PURBANCHAL UNIVERSITY

**Biratnagar Nepal**

****

A Project report on

**“BakedBeats”**

In the partial fulfillment for the requirement of the 4th Semester Project-IV (BIT 256 CO) in the completion of **Bachelor of Information Technology (BIT)** degree at **KIST College of Information Technology**, under **Purbanchal University.**

**Submitted By: - Submitted To: - 1)Devashish Poudel Purbanchal University**

**2)Dip Kumar Gyawali**

**3) Swastik Thapa**

**Under The Guidance of**

**Mr. Roshan Shrestha**

**Project Teacher**

**KIST COLLEGE OF INFORMATION AND TECHNOLOGY KAMALPOKHARI, KATHMANDU**

**KIST COLLEGE OF INFORMATION AND TECHNOLOGY**

**KAMALPOKHARI, KATHMANDU**

****

**CERTIFICATE**

This is to certify that the project work entitled **“BAKEDBEATS”** is carried out **by DEVASHISH POUDEL(6088), DIP KUMAR GYAWALI(6064), SWASTIK THAPA (6093),** bona fide students of **KIST COLLEGE OF INFORMATION AND TECHNOLOGY** in partial fulfillment for the award of **BACHELOR IN INFORMATION AND TECHNOLOGY** of the **PURBANCHAL UNIVERSITY, BIRATNAGAR NEPAL**, during the year **2024**. It is certified that all corrections indicated for internal assessment have been incorporated in the report submitted in the department library. The project report has been approved, as it satisfied the academic requirements in respect of the project work prescribed for the said degree.

The details of the students are as follows: -

**NAME REGISTRATION NO. SYMBOL NO.**

Devashish Poudel 058-3-2-03789-2021 345489

Dip Kumar Gyawali 058-3-2-03791-2021 340385

Swastik Thapa 058-3-2-03815-2021 340408

Course: - BIT (Bachelor’s in Information Technology)

Semester: - 4th Semester

Subject: - Project-IV

Subject Code: - BIT 256 CO



Mr. Deepak Khadka (Co-ordinator)

**KIST COLLEGE OF INFORMATION TECHNOLOGY KAMALPOKHARI, KATHMANDU**

****

**Examiner’s Certification**

The Project Report

On

**“BakedBeats”**

**Developed by:**

**Devashish Poudel**

**Dip Kumar Gyawali**

**Swastik Thapa**

Is approved and is acceptable in qualified form.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Internal Examiner External Examiner** Name: Roshan Shrestha Name:

Designation: Project Teacher Designation:

**ACKNOWLEDGEMENT**

It is with greatest satisfaction and euphoria that we are submitting our project report entitled **“BAKEDBEATS”.** We have completed it as a part of the curriculum of **PURBANCHAL UNIVERSITY.**

We also take this opportunity to express a deep sense of gratitude to our **BIT Coordinator Mr. Deepak Khadka** and **Project Teacher Mr. Roshan Shrestha** for their amiable support, valuable information and guidance which helped us in completing this task throughout its various stages. We are indebted to all members of **KIST College,** for the valuable support and suggestions provided by them using their specific fields’ knowledge. We are grateful for their cooperation during the period of our project.

Finally, we would also like to express our gratitude towards **Purbanchal University** for designing such a wonderful course structure. It will help us to get more knowledge in the field of Information Technology & help us to have a bright future in the field of technology.

We hope our university will accept this attempt as a successful project.

Last but not the least, our sincere thanks to our parents, teaching and non-teaching staff of our college and our friends.

**DEVASHISH POUDEL (340383)**

**DIP KUMAR GYAWALI(340385)**

**SWASTIK THAPA (340408)**

**ABSTRACT**

This project aims to develop a Java-based application that transforms personal computers into virtual musical instrument platforms, catering primarily to younger generations. By providing a user-friendly interface, the application will offer a diverse range of virtual instruments, allowing users to explore, learn, and experiment with music creation. The software will leverage Java's capabilities for audio processing and graphical user interface development to create an immersive and engaging musical experience. Through interactive features and gamification elements, the application seeks to foster creativity, musical aptitude, and digital literacy among young users.

**LIST OF FIGURES**

**Figure 1.1:** Spiral Model in SDLC

**Figure 2.1:** Flowchart of Login Page

**Figure 2.2:** Flowchart of BakedBeats

**Figure 2.3:** Use Case Diagram

**Figure 2.4:** Gantt Chart

**TABLE OF CONTENTS**

**Approval Certificate**

**Acknowledgements**

**Abstract**

**List of Figures**

**Chapter 1: Introduction**

1.1 Background

1.2 Problem Statement

1.3 Project Objectives

1.4 Project Scope

1.4.1 Additional Features

1.5 Project Limitation

**Chapter 2: Literature Review**

2.1 Background

2.2 Functional and Non-functional requirement

2.3 Feasibility Study

**Chapter 3: System Design & Methodology**

3.1 Software Development Life Cycle

3.1.1 Used Model

3.2 Flowchart

**Chapter 4: Implementation**

4.1 Language used

4.1.1 Key Characteristics of JAVA

4.1.2 Application of JAVA

4.1.3 Core Components of JAVA

**Chapter 5: Conclusion & Future Recommendation**

5.1 Conclusion

5.2 Future Recommendation

**References**

**Appendices**

**Chapter 1**

**Introduction**

**1. Introduction**

The **“BakedBeats”** is an innovative software application designed to provide a dynamic and engaging platform for music education and exploration, specifically targeting younger generations. By leveraging the capabilities of modern computing, this project seeks to transform personal computers into versatile musical instrument platforms, offering a fun and interactive way to learn and create music.

**1.2 Problem Statement**

Despite the increasing accessibility of digital technology, many children and young adults lack engaging and affordable music education resources. Traditional music lessons can be expensive and inaccessible to many, while existing digital music learning platforms often fall short in terms of interactivity, comprehensiveness, and user experience. There is a clear need for a music learning application that is both fun and effective, catering to the needs and preferences of younger generations. The Virtual Instrument Playground aims to address this gap by providing a free, interactive, and comprehensive platform for music exploration and learning.

**1.3 Objectives**

The primary objective of the Virtual Instrument Playground is to democratize music education by making it accessible and enjoyable for a wide range of users. To achieve this, the project aims to:

* **Create a diverse instrument library:** Develop a comprehensive collection of virtual instruments, spanning various categories such as keyboards, drums, guitars, wind instruments, and orchestral instruments. This will allow users to explore different musical styles and genres.
* **Develop an intuitive user interface:** Design a user-friendly interface that is easy to navigate, even for young children. The interface should be visually appealing and engaging, encouraging exploration and experimentation.
* **Incorporate educational elements:** Integrate music theory concepts, rhythm training, and ear training exercises into the application to support learning and skill development.
* **Foster creativity and expression:** Provide tools and features that empower users to compose their own music, experiment with different sounds, and share their creations with others.
* **Enhance accessibility:** Ensure that the application is inclusive by providing options for users with disabilities, such as adjustable font sizes, color schemes, and audio cues.

**1.4 Project Scope**

* **Virtual Instrument Library:** Development of a foundational set of virtual instruments, including a piano drum guitar xylophone bass guitar and a mechanical keyboard.
* **User Interface:** Creation of a user-friendly interface with intuitive controls for instrument selection, playing, and sound adjustments.
* **Audio Engine:** Implementation of a basic audio engine capable of producing acceptable instrument sounds.
* **Basic Educational Features:** Inclusion of simple tutorials and exercises covering basic music theory concepts and rhythm training.

### 1.4.1 Additional Features (to be considered in future phases)

* **Expanded Instrument Library:** Addition of more instruments, such as wind instruments, string instruments, and percussion.
* **Advanced Audio Engine:** Improvement of audio quality and realism.
* **Comprehensive Educational Modules:** Development of in-depth music theory and ear training courses.
* **Multiplayer Functionality:** Enabling users to collaborate and play together.

**1.5 Project Limitation**

Project limitations are the constraints or restrictions that impact the project's scope, goals, or deliverables. Identifying limitations upfront helps manage expectations and focus development efforts.

* **Resource Constraints:** Limited development time, budget, and human resources can restrict the project's scope and complexity.
* **Technical Challenges:** Developing realistic instrument sounds, ensuring cross-platform compatibility, and optimizing performance can present technical hurdles.
* **Educational Content Development:** Creating effective and engaging educational materials requires expertise in music education and instructional design.
* **User Interface Design:** Designing a user-friendly interface that caters to a wide age range can be challenging.
* **Platform Compatibility:** Ensuring compatibility across different operating systems and devices can require additional development effort.

By acknowledging these limitations, we can prioritize development efforts, allocate resources effectively, and plan for potential challenges.

**Chapter-2**

**Literature Review**

**2.1 Background**

This project has been thoroughly researched and put together by the use of JAVA. **“BakedBeats”** is a JAVA based application that incorporates multiple instruments to be played on a computer with the help of a keyboard. This software aims to give the user the experience of playing multiple instruments by the use of just one device. Similar projects such as Garage Band, GuitarTuna, Band Labs were researched and taken inspiration from for the development of this project.

**2.2 Functional and Non- Functional Requirements**

The Requirements for the proper functioning of this software are as follows:

**REQUIREMENTS**

**Hardware:**

**Memory (RAM)**: 4 GB/ 6GB

**System Type:** 64-bit OS, x64 -based processor

**Storage Capacity:** 30 GB HDD

**CPU:** 2.30 GHZ

**Software:**

**Operating System:** Windows 10 or Higher

**Development Tools:** JAVA 19, IntelliJ IDEA

**Database:** MySQL

**2.3 Feasibility Study**

This feasibility study evaluates the potential of developing a musical instrument-based Java application. The application aims to provide a virtual musical instrument experience, accessible to users through various platforms.

### Market Feasibility

* Target Market: The target market includes music enthusiasts, learners, and casual users.
* Market Size: The global music software market is substantial and growing.
* Competition: There are numerous music applications available, but there's room for innovation and unique features.
* Market Trends: There's a growing trend towards digital music consumption and creation.

### Technical Feasibility

* Java Compatibility: Java is suitable for developing complex applications, including audio processing.
* User Interface: JavaFX or Swing can be employed for creating intuitive interfaces.
* Platform Compatibility: The application can be developed for desktop, web, and mobile platforms.
* Development Resources: Necessary skills and resources (developers, sound engineers) are available.

### Financial Feasibility

* Development Costs: Costs include software licenses, hardware, developer salaries, and marketing expenses.
* Revenue Generation: Potential revenue streams include app sales, in-app purchases, subscriptions, and advertising.

### Legal Feasibility

* Copyright and Licensing: Ensure compliance with copyright laws for sound samples and musical works.
* Intellectual Property: Protect the application's intellectual property through patents or copyrights.

### Risk Assessment

* Technical Challenges: Sound quality issues, performance optimization, and compatibility problems.
* Market Risks: Competition, changing market trends, and user acceptance.
* Financial Risks: Insufficient funding, higher-than-expected costs, and lower-than-expected revenue.

**Chapter 3**

**System Development & Methodology**

**3.1 Software Development Life Cycle**

**SDLC** stands for **Software Development Life Cycle**. It's a structured process that a development team follows to create, test, and deploy software. Think of it as a roadmap that guides the project from inception to delivery and beyond.

### Key Phases of SDLC

While specific SDLC models vary, the core phases typically include:

1. **Planning:**
   * Defining project goals, objectives, and scope.
   * Identifying target audience and their needs.
   * Creating a project plan, including timelines and resources.
   * Conducting feasibility studies.
2. **Requirements Gathering:**
   * Collecting and documenting user requirements.
   * Creating detailed specifications for the software.
   * Prioritizing requirements based on importance and feasibility.
3. **Design:**
   * Creating the software architecture.
   * Designing the user interface (UI) and user experience (UX).
   * Developing data models and database structures.
4. **Development/Coding:**
   * Writing the actual code for the software.
   * Building the software components.
   * Integrating different modules.
5. **Testing:**
   * Identifying and fixing bugs and errors.
   * Ensuring the software meets the specified requirements.
   * Conducting different types of testing (unit, integration, system, acceptance).
6. **Deployment:**
   * Releasing the software to the market or end-users.
   * Installing the software on user systems.
   * Configuring the software for the target environment.
7. **Maintenance:**
   * Providing support and updates to the software.
   * Fixing bugs and issues reported by users.
   * Implementing new features and enhancements.

**3.1.1 Used Model**

The Spiral Model is a software development lifecycle (SDLC) model that combines the iterative development process with the systematic, controlled aspects of the waterfall model. It's particularly suited for large, complex projects where risk management is crucial.

### 

### 

### Figure 1.1 : Spiral Model in SDLC

### 

### 

### 

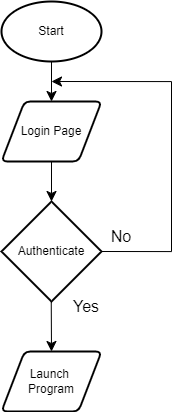
### How it Works

Imagine a spiral. Each loop of the spiral represents a phase in the development process. As the project progresses, it moves through multiple iterations of this spiral.

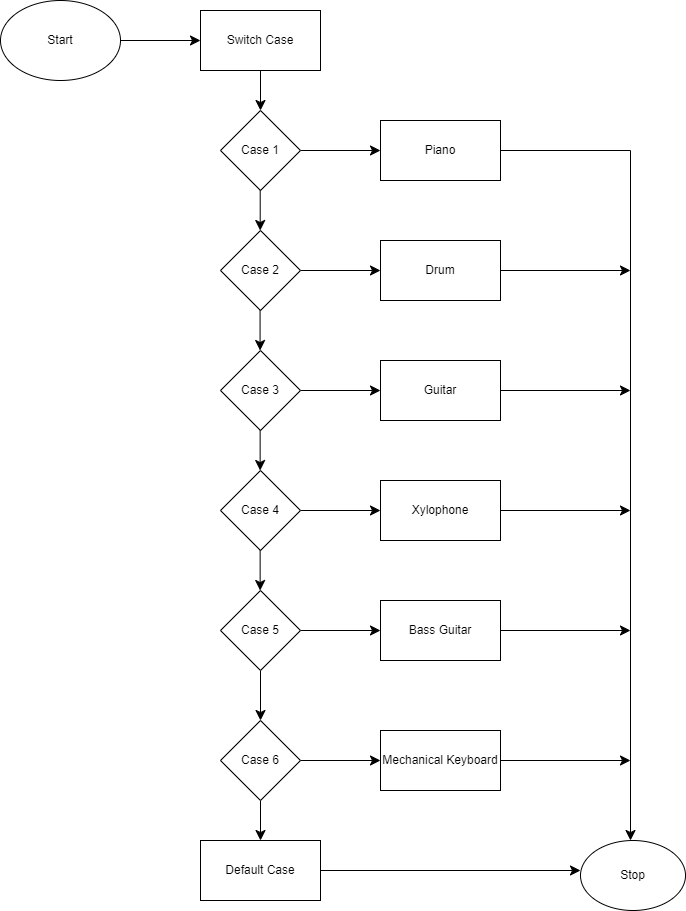
**Key Phases of the Spiral Model:**

1. **Planning:**
   * Define project goals and objectives.
   * Identify potential risks.
   * Create a project plan for the next phase.
2. **Risk Analysis:**
   * Analyze identified risks.
   * Develop strategies to mitigate risks.
   * Prioritize risks based on their impact.
3. **Engineering:**
   * Develop and test the product.
   * Create prototypes or product increments.
   * Gather feedback from stakeholders.
4. **Evaluation:**
   * Evaluate the product and the project.
   * Identify lessons learned.
   * Plan for the next iteration.

**3.2 Flowchart**

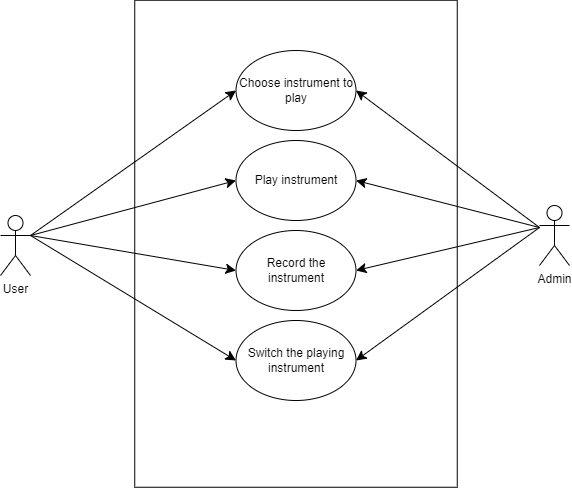
****

**Figure 2.1 : Flowchart of Login Page**

****

**Figure 2.2 : Flowchart of Instrument Selection**

**3.3 Use Case Diagram**

****

**Figure 2.4: UseCase Diagram of BakedBeats**

**Chapter 4**

**Implementation**

**4.1 Language Used**

**Java** is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. This means that compiled Java code can run on all platforms that support Java without the need for recompilation. This concept is often referred to as "write once, run anywhere" (WORA).

### 4.1.1 Key Characteristics of Java

* **Object-Oriented:** Java is based on the concept of objects, which encapsulate data and behavior.
* **Platform Independence:** Java code can run on different operating systems (Windows, macOS, Linux) thanks to the Java Virtual Machine (JVM).
* **Robust:** Java incorporates features like exception handling, garbage collection, and strong type checking to prevent errors and enhance reliability.
* **Secure:** Java has built-in security features to protect against viruses and malicious code.
* **High Performance:** While not as fast as languages like C++, Java offers good performance through just-in-time compilation and optimization techniques.

### 4.1.2 Applications of Java

Java is used in a wide range of applications, including:

* **Enterprise Applications:** Building large-scale, distributed systems for businesses.
* **Android App Development:** Creating mobile applications for the Android operating system.
* **Embedded Systems:** Developing software for devices with limited resources.
* **Scientific Computing:** Performing complex calculations and simulations.

### 

### 

### 

### 4.1.3 Core Components of Java

* **Java Virtual Machine (JVM):** The runtime environment that executes Java bytecode.
* **Java Development Kit (JDK):** A software development environment that includes the compiler, debugger, and other tools for Java development.
* **Java Runtime Environment (JRE):** The runtime environment necessary to run Java applications.

**Chapter 5**

**Conclusion & Future Recommendation**

**5.1 Conclusion**

Our project represents a modest initiative aimed at meeting the demands of music players. We have incorporated user-friendly coding to develop a software package that promises to be a robust solution for fulfilling the requirements of the music player service. The software planning's primary goal is to establish a framework that allows the manager to make reasonable estimates within a constrained time frame at the project's outset, with periodic updates throughout its development. We have successfully crafted a computerized music player system, which can securely store and effortlessly retrieve music records. This system encompasses data entry capabilities for tracks and artists, in addition to facilitating seamless playlist creation and music playback. The interconnected components of the music library, user preferences, and playback controls ensure the system's accuracy and efficiency.

**5.2 Future Recommendations**

Since this project is only the start of a new application, many things can be improved. In the future, we're planning to make our music player even better by doing a few key things. First, we'll make the sound quality even nicer by using better technology. Then, we'll make it easier for you to create, change, and share playlists. You'll be able to share your favorite songs and playlists on social media too. And, to make it even easier to use, we're thinking about adding voice control so you can tell the music player what to do without touching it. These improvements will make our music player more enjoyable and user-friendly.

**References**

Website: [www.virtualmusicalinstruments.com/](http://www.virtualmusicalinstruments.com/)

Youtube: <https://youtu.be/DFnUDyzF1Uk>

Google: [www.google.com](http://www.google.com)

GeeksForGeeks: [www.geeksforgeeks.com](http://www.geeksforgeeks.com)