ensuring secure transactions, and optimizing system responsiveness. This project demonstrates the effectiveness of Kanban in maintaining a streamlined, efficient, and user-focused development process.

**Project Phases**

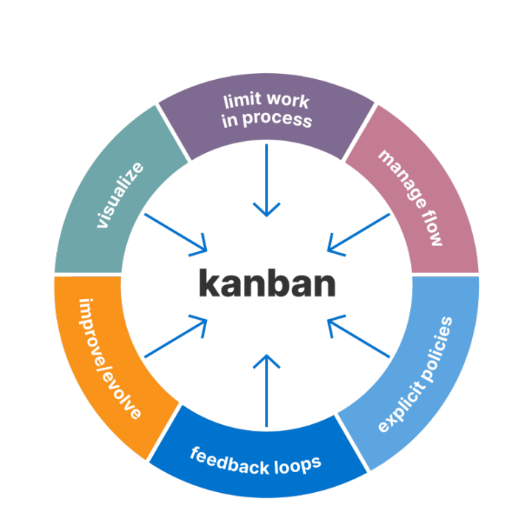
**1. Project Definition and Scope**

* + **Problem Statement**  
    Current ticket booking platforms often face issues like unresponsive interfaces, lack of real-time availability, and scalability during high demand. This project addresses these gaps by creating a platform that is responsive, secure, and scalable.
  + **Goals**
    1. Enable real-time ticket booking with live updates on availability.
    2. Provide a user-friendly interface for browsing and managing bookings.
    3. Implement secure payment systems with anti-fraud measures.
    4. Build a robust backend to handle concurrent user requests.
    5. Use a modular development approach for scalability and easy maintenance.
  + **Proposed Technical Approach**  
    The project adopts a Kanban-based Agile methodology, emphasizing iterative progress and task visualization. Technical tools and technologies include:
    1. **React.js** for a dynamic, responsive frontend.
    2. **Node.js** and **MongoDB** for scalable backend services.
    3. **Docker** for containerization and isolated environments.
    4. **Jenkins CI/CD pipelines** for automated testing and deployment.
    5. **AWS hosting** for reliability and scalability.

**2. Requirements Engineering and Design**

* + **Requirements**
    1. **Functional Requirements**
       - Real-time ticket availability and booking.
       - User authentication and profile management.
       - Search and filter functionality for events.
       - Secure payment processing with APIs like Stripe or PayPal.
    2. **Non-Functional Requirements**
       - Scalability to handle thousands of concurrent users.
       - High responsiveness across devices.
       - Fault tolerance and high availability.
  + **Design Architecture**  
    The architecture emphasizes modularity and scalability:
    1. **Frontend**: React.js with Material-UI for responsive design.
    2. **Backend**: Node.js with Express.js for API management and MongoDB for data storage.
    3. **Integration**:
       - API for fetching and managing ticket data.
       - Payment API integration for secure transactions.
    4. **Deployment**: Hosted on AWS with Docker and Kubernetes for orchestration.

**3. Development and Kanban Workflow**

* + **Kanban Framework Implementation**  
    The Kanban methodology ensures efficient task management and collaboration:
    1. **Kanban Board**:
       - **To Do**: Tasks ready for initiation.
       - **In Progress**: Tasks under active development (e.g., API integration, responsive UI design).
       - **Code Review**: Tasks completed and undergoing peer review.
       - **Done**: Verified and deployed features.
    2. **Work-in-Progress (WIP) Limits**: Prevent bottlenecks by limiting the number of active tasks per developer.
    3. **Incremental Updates**: Features are developed and tested in small increments to ensure steady progress.
  + **Task Breakdown**
    1. **Frontend Development**:
       - Implement responsive layouts and navigation menus.
       - Add event filters and search functionality.
    2. **Backend Development**:
       - Develop APIs for ticket availability and booking.
       - Integrate payment gateways and ensure data security.
    3. **API Integration**:
       - Fetch live event data using external APIs.
       - Handle edge cases like rate limits through caching mechanisms.

**4. Testing, Quality Assurance, and Deployment**

* + **Testing and QA**
    1. Unit testing using **Jest** for frontend and backend modules.
    2. API testing using **Postman** to ensure secure and reliable endpoints.
    3. End-to-end testing with **Cypress** to simulate user journeys, including booking and payment flows.
  + **Deployment and Monitoring**
    1. Hosted on **AWS** with a containerized environment managed by **Kubernetes**.
    2. Monitoring and logging implemented using **ELK Stack** (Elasticsearch, Logstash, Kibana).
    3. Real-time error tracking with **Sentry**.

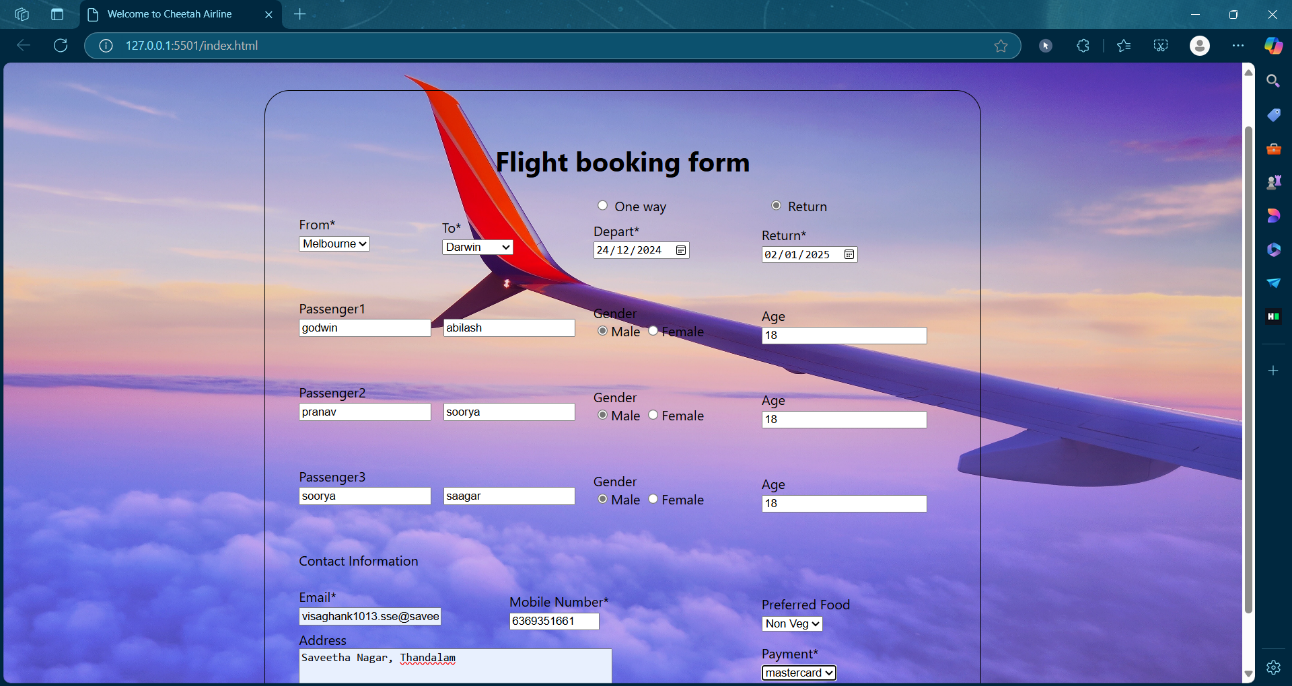
**5. Challenges and Solutions**

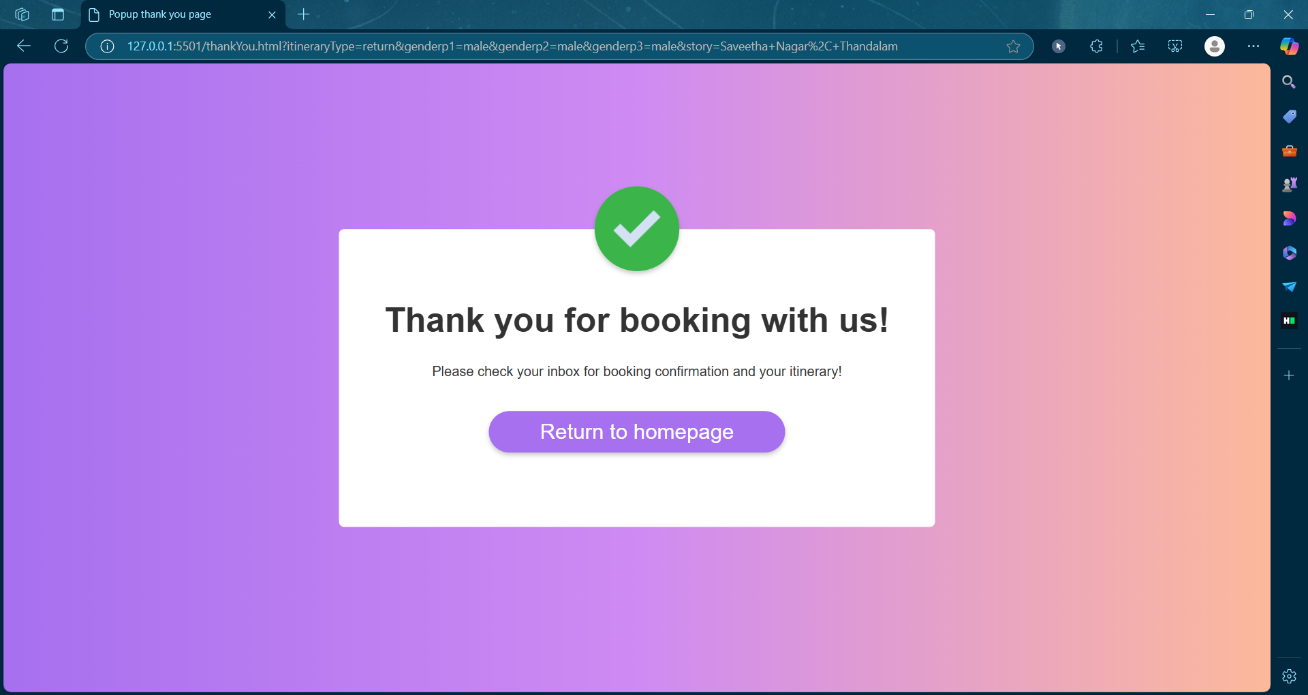
1. **Concurrency Issues**
   * 1. **Challenge**: Managing thousands of concurrent users during peak times.
     2. **Solution**: Implemented load balancing and optimized database queries to reduce latency.
2. **API Rate Limits**
   * 1. **Challenge**: Limited calls for event data from external APIs.
     2. **Solution**: Caching mechanisms and fallback data display during API downtime.
3. **Responsiveness**
   * 1. **Challenge**: Ensuring consistent UX across different devices.
     2. **Solution**: Used **media queries** and rigorous testing on various screen sizes.
4. **Payment Security**
   * 1. **Challenge**: Preventing fraud and ensuring secure transactions.
     2. **Solution**: Integrated encrypted payment gateways and implemented fraud-detection mechanisms.

**6. Ethics, UX, and Sustainability**

* + **Ethics**
    1. Ensured compliance with **GDPR** and other data protection regulations.
    2. Protected user privacy by encrypting sensitive data.
  + **User Experience (UX)**
    1. Conducted user testing sessions to gather feedback.
    2. Improved navigation and search functions based on feedback.
  + **Sustainability**
    1. Used energy-efficient cloud hosting solutions.
    2. Promoted digital tickets to reduce paper usage.

**7. User Interface**





**Conclusion**

The real-time **Ticket Booking System** demonstrates the effectiveness of using Agile methodologies with the Kanban framework for efficient project management and development. It addresses common issues in ticket booking platforms, offering a scalable, responsive, and user-friendly solution. Challenges such as high concurrency, API rate limits, and payment security were tackled effectively, ensuring a robust platform. The use of cloud deployment, modular architecture, and continuous delivery further highlights the project's alignment with modern software engineering practices.

**References:**

1. **"Agile Estimating and Planning"** – Mike Cohn, Prentice Hall, 2005.
2. **"Kanban: Successful Evolutionary Change for Your Technology Business"** – David J. Anderson, Blue Hole Press, 2010.
3. **"HTML and CSS: Design and Build Websites"** – Jon Duckett, Wiley, 2011.
4. **"Building Microservices"** – Sam Newman, O'Reilly Media, 2015.
5. **"Don't Make Me Think: A Common Sense Approach to Web Usability"** – Steve Krug, New Riders, 2000.